Efficient and Accurate Numerical Methods for Fluid-Structure and Fluid-Particle Interactions

by

Yuan Xun Bao

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Mathematics

New York University

September 2018

Professor Aleksandar Donev

Professor Leslie Greengard

Dedication

To my family.

Acknowledgements

I will need more time to write acknowledgments. But first and foremost, I would like to thank the committee members for reviewing my dissertation.

Abstract

We present two efficient and accurate numerical methods for simulating elastic and rigid structures immersed in fluids. In the first part, we focus on the Immersed Boundary (IB) method for studying fluid-structure interaction in problems involving an elastic structure immersed in a viscous incompressible fluid at finite Reynolds numbers. It is well-known that the conventional IB method suffers from poor volume conservation. This arises because the interpolated Lagrangian velocity is not generally divergence-free. We develop a Divergence-Free Immersed Boundary (DFIB) method that substantially reduces the volume loss in an immersed body as it moves and deforms in the process of interacting with the fluid. We introduce a new velocity-interpolation scheme with the property that the interpolated velocity field in which the structure moves is continuously differentiable, and satisfies a continuous divergence-free condition. We also develop a new force-spreading scheme that is the adjoint of the velocity-interpolation operator. We confirm through numerical experiments in two and three spatial dimensions that DFIB is able to achieve substantial improvement in volume conservation compared to other existing IB methods, at the expense of a modest increase in the computational cost. Furthermore, the new method provides smoother Lagrangian forces (tractions) than traditional IB methods.

In the second part, we present a fluctuating boundary integral method (FBIM) for overdamped Brownian Dynamics of two-dimensional periodic suspensions of rigid particles immersed in a Stokes fluid. At small scales and low Reynolds numbers, the motion of immersed particles is strongly influenced by thermal fluctuations, giving rise to Brownian motion strongly correlated with hydrodynamic effects. We develop a novel approach for generating Brownian displacements that arise in response to the thermal fluctuations in the fluid. Our approach relies on a first-kind boundary integral formulation of a mobility problem in which a random surface velocity is prescribed on the particle surface, with zero mean and covariance proportional to the Greens function for Stokes flow (Stokeslet). This approach yields an algorithm that scales linearly in the number of particles for both deterministic and stochastic dynamics, handles particles of complex shape, achieves high order of accuracy, and can be generalized to three dimensions and other boundary conditions. We show that Brownian displacements generated by our method obey the discrete fluctuation-dissipation balance relation (DFDB). FBIM provides the key ingredient for time integration of the overdamped Langevin equations for Brownian suspensions of rigid particles. We demonstrate that FBIM obeys DFDB by performing equilibrium BD simulations of suspensions of starfish-shaped bodies using a random finite difference temporal integrator.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Fluid-Structure Interaction (FSI) is a broad subject that spans an incredible range of length scales in science and engineering applications, for example, ranging from tens of meters in wind turbine modeling, to a few centimeters in swimming and flying animals, to several millimeters in diameter for blood flows in human aorta, to a few micrometers for an individual cell, self-propelled micro-swimmers and colloidal particles, and further down to nanometer scale for macromolecules and Brownian motors. FSI is of significant interest in many scientific disciplines, however, due to its strong nonlinearity, classical analytical methods are only applicable in certain idealized settings. Although the vast majority of scientific discovery in FSI are pioneered by experiments, the cost in time and resources of exploring large parameter space makes reproducing such experiments a daunting task, and thus, severely limits the repeatability and scalability of experimental approaches.

With the increasing speed of CPUs, GPUs and large parallel computers, modeling FSI through computer simulations has become a practical approach, and has been popular among scientists and engineers for decades. However, developing an efficient and accurate numerical method that captures the multiphysics and multiscale nature of FSI still remains a formidable challenge due to the nonlinear coupling between the fluids and immersed structures, and is further complicated by the presence of complex moving geometries.

Various types of numerical methods for simulating different FSI problems have been developed in the past few decades. A general classification of FSI algorithms is based on the treatment of the underlying computational grids: *conforming-mesh* (body-fitted grids) versus *non-conforming mesh* approaches. In the field of continuum mechanics, methods based on Finite Element (FE) and conforming meshes are commonly employed to study fluid-solid coupling. In these methods, physical boundary conditions are imposed at the fluid-solid interface, and a computational mesh that conforms to the interface is used for discretization. Although a conforming mesh can handle arbitrary complex moving geometries, a major drawback is attributed to its excessive computational cost for re-meshing when the interface experiences large deformations. Furthermore, it requires sophisticated procedures to avoid severe mesh distortions in order to achieve desired accuracy. On the other hand, methods based on non-conforming meshes avoid re-meshing by treating the computational domains for fluids and immersed bodies separately via adding additional forcing or constraints to the governing equations. Representative examples that fall into this category include the *Immersed Boundary* (IB) method and its variant, the *Immersed Interface Method* (IIM).

The first part of this dissertation presents a novel IB method that improves the conventional IB method in the volume (or mass) conservation. The IB method is both a mathematical and a computational framework for simulating FSI problems. Although there are extensions to the conventional IB method to account for solids or rigid bodies, here we focus on its primitive form by assuming a closed, neutrally-buoyant and *elastic* body immersed in a viscous incompressible fluid at finite (intermediate or moderately high) Reynolds numbers. The IB formulation uses an Eulerian representation of the fluid and a Lagrangian representation of the structure. The Lagrangian and Eulerian frames are coupled by integral transforms with delta function kernels. The discretized fluid-structure coupling equations use approximations to these transforms with a regularized delta function kernel to interpolate the fluid velocity to the structure, and to spread structural forces to the fluid.

Despite its wide applicability and ease of implementation, the conventional IB method can be poor of volume (mass) conservation. This arises because the interpolated velocity at which the Lagrangian structure moves does not satisfies a *continuous* divergence-free condition, even though the discrete fluid velocity is enforced to be *discretely* divergence-free by the fluid solver. In chapter 2, we develop a novel *Divergence-Free Immersed Boundary* (DFIB) method that substantially improves the volume loss in the immersed body compared to conventional IB methods. We introduce new velocity-interpolation scheme with the property that the interpolated Lagrangian velocity is continuously differentiable and satisfies the continuous divergence-free condition. To preserve the energy transfer between the Eulerian-Lagrangian interactions, we also develop a new force-spreading scheme that is the adjoint of the velocity interpolation scheme.

The second part of this dissertation is concerned with fluid-structure interaction in low Reynolds number flows. More specifically, we are interested in the collective motion of suspension of micrometer-sized *rigid* particles immersed in a viscous fluid. At small scales and in the limit of zero Reynolds numbers, the motion of immersed particles is strongly influenced by thermal fluctuations in the fluid, giving rise to Brownian motion strongly correlated with hydrodynamic effects. Therefore, the two key ingredients that need to be included in a computational method for particle suspensions are the long-ranged hydrodynamic interactions (HIs) among *all* particles and physical boundaries, and the correlated Brownian motion of particles.

In the absence of Brownian motion, describing the hydrodynamics of particulate suspension requires solving the Stokes mobility problem, i.e., computing the linear and angular velocities of the particles in response to applied (external) forces and torques. For passive suspensions of spherical particles, the method of regularized Stokeslet, Brownian (BD) and Stokesian Dynamics (SD), and Force-Coupling Method (FCM) are commonly used to capture the far-field behavior of the hydrodynamic interactions. Although modern fast algorithms can solve the deterministic Stokes problem with linear-scaling by using the Fast Multipole Method (FMM) for an unbounded domain, and using Ewald-like methods for periodic and confined domains, Brownian (stochastic) displacements of particles are typically generated generated iteratively by a Chebyshev polynomial approximation method, or by the Lanczos algorithm with super-linear scaling as the number of particles grows.

Essentially all commonly-used methods are limited to spherical particles, and generalization to include particles of complex shape is generally difficult. Furthermore, these methods employ an uncontrolled truncation of a multipole expansion hierarchy and therefore become inaccurate when particles get close to one another as in dense suspensions.

For deterministic Stokes problems, the Boundary Integral Method (BIM) [1] is very well-developed [2, 3, 4] and allows one to handle complex particle shapes and achieve *controlled accuracy* even for dense suspensions [5, 6]. In the boundary integral framework, the steady Stokes mobility problem is reformulated as an integral equation of unknown densities that are defined on the boundary, using a first-kind (single-layer densities) or second-kind (double-layer densities) formulation, or a mixture of both. Suspended particles of complex geometry can be directly discretized by a surface mesh, and by a suitable choice of surface quadrature higher-order, or even spectral accuracy, can be achieved.

The current development of BIM for particle suspensions is limited to the deterministic case only. In chapter 3 we present a fluctuating boundary integral method (FBIM) for overdamped Brownian Dynamics (BD) of two-dimensional periodic suspensions of rigid particles of complex shape immersed in a Stokes fluid. To the best of our knowledge, this is the first boundary integral method that accounts for Brownian motion of nonspherical particles. Our approach relies on a first-kind boundary integral formulation of a mobility problem in which a random surface velocity is prescribed on the particle surface, with zero mean and covariance proportional to the Green's function for Stokes flow (Stokeslet). This approach yields an algorithm that scales linearly in the number of particles for both deterministic and stochastic dynamics, handles particles of complex shape, achieves high order of accuracy, and can be generalized to three dimensions and other boundary conditions. We show that Brownian displacements generated by our method obey the discrete fluctuation-dissipation balance relation (DFDB).

Because of the distinct nature of the two subjects discussed in this dissertation, we give more detailed motivation and background knowledge in the introduction section of each chapter.

Material presented in this dissertation has previously appeared in the following peer-reviewed publications/preprint:

- Y. Bao, J. Kaye, C. S. Peskin, A gaussian-like immersed-boundary kernel with three continuous derivatives and improved translational invariance, Journal of Computational Physics 316 (2016) 139 – 144, software and updated documentation available at `https://github.com/stochasticHydroTools/IBMethod`, including also a new 5-pt kernel with three continuous derivatives. `doi:http://dx.doi.org/10.1016/j.jcp.2016.04.024`. URL `http://www.sciencedirect.com/science/article/pii/S0021999116300663`

- Y. Bao, A. Donev, B. E. Griffith, D. M. McQueen, C. S. Peskin, An Immersed Boundary Method with Divergence-Free Velocity Interpolation and Force Spreading, Journal of Computational Physics 347 (2017) 183–206. `doi:http://dx.doi.org/10.1016/j.jcp.2017.06.041`

- Y. Bao, M. Rachh, E. Keaveny, L. Greengard, A. Donev, A fluctuating boundary integral method for Brownian suspensions, submitted to J. Comp. Phys., preprint ArXiv:1709.01480 (2017)

Chapter 2

An Immersed Boundary Method with Divergence-Free Velocity Interpolation and Force Spreading

The Immersed Boundary (IB) method is a mathematical framework for constructing robust numerical methods to study fluid-structure interaction in problems involving an elastic structure immersed in a viscous incompressible fluid. The IB formulation uses an Eulerian representation of the fluid and a Lagrangian representation of the structure. The Lagrangian and Eulerian frames are coupled by integral transforms with delta function kernels. The discretized IB equations use approximations to these transforms with regularized delta function kernels to interpolate the fluid velocity to the structure, and to spread structural forces to the fluid. It is well-known that the conventional IB method suffers from poor volume conservation. This arises because the interpolated Lagrangian velocity is not generally divergence-free. In this chapter we present a Divergence-Free Immersed Boundary (DFIB) method that susbstantially reduces the volume loss in an immersed body as it moves and deforms in the process of interacting with the fluid. We introduce a new velocity-interpolation scheme with the property that the interpolated velocity field in which the structure moves is continuously differentiable, and satisfies a continuous divergence-free condition. We also develop a new force-spreading scheme that is the adjoint of the velocity-interpolation operator. We present a new family of regularized delta functions (IB kernels) that have three continuous derivatives and improved translational invariance. We confirm through numerical experiments in two and three spatial dimensions that this new IB method is able to achieve substantial improvement in volume conservation compared to other existing IB methods, at the expense of a modest increase in the computational cost. Furthermore, the new method provides smoother Lagrangian forces (tractions) than traditional IB methods.

2.1 Introduction

The *Immersed Boundary* (IB) method [10] is a general mathematical framework for the numerical solution of *fluid-structure interaction* problems arising in biological and engineering applications. The IB method was introduced to simulate flow patterns around the heart valves [11, 12], and since its success in modeling cardiac fluid dynamics [13, 14, 15, 16], it has been extended and applied to various other applications, including but not limited to motion of biological swimmers [17, 18], dynamics of red-blood cells [19] and dry foam [20, 21], and rigid body motion [22, 23].

The essence of the IB method as a numerical scheme lies in its simple way of coupling an Eulerian representation of the fluid and a Lagrangian representation of the structure. The *force spreading* linear operator $\boldsymbol{S}$ that spreads forces (stresses) from the structure to the fluid and the *velocity interpolation* linear operator $\boldsymbol{S}^\star$ that interpolates velocities from the fluid to the structure are carried out via a regularized delta function δ_h. One effective way to construct δ_h is to require the regularized delta function to satisfy a set of moment conditions to achieve approximate grid translation-invariance and desired interpolation accuracy [7], thereby avoiding special grid treatment near the fluid-structure interface. In spite of its wide applicability and ease of implementation, the conventional IB method with a collocated-grid discretization (referred to herein as IBCollocated) has two well-known shortcomings in accuracy: it achieves only first-order convergence for problems that possess sharp-interface solutions [24, 25], and it can be relatively poor of volume conservation [26]. Much research effort has been put into improving the convergence rate of the IB method to second order or even higher order for problems with singular forcing at the sharp interface. Notable examples include, the *Immersed Interface Method* (IIM) [27, 28], and more recently, a new method known as *Immersed Boundary Smooth Extension* [29, 30]. Our focus here, however, is on improving the volume conservation properties of the IB method.

As an immediate consequence of fluid incompressibility, which is one of the basic assumptions of the IB formulation, the volume enclosed by the immersed structure is exactly conserved as it deforms and moves with the fluid in the continuum setting. Thus, a desirable feature of an IB method is to conserve volume as nearly as possible. In practice, however, it is observed that, even in the simplest case of a quasi-static pressurized membrane [31], the conventional IB method (regardless of collocated- or staggered-grid discretization) produces volume error that persistently grows in time, as if fluid "leaks" through the boundary. An intuitive explanation for this "leak" is that fluid is "squeezing" between the marker points used to discretize the boundary in a conventional IB method; however, this is *not* the full story, because refining the Lagrangian discretization does not improve the volume conservation of the method for a fixed Eulerian discretization.

In the conventional IB method, we can extend the notion of velocity interpolation to any point in the domain (not restricted to the immersed structure), denoted here with an italic $\boldsymbol{X}$. The *continuous* interpolated velocity field can be written as $\boldsymbol{U}(\boldsymbol{X}) = (\boldsymbol{\mathcal{J}}\mathbf{u})(\boldsymbol{X})$, where $\boldsymbol{\mathcal{J}}$ denotes the *continuous* interpolation operator that interpolates the velocity at $\boldsymbol{X}$ from the discrete fluid velocity $\mathbf{u}$. If a closed surface moves with velocity that is *continuously* divergence-free with respect to the *continuum* divergence operator, i.e., $(\boldsymbol{\nabla} \cdot \boldsymbol{U})(\boldsymbol{X}) = 0$, then the volume enclosed by the (deformed) surface will be exactly conserved. However, in the discrete setting, even if the interpolated velocity field is *continuously* divergence-free, exact volume conservation is generally not achieved because of the time-stepping error from the temporal integrator. Another source of error comes from discretizing the surface itself. In the IB method, only a discrete collection of points on the surface, i.e., the Lagrangian markers, move according to the interpolated velocity field. A closed discretized surface can be constructed by simply connecting the Lagrangian markers defining a facet, and the resulting faceted surface by this construction does not enclose a constant volume. In the absence of temporal integration errors, this kind of volume-conservation error will approach zero as the discretization of the surface is refined. Peskin and Printz realized that the major cause of poor volume conservation of IBCollocated is that the *continuous* interpolated velocity field given by the conventional IB interpolation operator (denoted by $\boldsymbol{\mathcal{J}}_{\mathrm{IB}}$) is not *continuously* divergence-free [26], despite that the discrete fluid velocity is enforced to be *discretely* divergence-free with respect to the *discrete* divergence operator by the fluid solver.

To improve the volume conservation of the conventional IB method, Peskin and Printz proposed a modified finite-difference approximation to the discrete divergence operator to ensure that the *average* of the continuous divergence of the interpolated velocity is equal to zero in a small control volume with size of a grid cell [26]. Their IB method with modified finite-difference operators (herein referred to as IBModified) was applied to a two-dimensional model of the heart, and it achieved improvement in volume conservation by one-to-two orders of magnitude compared to IBCollocated. Nevertheless, a major drawback of IBModified that limits its use in applications is its complex, non-standard finite-difference operators that uses coefficients derived from the regularized delta function (but see [20, 21] for applications). To address the issue of spurious currents across immersed structure supporting extremely large pressure differences, Guy and Strychalski [32] developed a different extension of the

IB method that uses non-uniform Fast Fourier Transform [33, 34] (NUFFT) to generate "spectral" approximations to the delta function, which also has superior volume conservation.

Over the past two decades, the staggered-grid (MAC) discretization has been widely adopted by the IB community [14, 16, 17, 22, 23, 35]. In addition to its most celebrated feature of avoiding the odd-even decoupling in the Poisson solver that can otherwise occur with collocated-grid discretization, which leads to "checkerboard" instability in the solutions, Griffith [31] concluded from his numerical studies that the improvement in volume conservation of the IB method with staggered-grid discretization (IBMAC) is essentially the same as that of IBModified. In practice, IBMAC is more practical than IBModified in that the improvement in volume conservation directly comes as a byproduct of grid discretization without any modification to the finite-difference operators, and it is relatively straightforward to extend IBMAC to include adaptive mesh refinement [14, 36] and physical boundary conditions [37]. However, we emphasize that the nature of Lagrangian velocity interpolation of IBMAC remains the same as that of IBCollocated, and, hence, there is much room for further improvement in volume conservation by ensuring that the interpolated velocity is constructed to be nearly or exactly divergence-free. We note that the methods designed to improve the convergence rate of IB methods, such as IIM [27, 28] and the Blob-Projection method [38], also improve volume conservation, because the solution near the interface is computed more accurately. These methods, however, are somewhat more complex and less generalizable than the conventional IB method.

This chapter is concerned with further improving volume conservation of IBMAC by constructing a *continuous* velocity-interpolation operator $\boldsymbol{\mathcal{J}}$ that is divergence-free in the *continuous* sense. The discrete IB interpolation operator $\boldsymbol{S}^{\star}$ is simply the restriction of $\boldsymbol{\mathcal{J}}$ to the Lagrangian markers. The key idea introduced in this chapter is first to construct a *discrete vector potential* that lives on an *edge-centered* staggered grid from the discretely divergence-free fluid velocity, and then to apply the conventional IB interpolation scheme to obtain a *continuum* vector potential, from which the interpolated velocity field is obtained by applying the continuum curl operator. Note that the existence of the discrete vector potential relies on the fact that the discrete velocity field is discretely divergence-free. The interpolated velocity field obtained in this manner is guaranteed to be continuously divergence-free, since the divergence of the curl of any vector field is zero. We also propose a new force-spreading operator $\boldsymbol{S}$ that is defined to be the new adjoint of the interpolation operator $\boldsymbol{S}^{\star}$, so that Lagrangian-Eulerian interaction conserves energy. The Eulerian force density that is the result of applying this force-spreading operator to a Lagrangian force field turns out to be discretely divergence-free, so we refer to this new force-spreading operation as divergence-free force spreading. We name the IB method equipped with the new interpolation and spreading operators as the *Divergence-Free Immersed Boundary* (DFIB) method. As presented here, the DFIB method is limited to periodic domains.

In contrast to the local nature of interpolation and spreading in the conventional IB method, the spreading and interpolation operators of the DFIB method turn out to be non-local in that their construction requires the solution of discrete Poisson equations, although these operators can be evaluated efficiently using the Fast Fourier Transform (FFT) or multigrid methods. Another new feature of our method is that transferring information between the Eulerian grid and the Lagrangian mesh involves derivatives of the regularized delta function $\nabla\delta_h$ instead of only δ_h. We confirm through various numerical tests in both two and three spatial dimensions that the DFIB method is able to reduce volume error by several orders of magnitude compared to IBMAC and IBModified at the expense of only a modest increase in the computational cost. Moreover, we confirm that the volume error for DFIB decreases as the Lagrangian mesh is refined with the Eulerian grid size held fixed, which is not the case in the conventional IB method [26]. In addition to the substantial improvement in volume conservation, the DFIB method is quite straightforward to realize from an existing modular IB code with staggered-grid discretization, that is, by simply switching to the new velocity-interpolation and force-spreading schemes while leaving the fluid solver and time-stepping scheme unchanged.

The rest of the chapter is organized as follows. In Sec. 2.2, we begin by giving a brief description of the continuum equations of motion in the IB framework. Then we define the staggered grid on which the fluid variables live and introduce the spatial discretization of the equations of motion. Sec. 2.4 introduces the two main contributions of this chapter: divergence-free velocity interpolation and force spreading. In Sec. 2.5, we present a formally second-order time-stepping scheme that is used to evolve the spatially-discretized equations, followed by a cost comparison of DFIB and IBMAC. Numerical examples of applying DFIB to problems in two and three spatial dimensions are presented in Sec. 2.6, where the volume-conserving characteristics of the new scheme are assessed.

2.2 Equations of motion and spatial discretization

2.2.1 Equations of motion

This section provides a brief description of the continuum equations of motion in the IB framework [10]. We assume a neutrally-buoyant elastic structure Γ that is described by the Lagrangian variables $\boldsymbol{s}$, immersed in a viscous incompressible fluid occupying the whole fluid domain $\Omega \subset \mathbb{R}^3$ that is described by the Eulerian variables $\boldsymbol{x}$. Eqs. (2.2.1) and (2.2.2) are the incompressible Navier-Stokes equations describing mass and momentum conservation of the fluid, in which $\boldsymbol{u}(\boldsymbol{x}, t)$ denotes the fluid velocity, $p(\boldsymbol{x}, t)$ is the pressure, and $\boldsymbol{f}(\boldsymbol{x}, t)$ is the Eulerian force density (force per unit volume) exerted by the structure on the fluid. In this formulation, we assume that the density ρ and the viscosity μ of the fluid are constant. The fluid-structure coupled equations are:

$$\rho\left(\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u}\cdot\nabla\boldsymbol{u}\right) + \nabla p = \mu\nabla^2\boldsymbol{u} + \boldsymbol{f}, \tag{2.2.1}$$

$$\boldsymbol{\nabla}\cdot\boldsymbol{u} = 0, \tag{2.2.2}$$

$$\boldsymbol{f}(\boldsymbol{x},t) = \int_\Gamma \boldsymbol{F}(\boldsymbol{s},t)\,\delta(\boldsymbol{x} - \boldsymbol{\mathcal{X}}(\boldsymbol{s},t))\,\mathrm{d}\boldsymbol{s}, \tag{2.2.3}$$

$$\frac{\partial \boldsymbol{\mathcal{X}}}{\partial t}(\boldsymbol{s},t) = \boldsymbol{u}(\boldsymbol{\mathcal{X}}(\boldsymbol{s},t),t) = \int_\Omega \boldsymbol{u}(\boldsymbol{x},t)\,\delta(\boldsymbol{x} - \boldsymbol{\mathcal{X}}(\boldsymbol{s},t))\,\mathrm{d}\boldsymbol{x}, \tag{2.2.4}$$

$$\boldsymbol{F}(\boldsymbol{s},t) = \boldsymbol{\mathcal{F}}[\boldsymbol{\mathcal{X}}(\cdot,t)\,;\boldsymbol{s}] = -\frac{\delta E}{\delta\boldsymbol{\mathcal{X}}}(\boldsymbol{s},t). \tag{2.2.5}$$

Eqs. (2.2.3) and (2.2.4) are the fluid-structure interaction equations that couple the Eulerian and the Lagrangian variables. Eq. (2.2.3) relates the Lagrangian force density $\boldsymbol{F}(\boldsymbol{s},t)$ to the Eulerian force density $\boldsymbol{f}(\boldsymbol{x},t)$ using the Dirac delta function, where $\boldsymbol{\mathcal{X}}(\boldsymbol{s},t)$ is the physical position of the Lagrangian point $\boldsymbol{s}$. Eq. (2.2.4) is simply the no-slip boundary condition of the Lagrangian structure, i.e., the Lagrangian point $\boldsymbol{\mathcal{X}}(\boldsymbol{s},t)$ moves at the same velocity as the fluid at that point. In Eq. (2.2.5), the system is closed by expressing the Lagrangian force density $\boldsymbol{F}(\boldsymbol{s},t)$ in the form of a force density functional $\boldsymbol{\mathcal{F}}[\boldsymbol{\mathcal{X}}(\cdot,t)\,;\boldsymbol{s}]$, which in many cases can be derived from an elastic energy functional $E[\boldsymbol{\mathcal{X}}(\cdot,t)\,;\boldsymbol{s}]$ by taking the variational derivative, denoted here by $\delta/\delta\boldsymbol{\mathcal{X}}$, of the elastic energy.

2.2.2 Spatial discretization

Throughout the chapter, we assume the fluid occupies a *periodic* domain $\Omega = [0,L]^3$ that is discretized by a uniform $N\times N\times N$ Cartesian grid with meshwidth $h = \frac{L}{N}$. Each grid cell is indexed by (i,j,k) for $i,j,k = 0,\dots,N-1$. For the Eulerian fluid equations, we use the staggered-grid discretization, in which the pressure p is defined on the cell-centered grid (Fig. 2.1a), denoted by $\mathbb{C}$, i.e., at positions $\mathbf{x}_{i,j,k} = ((i+\frac{1}{2})h, (j+\frac{1}{2})h, (k+\frac{1}{2})h)$. The discrete fluid velocity $\mathbf{u}$ is defined on the face-centered grid (Fig. 2.1b), denoted by $\mathbb{F}$, with each

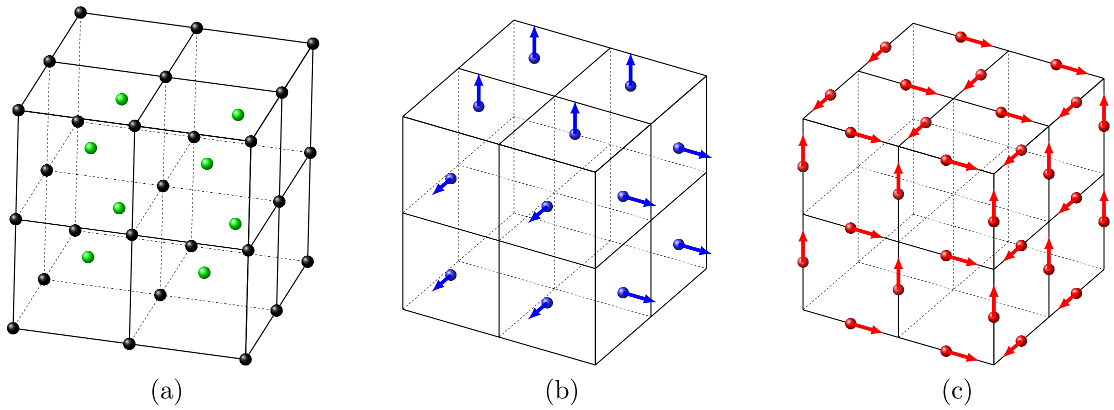


Fig. 2.1: Staggered grids on which discrete grid functions are defined. (a) Cell-centered (green) and node-centered (black) grids for scalar functions. (b) Face-centered grid for vector grid functions. (c) Edge-centered grid for vector grid functions.

component perpendicular to the corresponding cell faces, i.e., at positions $\mathbf{x}_{i-\frac{1}{2},j,k}$, $\mathbf{x}_{i,j-\frac{1}{2},k}$ and $\mathbf{x}_{i,j,k-\frac{1}{2}}$ for each velocity component respectively. We also introduce two additional shifted grids: the node-centered grid (Fig. 2.1a) for scalar grid functions, denoted by $\mathbb{N}$, i.e., at positions $\mathbf{x}_{i-\frac{1}{2},j-\frac{1}{2},k-\frac{1}{2}}$, and the edge-centered grid (Fig. 2.1c) for vector grid functions, denoted by $\mathbb{E}$, with each component defined to be parallel to the corresponding cell edges i.e., at positions $\mathbf{x}_{i,j-\frac{1}{2},k-\frac{1}{2}}$, $\mathbf{x}_{i-\frac{1}{2},j,k-\frac{1}{2}}$ and $\mathbf{x}_{i-\frac{1}{2},j-\frac{1}{2},k}$ for each component respectively. In Sec. 2.4, we will use these half-shifted staggered grids to construct divergence-free velocity interpolation and force spreading.

To discretize the differential operators in Eqs. (2.2.1) and (2.2.2), we introduce the central difference operators corresponding to the partial derivatives $\partial/\partial x_\alpha$,

$$D^h_\alpha \varphi := \frac{\varphi(\mathbf{x}+\frac{h}{2}\mathbf{e}_\alpha)-\varphi(\mathbf{x}-\frac{h}{2}\mathbf{e}_\alpha)}{h}, \quad \alpha = 1,2,3, \tag{2.2.6}$$

where φ is a scalar grid function and $\{\mathbf{e}_1,\mathbf{e}_2,\mathbf{e}_3\}$ is the standard basis of $\mathbb{R}^3$. We can use D^h_α to define the discrete gradient, divergence and curl operators:

$$\mathbf{G}^h\varphi := (D^h_1\varphi,\, D^h_2\varphi,\, D^h_3\varphi), \tag{2.2.7}$$

$$\mathbf{D}^h \cdot \mathbf{v} := D^h_\alpha v_\alpha, \tag{2.2.8}$$

$$\mathbf{D}^h \times \mathbf{v} := \epsilon_{ijk} D^h_j v_k, \tag{2.2.9}$$

where $\mathbf{v}$ is a vector grid function, ϵ_{ijk} is the totally antisymmetric tensor, and the Einstein summation convention is used here. The discrete differential operators may be defined on different pairs of domain and range (half-shifted staggered grids), and therefore, in a slight abuse of notation, we will use the same notation to denote the different operators,

$$\mathbf{G}^h : \varphi(\mathbb{C}) \longrightarrow \mathbf{v}(\mathbb{F}) \text{ or } \varphi(\mathbb{N}) \longrightarrow \mathbf{v}(\mathbb{E}), \tag{2.2.10}$$

$$\mathbf{D}^h \cdot : \mathbf{v}(\mathbb{E}) \longrightarrow \varphi(\mathbb{N}) \text{ or } \mathbf{v}(\mathbb{F}) \longrightarrow \varphi(\mathbb{C}), \tag{2.2.11}$$

$$\mathbf{D}^h \times : \mathbf{v}(\mathbb{E}) \longrightarrow \mathbf{v}(\mathbb{F}) \text{ or } \mathbf{v}(\mathbb{F}) \longrightarrow \mathbf{v}(\mathbb{E}). \tag{2.2.12}$$

Although the curl operator does not appear in the equations of motion explicitly, we define it here for use in Sec. 2.4. The discrete scalar Laplacian operator can be defined by $L^h = \mathbf{D}^h \cdot \mathbf{G}^h$, which yields the familiar compact second-order approximation to ∇^2:

$$L^h \varphi := \sum_{\alpha=1}^{3} \frac{\varphi(\mathbf{x} + h\mathbf{e}_\alpha) - 2\varphi(\mathbf{x}) + \varphi(\mathbf{x} - h\mathbf{e}_\alpha)}{h^2}. \tag{2.2.13}$$

Note that the range and domain of L^h are a set of grid functions defined on the same grid, and that grid can be $\mathbb{C}$ or $\mathbb{N}$ or any of the three subgrids of $\mathbb{E}$ or $\mathbb{F}$ on which the different components of vector-valued functions are defined. We will use the notation $\mathbf{L}^h$ to denote the discrete vector Laplacian operator that applies (the appropriately shifted) L^h to each component of a vector grid function.

2.2.2.1 Advection

We follow the same treatment of discretization of the advection term as in earlier presentations of the IB method [35]. From the incompressibility of the fluid flow $\boldsymbol{\nabla} \cdot \boldsymbol{u} = 0$, we can write the advection term in the skew-symmetric form

$$[(\boldsymbol{u} \cdot \nabla)\boldsymbol{u}]_\alpha = \frac{1}{2}\boldsymbol{u} \cdot (\nabla u_\alpha) + \frac{1}{2}\boldsymbol{\nabla} \cdot (\boldsymbol{u} u_\alpha), \quad \alpha = 1, 2, 3. \tag{2.2.14}$$

Let $\boldsymbol{N}(\mathbf{u})$ denote the discretization of Eq. (2.2.14), and we define

$$[\boldsymbol{N}(\mathbf{u})]_\alpha = \frac{1}{2}\tilde{\mathbf{u}} \cdot \mathbf{G}^{2h} u_\alpha + \frac{1}{2}\mathbf{D}^{2h} \cdot (\tilde{\mathbf{u}} u_\alpha), \quad \alpha = 1, 2, 3, \tag{2.2.15}$$

where $\tilde{\mathbf{u}}$ denotes an averaged collocated advective velocity whose components all live on the same grid as u_α. The advective velocity $\tilde{\mathbf{u}}$ in [35] is obtained by using the same interpolation scheme as the one used for moving the immersed structure. In our work, we simply take the average of $\mathbf{u}$ on the grid. For example, the three components of $\tilde{\mathbf{u}}$ in the x-component equation are

$$\begin{aligned}
\tilde{u}_1 &= u_1(\mathbf{x}_{i-\frac{1}{2},j,k})\,, \\
\tilde{u}_2 &= \frac{u_2(\mathbf{x}_{i,j-\frac{1}{2},k}) + u_2(\mathbf{x}_{i,j+\frac{1}{2},k}) + u_2(\mathbf{x}_{i-1,j-\frac{1}{2},k}) + u_2(\mathbf{x}_{i-1,j+\frac{1}{2},k})}{4}, \\
\tilde{u}_3 &= \frac{u_3(\mathbf{x}_{i,j,k-\frac{1}{2}}) + u_3(\mathbf{x}_{i,j,k+\frac{1}{2}}) + u_3(\mathbf{x}_{i-1,j,k-\frac{1}{2}}) + u_3(\mathbf{x}_{i-1,j,k+\frac{1}{2}})}{4}.
\end{aligned}$$

Note that in the y- and z-component equations, we need different averages of $\mathbf{u}$ to construct $\tilde{\mathbf{u}}$. We choose to use the wide-stencil operators in Eq. (2.2.15) so that the resulting grid functions are all defined on the same grid as u_α. A more compact discretization of the advection term has been previously described in [31, 37, 39].

2.2.2.2 Fluid-Structure Interaction

The immersed structure Γ is discretized by a Lagrangian mesh of M points or markers, denoted here by a non-italic $\mathbf{X} = \{\mathbf{X}_m\}_{m=1}^{M}$, and the discrete Lagrangian force densities defined on the Lagrangian markers are $\mathbf{F} = \{\mathbf{F}_m\}_{m=1}^{M}$. As discussed in the introduction, we can extend the notion of velocity interpolation to any point $\boldsymbol{X}$ in the domain, not just restricted to the Lagrangian markers $\mathbf{X}$, and define a *continuous* interpolated velocity field $\boldsymbol{U}(\boldsymbol{X}) = (\boldsymbol{\mathcal{J}}\mathbf{u})(\boldsymbol{X})$. In the conventional IB method, the *continuous* velocity-interpolation operator $\boldsymbol{\mathcal{J}}_{\text{IB}}$ can be defined as

$$\boldsymbol{U}(\boldsymbol{X}) = (\boldsymbol{\mathcal{J}}_{\text{IB}}\mathbf{u})(\boldsymbol{X}) := \sum_{\mathbf{x}\in\mathbb{F}} \mathbf{u}(\mathbf{x})\delta_h(\mathbf{x} - \boldsymbol{X})h^3. \tag{2.2.16}$$

We note that the interpolated velocity field given by Eq. (2.2.16) is *not* generally divergence-free with respect to the continuum divergence operator[1], i.e., generally

$$(\boldsymbol{\nabla} \cdot \boldsymbol{U})(\boldsymbol{X}) = -\sum_{\mathbf{x}\in\mathbb{F}} \mathbf{u}(\mathbf{x}) \cdot (\nabla\delta_h)(\mathbf{x} - \boldsymbol{X})h^3 \neq 0, \tag{2.2.17}$$

even if $\mathbf{u}$ is discretely divergence-free with respect to the discrete divergence operator. The restriction of $\boldsymbol{\mathcal{J}}$ to the collection of Lagrangian markers $\mathbf{X}$ defines the discrete IB interpolation operator

$$(\boldsymbol{S}^{\star}[\mathbf{X}]\mathbf{u})(\mathbf{X}) = (\boldsymbol{\mathcal{J}}\mathbf{u})(\mathbf{X}). \tag{2.2.18}$$

We will also develop a new force-spreading operator $\boldsymbol{S}[\mathbf{X}]$ that is the adjoint of the new velocity-interpolation operator $\boldsymbol{S}^{\star}[\mathbf{X}]$. Here we use the notation $[\mathbf{X}]$ to emphasize that these linear operators are parametrized by the position of the markers, as will be important when discussing temporal integration.

[1]The interpolated velocity given by Eq. (2.2.16) has the same regularity as the regularized delta function δ_h which are generally at least $\mathscr{C}^1$ in the IB method, and therefore, the divergence of $\boldsymbol{U}$ is well-defined.

The discretization of the interaction equations (Eqs. (2.2.3) and (2.2.4)) can be concisely written in the form

$$\mathbf{f}(\mathbf{x}) = (\boldsymbol{S}[\mathbf{X}]\mathbf{F})\,(\mathbf{x}), \tag{2.2.19}$$

$$\mathbf{U}(\mathbf{X}) = (\boldsymbol{S}^{\star}[\mathbf{X}]\mathbf{u})\,(\mathbf{X}), \tag{2.2.20}$$

where $\mathbf{f}$ is the discrete Eulerian force density defined on the appropriate subgrid of $\mathbb{F}$ for each component, and $\mathbf{U} = \{\mathbf{U}_m\}_{m=1}^{M}$ denotes the interpolated velocities at the Lagrangian markers $\mathbf{X}$. In the conventional IB method, the force-spreading operator $\boldsymbol{S}_{\text{IB}}$ and the velocity-interpolation operator $\boldsymbol{S}_{\text{IB}}^{\star}$ are simply discrete approximations of the surface and volume integrals in Eqs. (2.2.3) and (2.2.4), i.e.,

$$(\boldsymbol{S}_{\text{IB}}[\mathbf{X}]\mathbf{F})(\mathbf{x}) := \sum_{m=1}^{M} \mathbf{F}_m\, \delta_h(\mathbf{x} - \mathbf{X}_m)\Delta\mathbf{s}, \tag{2.2.21}$$

$$(\boldsymbol{S}_{\text{IB}}^{\star}[\mathbf{X}]\mathbf{u})(\mathbf{X}) := \sum_{\mathbf{x}\in\mathbb{F}} \mathbf{u}(\mathbf{x})\delta_h(\mathbf{x} - \mathbf{X})h^3, \tag{2.2.22}$$

and they are adjoint operators with respect to the power identity (inner product) defined later in Eq. (2.4.10). Note that Eq. (2.2.22) is a vector equation. For each of the three components of the equation, the sum $\mathbf{x} \in \mathbb{F}$ is to be understood here and in similar expressions as the sum over the appropriate subgrid of $\mathbb{F}$. In Eqs. (2.2.21) and (2.2.22), the Dirac delta function is replaced by a regularized delta function δ_h to facilitate the coupling between the Eulerian and Lagrangian grids. In the following section we introduct a new family of regularized delta functions that have three continuous and yield a substantially improved translational invariance.

2.3 A new family of regularized delta functions

In the IB method, the 3D discrete delta function is assumed to be represented by a tensor product of a single-variable kernel $\phi(r)$,

$$\delta_h(\mathbf{x}) = \frac{1}{h^3}\phi\left(\frac{x_1}{h}\right)\phi\left(\frac{x_2}{h}\right)\phi\left(\frac{x_3}{h}\right), \tag{2.3.1}$$

where x_1, x_2, x_3 are the Cartesian components of $\mathbf{x}$ and h is the meshwidth. This representation is not essential, but it significantly simplifies the discussion, since the single-variable kernel $\phi(r)$ is the object of interest. We first require that $\phi(r)$ be continuous for all real r and have compact support, i.e., $\phi(r) = 0$ for $|r| \geq r_s$, where r_s is the radius of support. Continuity of ϕ is assumed in order to avoid sudden jumps in the interpolated velocity or applied force as Lagrangian markers move through the Eulerian grid. It turns out that most IB kernels are $\mathscr{C}^1$ even though the higher regularity is not explicitly assumed. The reason for that is still a mystery, but higher regularity of the IB kernel is a nice feature to have in certain applications, such as the interpolation of derivatives or the spreading of a force dipole. Compact support of ϕ is required for computational efficiency.

The function $\phi(r)$ is constructed by requiring a subset of the following moment conditions:

$$\begin{aligned}
&\text{(i)} \quad \text{Zeroth moment:} && \sum_j \phi(r-j) = 1, \\
&\text{(ii)} \quad \text{Even-odd:} && \sum_{j \text{ even}} \phi(r-j) = \sum_{j \text{ odd}} \phi(r-j) = \frac{1}{2}, \\
&\text{(iii)} \quad \text{First moment:} && \sum_j (r-j)\,\phi(r-j) = 0, \\
&\text{(iv)} \quad \text{Second moment:} && \sum_j (r-j)^2 \phi(r-j) = K, \ \text{for some constant } K, \\
&\text{(v)} \quad \text{Third moment:} && \sum_j (r-j)^3 \phi(r-j) = 0.
\end{aligned}$$

The motivation of imposing moment conditions is well discussed in [40, 41]. Briefly, the zeroth moment condition implies that the total force is the same in the Eulerian and Lagrangian grids when δ_h is used for force spreading. The even-odd condition implies (i), and was originally proposed to avoid the "checkerboard" instability that may arise from using a collocated-grid fluid solver. Liu and Mori [41] generalized this condition to the so called "smoothing order" condition and showed that it has the effect of suppressing high-frequency errors and preventing Gibbs-type phenomena. Conservation of total torque relies on the first moment condition. Moreover, (i) and (iii) guarantee that a smooth function is interpolated with second-order accuracy when δ_h is used for interpolation. The second moment condition with $K = 0$ and the third moment condition are needed to derive kernels with a higher order of interpolation accuracy.

In addition to moment conditions, $\phi(r)$ is required to satisfy the sum-of-squares condition,

$$\sum_j (\phi(r-j))^2 = C, \ \text{for some constant } C. \qquad (2.3.2)$$

The sum-of-squares condition Eq. (2.3.2) is a weaker version of exact grid translational invariance,

$$\tilde{\phi}(r_1, r_2) = \sum_j \phi(r_1 - j)\,\phi(r_2 - j) = \Phi(r_1 - r_2), \text{ for all } r_1, r_2. \tag{2.3.3}$$

In other words, the coupling of $\phi(r)$ between any arbitrary two points r_1, r_2 is a function of $r_1 - r_2$ only. However, it can be shown that Eq. (2.3.3) is inconsistent with the assumption of ϕ being compactly supported [40]. The sum-of-squares condition does give some information about the coupling of ϕ, since it can be deduced from the Cauchy-Schwarz inequality that

$$\left|\tilde{\phi}(r_1, r_2)\right| = \left|\sum_j \phi(r_1 - j)\,\phi(r_2 - j)\right| \leq C, \text{ for all } r_1, r_2. \tag{2.3.4}$$

Eq. (2.3.4) guarantees that the coupling between two Lagrangian markers is strongest when the markers coincide, and furthermore Eq. (2.3.2) implies that the self-coupling is independent of the marker position.

IB Kernel	Support r_s	Even-Odd	Zeroth Moment	First Moment	Second Moment	Third Moment	Sum of Squares	Regularity
Standard 3-point	1.5	✗	✓	✓	✗	✗	$\frac{1}{2}$	$\mathscr{C}^1$
Smoothed 3-point	2	✗	✓	✓	✗	✗	✗	$\mathscr{C}^2$
Standard 4-point	2	✓	✓	✓	✗	✗	$\frac{3}{8}$	$\mathscr{C}^1$
Smoothed 4-point	2.5	✗	✓	✓	✗	✗	✗	$\mathscr{C}^2$
Standard 6-point	3	✓	✓	✓	0	✓	$\frac{67}{128}$	$\mathscr{C}^1$
New 5-point	2.5	✗	✓	✓	$\frac{38}{60} - \frac{\sqrt{69}}{60}$	✓	$\approx .393$	$\mathscr{C}^3$
New 6-point	3	✓	✓	✓	$\frac{59}{60} - \frac{\sqrt{29}}{20}$	✓	$\approx .326$	$\mathscr{C}^3$

Table 2.1: Common immersed-boundary kernels with their properties and moment conditions they satisfy. ✓: the kernel satisfies the moment condition; ✗: the kernel does not satisfy the moment condition. In the second moment column, the value of the second moment constant K is given when the second moment condition is satisfied. The regularity column shows the number of continuous derivatives each IB kernel has.

In Table 2.1, we list some common IB kernels and the conditions they satisfy. The most widely used IB kernel is the standard 4-point kernel [40]. The standard 3-point kernel satisfies the zeroth moment condition but not the even-odd condition. It was first introduced in an adaptive IB method using the staggered-grid discretization [36]. The standard 6-point kernel (with $K = 0$) satisfies all the moment conditions listed above [42]. It can be shown that the standard 6-point kernel interpolates cubic functions exactly and smooth functions with fourth-order accuracy. However, as shown in Fig. 2.5, it has a larger deviation from translational invariance for a pair of markers with distance $d \approx 2.5$ even compared to the standard 4-point kernel. In terms of its defining postulates, our new 6-point kernel differs from the standard 6-point kernel only in the nonzero second-moment constant K (the sum-of-squares constant C is determined once K is fixed). The new 5-point kernel assumes the same postulates as the new 6-point kernel except for the "even-odd" condition.

The special choice of K given in Eq. (2.3.16) and Eq. (2.3.19) lead to new 5-point and 6-point kernels that are $\mathscr{C}^3$ and significantly improve translational invariance compared with other IB kernels. The construction of an IB kernel with a positive and constant second moment K was originally motivated by the important physical implication of the second moment in particle suspensions, namely it is associated with the quadrupole correction in the Faxén relation for a rigid sphere in an arbitrary Stokes flow [43]. The result that the new kernels have three continuous derivatives is unexpected, however, this makes the new kernel more generally useful. By applying a smoothing technique to the standard IB kernels, Yang, *et al.* [44] developed a family of $\mathscr{C}^2$ IB kernels whose first derivative satisfies up to the second moment condition for the derivative. They showed that these derivative moment conditions are intrinsically linked to the error of force spreading in the IB scheme, and IB kernels that satisfy these conditions can significantly reduce non-physical spurious oscillations of force spreading in moving-boundary problems. By differentiating the moment conditions satisfied by $\phi(r)$, we can verify that the derivative of our new $\mathscr{C}^3$ kernels satisfy up to third moment conditions as advocated by Yang *et al.* [44]. We will also include the smoothed 3-point and 4-point kernels in the comparison of translational invariance in Sec. 2.3.2. Liu and Mori developed a MATLAB routine that automatically generates all the standard IB kernels as well as many others [41]. We have also made our MATLAB codes for generating the new kernels available at `https://github.com/stochasticHydroTools/IBMethod`.

2.3.1 Two new kernels

2.3.1.1 A new 5-point kernel

Our new 5-point kernel satisfies the sum-of-squares condition Eq. (2.3.2) and the moment conditions (i), (iii)-(v), but it does not satisfy the even-odd condition (ii). The support is defined to be five grid points, i.e., $r_s = \frac{5}{2}$. We follow a similar derivation of the standard 4-point kernel [40]. By first restricting $r \in \left[-\frac{1}{2}, \frac{1}{2}\right]$, we have 5 unknowns:

$$\{\phi(r-2),\, \phi(r-1),\, \phi(r),\, \phi(r+1),\, \phi(r+2)\}\,. \tag{2.3.5}$$

Note that the moment conditions (i), (iii)-(v) are four linear equations in these five unknowns, and we can express all the other four unknowns in terms of $\phi(r)$,

$$\begin{aligned}
\phi(r-2) &= \frac{1}{12}\left(2\phi(r)+3Kr+2K+r^3+2r^2-r-2\right), && (2.3.6)\\
\phi(r-1) &= \frac{1}{6}\left(-4\phi(r)-3Kr-K-r^3-r^2+4r+4\right), && (2.3.7)\\
\phi(r+1) &= \frac{1}{6}\left(-4\phi(r)+3Kr-K+r^3-r^2-4r+4\right), && (2.3.8)\\
\phi(r+2) &= \frac{1}{12}\left(2\phi(r)-3Kr+2K-r^3+2r^2+r-2\right). && (2.3.9)
\end{aligned}$$

For the special value $r = 1/2$ so that $\phi\left(\frac{5}{2}\right) = 0$, we get

$$\left\{\phi\left(-\frac{3}{2}\right), \phi\left(-\frac{1}{2}\right), \phi\left(\frac{1}{2}\right), \phi\left(\frac{3}{2}\right)\right\} = \left\{\frac{1}{16}(4K-1), \frac{1}{16}(9-4K), \frac{1}{16}(9-4K), \frac{1}{16}(4K-1)\right\}. \tag{2.3.10}$$

Substituting these values into the sum of squares condition Eq. (2.3.2), we obtain an expression for $C(K)$,

$$C(K) = \frac{1}{128}(9-4K)^2 + \frac{1}{128}(4K-1)^2, \tag{2.3.11}$$

where the value of K remains to be determined. Next we solve the quadratic equation of $\phi(r)$ from the sum-of-squares condition Eq. (2.3.2) by using Eqs. (2.3.6) to (2.3.9) and Eq. (2.3.11),

$$\phi(r) = \frac{1}{280}\left(-40K-40r^2+136+\sqrt{2\beta(r)+2\gamma(r)}\right), \tag{2.3.12}$$

where

$$\begin{aligned}
\beta(r) &= -12600K^2r^2+3600K^2-8400Kr^4+25680Kr^2-6840K+3123, && (2.3.13)\\
\gamma(r) &= -40r^2\left(35r^4-202r^2+311\right). && (2.3.14)
\end{aligned}$$

We note that the positive square root is chosen in Eq. (2.3.12) because of the continuity assumption of ϕ, i.e., by setting $r = \frac{1}{2}$, we select the branch that gives $\phi\left(\frac{1}{2}\right) = \frac{1}{16}(9 - 4K)$ as in Eq. (2.3.10).

We have the freedom to choose K to construct ϕ with higher regularity. A symbolic calculation in *Mathematica* matching derivatives of $\phi(r)$ at $r = \frac{1}{2}$ reveals that for any $K \in \left[0, \frac{21}{20}\right)$, $\phi \in \mathscr{C}^1$. For the second derivative to be continuous at $r = \frac{1}{2}$, K must satisfy

$$720K^2 - 912K + 275 = 0. \tag{2.3.15}$$

We can verify (by plotting) that exactly one of the two roots of Eq. (2.3.15) guarantees that $\beta(r) + \gamma(r) \geq 0$, so that $\phi(r)$ is real-valued for $r \in \left[\frac{1}{2}, \frac{1}{2}\right]$, and this special value of K is

$$K = \frac{1}{60}\left(38 - \sqrt{69}\right). \tag{2.3.16}$$

If we proceed further with matching the third derivative to be continuous at $r = \frac{1}{2}$, then K must satisfy

$$60480K^3 - 116208K^2 + 73260K - 15125 = 0, \tag{2.3.17}$$

whose roots are

$$K = \left\{\frac{55}{84}, \frac{1}{60}\left(38 + \sqrt{69}\right), \frac{1}{60}\left(38 - \sqrt{69}\right)\right\}. \tag{2.3.18}$$

We observe that our choice of K in Eq. (2.3.16) is among the roots of the third derivative matching condition, and therefore, it also makes the third derivative of ϕ continuous at $r = \frac{1}{2}$.

Remarkably, the derivative matching conditions at $r = \frac{1}{2}$ are sufficient to ensure that $\phi \in \mathscr{C}^3$ everywhere. Existence and continuity of derivatives was never assumed a priori, but was only a consequence of an appropriate choice of K. Moreover, note that for $K \in \left[0, \frac{1}{60}\left(38 - \sqrt{69}\right)\right)$, we have $\phi''\left(\frac{5}{2}\right) < 0$. Since $\phi\left(\frac{5}{2}\right) = \phi'\left(\frac{5}{2}\right) = 0$, this implies that $\phi(r)$ takes negative values in a neighborhood of $r = \frac{5}{2}$. We emphasize that the special choice of K given by Eq. (2.3.16) is the smallest positive K for which ϕ is non-negative, and it is also the only value of K that gives three continuous derivatives of ϕ.

2.3.1.2 A new 6-point kernel

Our new 6-point kernel satisfies the sum-of-squares condition Eq. (2.3.2) and the moment conditions (ii)-(v) (and therefore (i)) with the second-moment constant

$$K = \frac{59}{60} - \frac{\sqrt{29}}{20}. \tag{2.3.19}$$

The derivation of this kernel follows a nearly identical procedure to that of the new 5-point kernel. First, the sum-of-squares constant C can be expressed in terms of K by considering the special case $r = 0$. Next, by restricting r to the interval $[0, 1]$, we have 6 unknowns: $\phi(r-3), \phi(r-2)$, $\phi(r-1), \phi(r), \phi(r+1), \phi(r+2)$ and 6 equations (the even-odd condition accounts for two equations). By expressing all the other unknowns in terms of $\phi(r-3)$ using (ii)-(v), we can solve for $\phi(r-3)$ from the quadratic equation determined by Eq. (2.3.2). The continuity assumption of ϕ is now used to select the appropriate root to piece together a continuous function, i.e., by setting $r = 0$, we select the root that gives $\phi(-3) = 0$. As mentioned earlier, ϕ being $\mathscr{C}^1$ follows implicitly from our defining postulates, i.e., $\phi'(-3) = 0$. We have the freedom to choose K so that $\phi''(-3) = 0$, which uniquely determines the special value Eq. (2.3.19). The formula for the new 6-point kernel is given by

$$\begin{aligned} \beta(r) &= \frac{9}{4} - \frac{3}{2}(K + r^2)r + (\frac{22}{3} - 7K)r - \frac{7}{3}r^3\,, \qquad (2.3.20) \\ \gamma(r) &= -\frac{11}{32}r^2 + \frac{3}{32}(2K + r^2)r^2 + \end{aligned}$$

$$\frac{1}{72}\left((3K-1)r+r^3\right)^2+\frac{1}{18}\left((4-3K)r-r^3\right)^2\,, \tag{2.3.21}$$

$$\begin{aligned}
\phi(r-3) &= \frac{-\beta(r)+\operatorname{sgn}\left(\frac{3}{2}-K\right)\sqrt{\beta^2(r)-112\gamma(r)}}{56}\,, && (2.3.22)\\
\phi(r-2) &= -3\phi(r-3)-\frac{1}{16}+\frac{K+r^2}{8}+\frac{(3K-1)r}{12}+\frac{r^3}{12}\,, && (2.3.23)\\
\phi(r-1) &= 2\phi(r-3)+\frac{1}{4}+\frac{(4-3K)r}{6}-\frac{r^3}{6}\,, && (2.3.24)\\
\phi(r) &= 2\phi(r-3)+\frac{5}{8}-\frac{K+r^2}{4}\,, && (2.3.25)\\
\phi(r+1) &= -3\phi(r-3)+\frac{1}{4}-\frac{(4-3K)r}{6}+\frac{r^3}{6}\,, && (2.3.26)\\
\phi(r+2) &= \phi(r-3)-\frac{1}{16}+\frac{K+r^2}{8}-\frac{(3K-1)r}{12}-\frac{r^3}{12}\,. && (2.3.27)
\end{aligned}$$

Note that, in the formula presented above, $r \in [0,1]$. The new 5-point and 6-point kernels are Gaussian-like function, as shown in Fig. 2.2 and Fig. 2.3, and they both have three continuous derivatives. As a comparison, the standard 3-point, 4-point, 6-point kernels and their continuous first derivative are plotted in Fig. 2.4. We notice that the new 5-point and 6-point kernels are non-negative for all r, whereas the standard 6-point kernel has negative tails.

2.3.2 Tests for translational invariance

In this section, we demonstrate a significant improvement in the translational invariance[2] of our new 6-point kernel. We randomly select 10^5 pairs of Lagrangian markers $\mathbf{X}_1, \mathbf{X}_2$ in a periodic box $[0,32]^3$ with meshwidth $h=1$ and compute the 3D generalization of Eq. (2.3.3),

$$\tilde{\delta}(\mathbf{X}_1,\mathbf{X}_2)=\sum_{\mathbf{x}\in g_h}\delta_h(\mathbf{x}-\mathbf{X}_1)\,\delta_h(\mathbf{x}-\mathbf{X}_2), \tag{2.3.28}$$

[2]The test that we use actually checks for rotational invariance at the same time, since it involves the Euclidean distance between a pair of markers, and not merely the vector from one marker to the other.

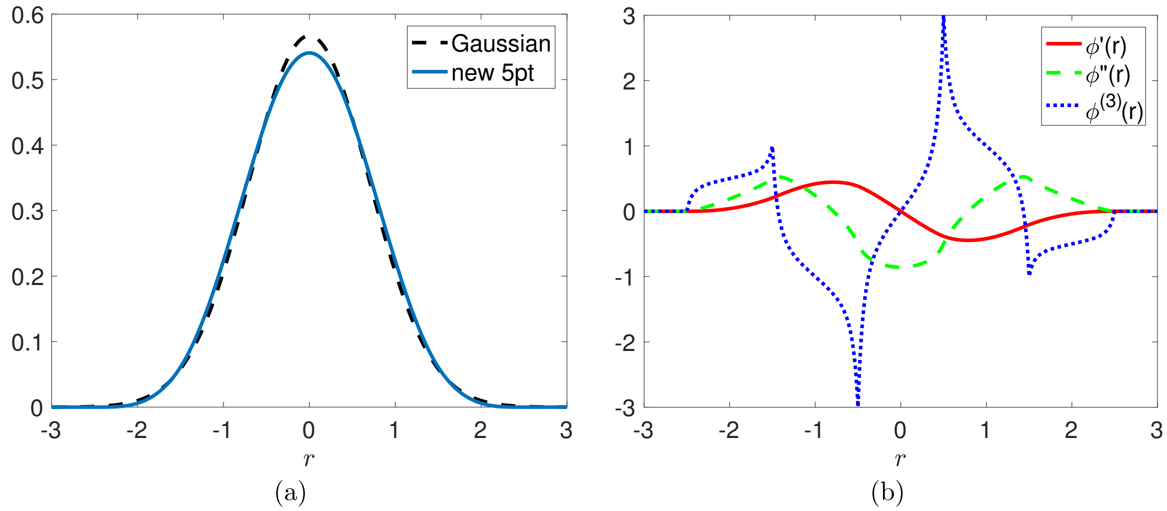


Fig. 2.2: (a) The new 5-point kernel compared with the Gaussian with the second moment given by Eq. (2.3.16). (b) The first three derivatives of the new 5-point kernel.

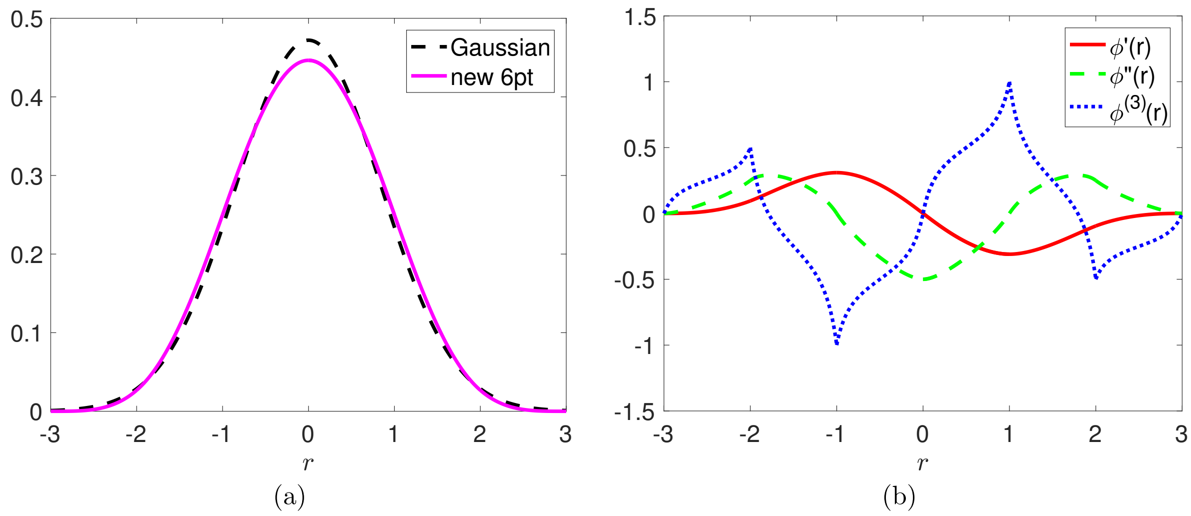
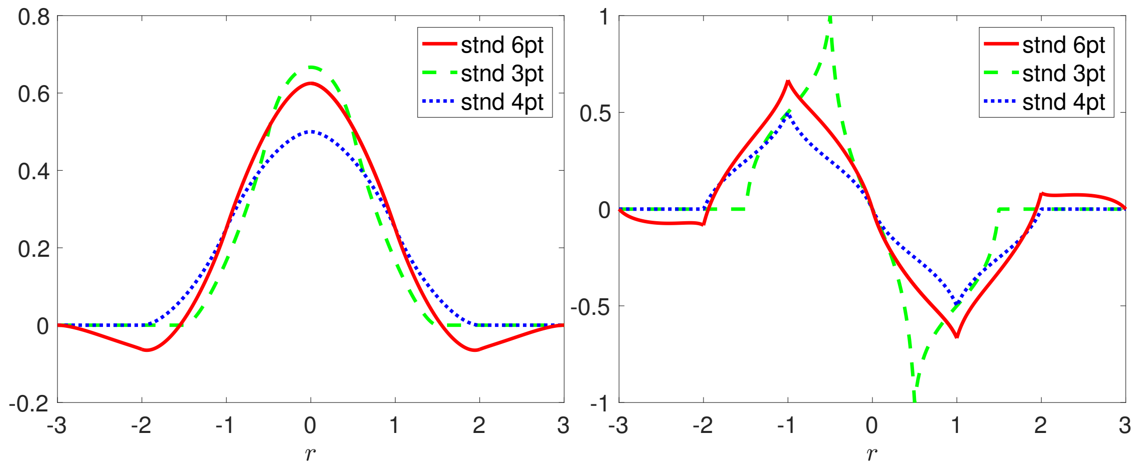


Fig. 2.3: (a) The new 6-point kernel compared with the Gaussian with the second moment given by Eq. (2.3.19). (b) The first three derivatives of the new 6-point kernel.



(a) (b)

Fig. 2.4: (a) The standard 3-point, 4-point, and 6-point kernels. (b) The first derivatives of the standard 3-point, 4-point, and 6-point kernels.

where $\mathbf{x}$ denotes a grid point on the Eulerian grid g_h. In Fig. 2.5, we plot $\tilde{\delta}(\mathbf{X}_1, \mathbf{X}_2)$ normalized by the constant C^3 from Eq. (2.3.2), versus the distance $d = |\mathbf{X}_1 - \mathbf{X}_2|$. The data are binned according to $d = |\mathbf{X}_1 - \mathbf{X}_2|$, and error-bars showing the maximum, mean and minimum of each bin are overlaid with the data. If an IB kernel were exactly translation-invariant, the plot of $\tilde{\delta}(\mathbf{X}_1, \mathbf{X}_2)$ would be a curve. The spreading pattern in the data around this curve clearly indicates that none of the IB kernels compared here are exactly translation-invariant, but gives a qualitative picture of how close to translational invariance each kernel is. The data of the new IB kernels and the smoothed 4-point kernel almost form a curve, while the data of the other kernels have larger deviations from the mean. Moreover, the deviation in the data of the new IB kernels is uniform for all distances, but the standard 6-point kernel has a much larger deviation for $d \approx 2.5$. For a more quantitative comparison, we summarize the maximum standard deviation of all bins for each IB kernel in Table 2.2. The maximum standard deviation of the new IB kernels is an order of magnitude smaller than that of the standard IB kernels, and is about half of the deviation of the smoothed 4-point kernel.

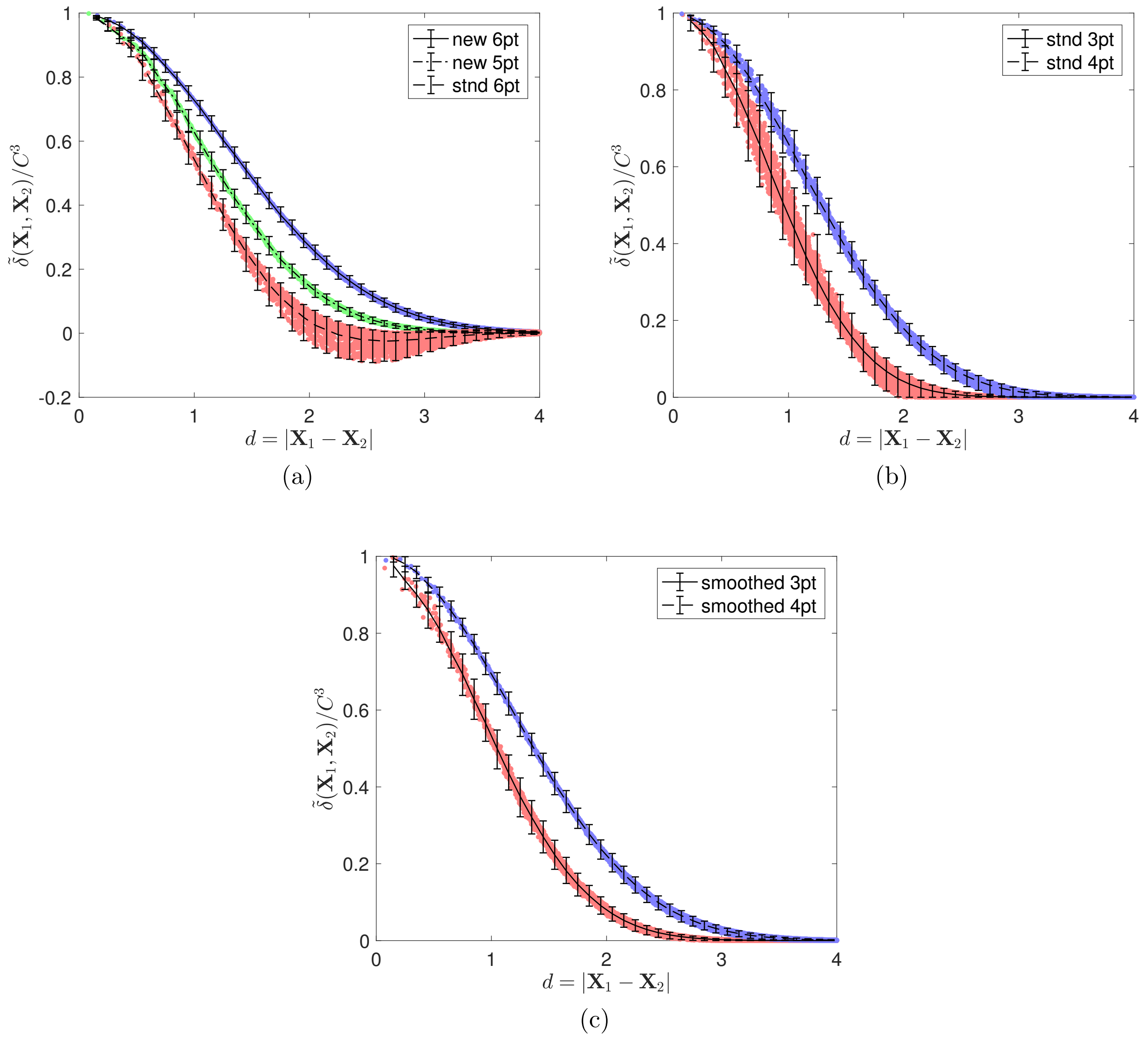


Fig. 2.5: Normalized $\tilde{\delta}(\mathbf{X}_1, \mathbf{X}_2)$ is plotted versus $d = |\mathbf{X}_1 - \mathbf{X}_2|$ for 10^5 pairs of randomly selected Lagrangian markers. The data are binned according to $d = |\mathbf{X}_1 - \mathbf{X}_2|$ and error-bars showing the maximum, mean and minimum of each bin are overlaid with the data. The deviation in the data gives a quantitative measure of translational invariance of an IB kernel. The standard deviation in the data for the new 6-point kernel (blue) is an order of magnitude smaller than for the standard IB kernels. (a) The standard 6-point kernel vs. the new 5-point kernel vs. the new 6-point kernel. (b) The standard 3-point kernel vs. the standard 4-point kernel. (c) The smoothed 3-point kernel vs. the smoothed 4-point kernel.

In terms of computational cost, we summarize the computation time of $\tilde{\delta}(\mathbf{X}_1, \mathbf{X}_2)$ for 10^5 pairs of Lagrangian markers using the kernels we have compared. The timings are based on simulations performed on a desktop with Intel Core i7-4770 CPU 3.40GHz under the MATLAB environment. The main cost of using an IB kernel in spreading/interpolation depends on its support size. In Table 2.2, the new 5-point kernel is about two times more expensive, and the new 6-point kernel is about three-to-four times more expensive than a 4-point kernel in our comparison, because the new 5-point and 6-point kernels communicate with 125 and 216 nearby grid points respectively in spreading/interpolation in 3D, while a 4-point kernel only communicates with 64 nearby grid points. The smoothed 3-point and 4-point kernels are more expensive than their standard counterparts in that they have wider supports as shown in Table 2.1. In all respects, the new IB kernels achieve significant improvement in grid translational invariance with a modest increase in computational cost.

	Standard 3-point	Smoothed 3-point	Standard 4-point	Smoothed 4-point	Standard 6-point	New 5-point	New 6-point
maximum std. dev.	0.0428	0.0212	0.0168	0.0083	0.0296	0.0051	0.0042
computation time	6.34s	9.52s	9.01s	12.06s	31.19s	17.54s	30.86s

Table 2.2: Maximum standard deviation of $\tilde{\delta}(\mathbf{X}_1, \mathbf{X}_2)$ over all bins for various IB kernels, and the computation time for computing $\tilde{\delta}(\mathbf{X}_1, \mathbf{X}_2)$ for 10^5 markers.

2.4 Divergence-free velocity interpolation and force spreading

This section presents the two main contributions of this chapter: divergence-free velocity interpolation and force spreading. Familiarity with discrete differential operators on staggered grids and with some discrete vector identities, reviewed and summarized in 2.8, will facilitate the reading of this section.

2.4.1 Divergence-free velocity interpolation

Here we introduce a new recipe for constructing an interpolated velocity field $\boldsymbol{U}(\boldsymbol{X}) = (\boldsymbol{\mathcal{J}}\mathbf{u})(\boldsymbol{X})$ that is *continuously* divergence-free with respect to the continuum divergence operator, i.e., $(\nabla \cdot \boldsymbol{U})(\boldsymbol{X}) = 0$ for all $\boldsymbol{X}$. For now we drop the dependence on time and emphasize again that $\boldsymbol{X}$ is an arbitrary position in the domain $\Omega \subset \mathbb{R}^3$, not just on the Lagrangian structure Γ. The main idea is first to construct a discrete vector potential $\mathbf{a}(\mathbf{x})$ that is defined on the edge-centered staggered grid $\mathbb{E}$, and then to apply the conventional IB interpolation to $\mathbf{a}(\mathbf{x})$ to obtain a continuum vector potential $\boldsymbol{A}(\boldsymbol{X})$, so that the Lagrangian velocity defined by $\boldsymbol{U}(\boldsymbol{X}) = (\nabla \times \boldsymbol{A})(\boldsymbol{X})$ is automatically divergence-free.

Suppose the discrete velocity field $\mathbf{u}(\mathbf{x})$ is defined on $\mathbb{F}$ and is discretely divergence-free, i.e., $\mathbf{D}^h \cdot \mathbf{u} = 0$. Let $\mathbf{u}_0$ be the mean of $\mathbf{u}(\mathbf{x})$,

$$\mathbf{u}_0 = \frac{1}{V} \sum_{\mathbf{x} \in \mathbb{F}} \mathbf{u}(\mathbf{x}) h^3, \tag{2.4.1}$$

where $V = \sum_{\mathbf{x} \in \mathbb{F}} h^3$ is the volume of the domain. Using the Helmholtz decomposition, we construct a discrete velocity potential $\mathbf{a}(\mathbf{x})$ for $\mathbf{x} \in \mathbb{E}$ that satisfies

$$\begin{cases} \mathbf{D}^h \times \mathbf{a} & = & \mathbf{u} - \mathbf{u}_0, \\ \mathbf{D}^h \cdot \mathbf{a} & = & 0, \end{cases} \tag{2.4.2}$$

where the requirement that $\mathbf{a}(\mathbf{x})$ is discretely divergence-free is an arbitrary gauge condition that makes $\mathbf{a}(\mathbf{x})$ uniquely defined up to a constant. If the gauge condition of $\mathbf{a}(\mathbf{x})$ is omitted in Eq. (2.4.2), then the discrete velocity potential $\mathbf{a}(\mathbf{x})$ is only uniquely defined up to $\mathbf{G}^h \psi$, where ψ is some unknown scalar grid function defined on $\mathbb{N}$. Note that $\mathbf{D}^h \cdot \mathbf{a}$ is a scalar field defined on $\mathbb{N}$. In 2.9, we prove that the discrete vector potential $\mathbf{a}(\mathbf{x})$ defined by Eq. (2.4.2) exists (see Theorem 3). To determine $\mathbf{a}(\mathbf{x})$ explicitly, we take the discrete curl of the first equation in Eq. (2.4.2) and use the identity Eq. (2.8.3) with the gauge condition of $\mathbf{a}(\mathbf{x})$, which leads to a vector Poisson equation for $\mathbf{a}(\mathbf{x})$,

$$-\mathbf{L}^h\,\mathbf{a} = \mathbf{D}^h \times \mathbf{u}, \tag{2.4.3}$$

that can be efficiently solved. Note that the solution of the Poisson problem Eq. (2.4.3) determines $\mathbf{a}(\mathbf{x})$ up to an arbitrary constant (it is not necessary to uniquely determine $\mathbf{a}(\mathbf{x})$ because the constant term vanishes upon subsequent differentiation).

The next step is to interpolate the discrete vector potential $\mathbf{a}(\mathbf{x})$ to obtain the continuum vector potential

$$\boldsymbol{A}(\boldsymbol{X}) = \sum_{\mathbf{x}\in\mathbb{E}} \mathbf{a}(\mathbf{x})\,\delta_h(\mathbf{x}-\boldsymbol{X})h^3. \tag{2.4.4}$$

Lastly, we take the continuum curl of $\boldsymbol{A}(\boldsymbol{X})$ with respect to $\boldsymbol{X}$,

$$(\nabla\times\boldsymbol{A})(\boldsymbol{X}) = \sum_{\mathbf{x}\in\mathbb{E}} \mathbf{a}(\mathbf{x})\times(\nabla\delta_h)(\mathbf{x}-\boldsymbol{X})h^3, \tag{2.4.5}$$

and our new interpolation is completed by adding the mean flow $\mathbf{u}_0$, that is,

$$\boldsymbol{U}(\boldsymbol{X}) = (\boldsymbol{\mathcal{J}}\mathbf{u})(\boldsymbol{X}) = \mathbf{u}_0 + \sum_{\mathbf{x}\in\mathbb{E}} \mathbf{a}(\mathbf{x})\times(\nabla\delta_h)(\mathbf{x}-\boldsymbol{X})h^3. \tag{2.4.6}$$

We note that the interpolation Eq. (2.4.4) is not performed in the actual implementation of the scheme. Instead, $\nabla\delta_h$ is computed on the edge-centered staggered grid $\mathbb{E}$ in Eq. (2.4.6). Notice that, by construction, the interpolated velocity in Eq. (2.4.6) is continuously divergence-free.

There are two important features of our new interpolation scheme that are worth mentioning. First, in comparison to locally interpolating the velocity from the nearby fluid grid in the conventional IB method, our new interpolation scheme is non-local, in that it involves solving the discrete Poisson problem Eq. (2.4.3). Second, if the regularized delta function δ_h is $\mathscr{C}^k$, we note that the interpolated velocity field given by Eq. (2.4.6) is a globally-defined function that is $\mathscr{C}^{k-1}$. We can think of the regularized delta function concentrated at $\boldsymbol{X}$ as being defined everywhere with zero outside a cube of fixed edge length (e.g. $6h$ for the $\mathscr{C}^3$ 6-point kernel [7]). Moreover, the continuity of derivatives of δ_h also applies globally, including at the edges for the cube. Since the continuum vector potential defined by Eq. (2.4.4) is a finite sum of such $\mathscr{C}^k$ functions, and the interpolated velocity field $\boldsymbol{U}(\boldsymbol{X})$ is obtained by differentiating $\boldsymbol{A}(\boldsymbol{X})$ once, then the resulting interpolated velocity field must have $k-1$ continuous derivatives. Note that if we use an IB kernel that is $\mathscr{C}^1$, then the interpolated velocity $\boldsymbol{U}$ is $\mathscr{C}^0$, and $\boldsymbol{\nabla}\cdot\boldsymbol{U}$ is defined in only a piecewise manner. This naturally brings into question whether the volume of a closed surface is strictly conserved as the surface passes over the discontinuity of the velocity derivatives. Indeed, we observe numerically that the DFIB method offers only marginal improvement in volume conservation for $\mathscr{C}^1$ kernel functions, such as the standard 4-point kernel [10], unless the Lagrangian mesh is discretized with impractically high resolution (8 markers per fluid meshwidth, see Fig. 2.9). By contrast, we will show that with only a moderate Lagrangian mesh size (1 to 2 markers per fluid meshwidth), the DFIB method offers a substantial improvement in volume conservation for kernels of higher smoothness, which gives a continuously differentiable interpolated velocity $\boldsymbol{U}$. Further, we observe that volume conservation of the DFIB method improves with the smoothness of the interpolated velocity field.

In addition to the standard 4-point kernel (denoted by ϕ_{4h}), the IB kernels considered in this chapter include the $\mathscr{C}^3$ 5-point and 6-point kernels [7] (denoted by ϕ_{5h}^{new} and ϕ_{6h}^{new}

respectively), and the $\mathscr{C}^2$ 4-point B-spline kernel [45],

$$\phi^B_{4h}(r) = \begin{cases} \frac{2}{3} - r^2 + \frac{1}{2}r^3 & 0 \le |r| < 1, \\ \frac{4}{3} - 2r + r^2 - \frac{1}{6}r^3 & 1 \le |r| < 2, \\ 0 & |r| \ge 2, \end{cases} \tag{2.4.7}$$

and the $\mathscr{C}^4$ 6-point B-spline kernel,

$$\phi^B_{6h}(r) = \begin{cases} \frac{11}{20} - \frac{1}{2}r^2 + \frac{1}{4}r^4 - \frac{1}{12}r^5 & 0 \le |r| < 1, \\ \frac{17}{40} + \frac{5}{8}r - \frac{7}{4}r^2 + \frac{5}{4}r^3 - \frac{3}{8}r^4 + \frac{1}{24}r^5 & 1 \le |r| < 2, \\ \frac{81}{40} - \frac{27}{8}r + \frac{9}{4}r^2 - \frac{3}{4}r^3 + \frac{1}{8}r^4 - \frac{1}{120}r^5 & 2 \le |r| < 3, \\ 0 & |r| \ge 3. \end{cases} \tag{2.4.8}$$

These B-spline kernels are members of a sequence of functions obtained by recursively convolving each successive kernel function against a rectangular pulse (also known as the window function), starting from the window function itself [45]. The limiting function in this sequence is a Gaussian [46], which is exactly translation-invarant and isotropic. The family of IB kernels with nonzero even moment conditions, such as ϕ_{4h} and ϕ^{new}_{6h}, also have a Gaussian-like shape, but it is not currently known whether this sequence of functions also converges to a Gaussian.

2.4.2 The force-spreading operator

The force-spreading operator $\boldsymbol{S}$ is constructed to be adjoint to the velocity-interpolation opeartor $\boldsymbol{S}^\star$ so that energy is conserved by the Lagrangian-Eulerian interaction,

$$(\mathbf{u}, \boldsymbol{S}\mathbf{F})_{\mathbf{x}} = (\boldsymbol{S}^\star \mathbf{u}, \mathbf{F})_{\mathbf{X}}, \tag{2.4.9}$$

where $(\cdot,\cdot)_{\mathbf{x}}$ and $(\cdot,\cdot)_{\mathbf{X}}$ denote the corresponding discrete inner products on the Eulerian and Lagrangian grids. In other words, the power generated by the elastic body forces is transferred to the fluid without loss,[3]

$$\sum_{\mathbf{x}\in\mathbb{F}} \mathbf{u}(\mathbf{x}) \cdot \mathbf{f}(\mathbf{x})\, h^3 = \sum_{m=1}^{M} \mathbf{U}_m \cdot \mathbf{F}_m \Delta\mathbf{s}, \tag{2.4.10}$$

where $\mathbf{U}_m$ is the Lagrangian marker velocity at $\mathbf{X}_m$, and $\mathbf{F}_m\Delta\mathbf{s}$ is the Lagrangian force applied to the fluid by the Lagrangian marker $\mathbf{X}_m$. Our goal is to find an Eulerian force density $\mathbf{f}(\mathbf{x})$ that satisfies the power identity Eq. (2.4.10). To see what Eq. (2.4.10) implies about $\mathbf{f}(\mathbf{x})$, we rewrite both sides in terms of $\mathbf{a}(\mathbf{x})$. On the left-hand side of Eq. (2.4.10), we use Eq. (2.4.2) to obtain

$$\begin{aligned}\sum_{\mathbf{x}\in\mathbb{F}} \mathbf{u}(\mathbf{x}) \cdot \mathbf{f}(\mathbf{x})\, h^3 &= \mathbf{u}_0 \cdot \sum_{\mathbf{x}\in\mathbb{F}} \mathbf{f}(\mathbf{x}) h^3 + \sum_{\mathbf{x}\in\mathbb{F}} (\mathbf{D}^h \times \mathbf{a})(\mathbf{x}) \cdot \mathbf{f}(\mathbf{x})\, h^3 \\ &= \mathbf{u}_0 \cdot \mathbf{f}_0 V + \sum_{\mathbf{x}\in\mathbb{E}} \mathbf{a}(\mathbf{x}) \cdot (\mathbf{D}^h \times \mathbf{f})(\mathbf{x})\, h^3, \end{aligned} \tag{2.4.11}$$

where the average of $\mathbf{f}(\mathbf{x})$ over the domain is

$$\mathbf{f}_0 = \frac{1}{V} \sum_{\mathbf{x}\in\mathbb{F}} \mathbf{f}(\mathbf{x})\, h^3. \tag{2.4.12}$$

Note that we have used the summation-by-parts identity Eq. (2.8.5) to transfer the discrete curl operator $\mathbf{D}^h\times$ from $\mathbf{a}(\mathbf{x})$ to $\mathbf{f}(\mathbf{x})$, and thus, the grid on which the summation is performed in Eq. (2.4.11) is $\mathbb{E}$ not $\mathbb{F}$. On the the right-hand side of Eq. (2.4.10), we substitute for $\mathbf{U}_m$ by using the divergence-free velocity interpolation Eq. (2.4.6),

$$\sum_{m=1}^{M} \mathbf{U}_m \cdot \mathbf{F}_m \Delta\mathbf{s} = \mathbf{u}_0 \cdot \sum_{m=1}^{M} \mathbf{F}_m \Delta\mathbf{s} + \sum_{m=1}^{M} \sum_{\mathbf{x}\in\mathbb{E}} \mathbf{a}(\mathbf{x}) \times (\nabla\delta_h)(\mathbf{x} - \mathbf{X}_m) \cdot (\mathbf{F}_m \Delta\mathbf{s})\, h^3$$

[3]Here and in similar expressions, $\sum_{\mathbf{x}\in\mathbb{F}} \mathbf{u}(\mathbf{x}) \cdot \mathbf{f}(\mathbf{x}) h^3$ is a shorthand for $\sum_{i=1}^{3} \sum_{\mathbf{x}\in\mathbb{F}} u_i(\mathbf{x}) f_i(\mathbf{x}) h^3$.

$$= \mathbf{u}_0 \cdot \sum_{m=1}^{M} \mathbf{F}_m \Delta \mathbf{s} + \sum_{\mathbf{x}\in\mathbb{E}} \mathbf{a}(\mathbf{x}) \cdot \sum_{m=1}^{M} (\nabla \delta_h)(\mathbf{x} - \mathbf{X}_m) \times (\mathbf{F}_m \Delta \mathbf{s})\, h^3. \tag{2.4.13}$$

Since $\mathbf{u}_0$ and $\mathbf{a}(\mathbf{x})$ are arbitrary (except for $\mathbf{D}^h \cdot \mathbf{a} = 0$), the power identity Eq. (2.4.10) is satisfied if and only if

$$\mathbf{f}_0 = \frac{1}{V} \sum_{m=1}^{M} \mathbf{F}_m \Delta \mathbf{s} \tag{2.4.14}$$

and

$$\mathbf{D}^h \times \mathbf{f} = \sum_{k=1}^{M} (\nabla \delta_h)(\mathbf{x} - \mathbf{X}_m) \times (\mathbf{F}_m \Delta \mathbf{s}) + \mathbf{G}^h \varphi, \quad \text{for all } \mathbf{x} \in \mathbb{E}, \tag{2.4.15}$$

where φ is an arbitrary scalar field that lives on the node-centered grid $\mathbb{N}$. Note that we have the freedom to add the term $\mathbf{G}^h \varphi$ in Eq. (2.4.15), since from the identity Eq. (2.8.4) and $\mathbf{D}^h \cdot \mathbf{a} = 0$, we have

$$\sum_{\mathbf{x}\in\mathbb{E}} \mathbf{a}(\mathbf{x}) \cdot \left(\mathbf{G}^h \varphi\right) h^3 = - \sum_{\mathbf{x}\in\mathbb{N}} \left(\mathbf{D}^h \cdot \mathbf{a}\right)(\mathbf{x})\, \varphi(\mathbf{x})\, h^3 = 0.$$

Indeed, we are required to include this term since the left-hand side of Eq. (2.4.15) is discretely divergence-free but there is no reason to expect the first term on the right-hand side of Eq. (2.4.15) is also divergence-free. Note that it is not required to find φ in order to determine $\mathbf{f}(\mathbf{x})$, because we can eliminate φ by taking the discrete curl on both sides of Eq. (2.4.15),

$$\mathbf{D}^h \times (\mathbf{D}^h \times \mathbf{f}) = \mathbf{D}^h \times \left(\sum_{m=1}^{M} (\nabla \delta_h)(\mathbf{x} - \mathbf{X}_m) \times (\mathbf{F}_m \Delta \mathbf{s}) \right), \quad \text{for all } \mathbf{x} \in \mathbb{E}. \tag{2.4.16}$$

By imposing the gauge condition

$$\mathbf{D}^h \cdot \mathbf{f} = 0, \tag{2.4.17}$$

we obtain a vector Poisson equation for $\mathbf{f}(\mathbf{x})$,

$$-(\mathbf{L}^h \mathbf{f})(\mathbf{x}) = \mathbf{D}^h \times \left(\sum_{m=1}^{M} (\nabla\delta_h)(\mathbf{x} - \mathbf{X}_m) \times (\mathbf{F}_m \Delta\mathbf{s}) \right), \text{ for all } \mathbf{x} \in \mathbb{E}. \tag{2.4.18}$$

Note again that $\nabla\delta_h$ is computed on $\mathbb{E}$, so that the cross-product with $\mathbf{F}_m$ is face-centered, which agrees with the left-hand side of Eq. (2.4.18). Note that the solution of Eq. (2.4.18) can be uniquely determined by the choice of $\mathbf{f}_0$. Like our velocity interpolation scheme, the new force-spreading scheme is also non-local because it requires solving discrete Poisson equations. We remark that the new force-spreading scheme is also constructed so that the resulting force density $\mathbf{f}(\mathbf{x})$ is discretely divergence-free. This means that $\mathbf{f}(\mathbf{x})$ includes the pressure gradient that is generated by the Lagrangian forces. We do not see a straightforward way to separate the pressure gradient from $\mathbf{f}(\mathbf{x})$ in case it is needed for output purposes.

2.5 Time-stepping scheme

The spatially-discretized equations of motion are

$$\rho \left(\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} + \boldsymbol{N}(\mathbf{u}) \right) + \mathbf{G}^h p = \mu \mathbf{L}^h \mathbf{u} + \boldsymbol{S}[\mathbf{X}]\mathbf{F}, \tag{2.5.1}$$

$$\mathbf{D}^h \cdot \mathbf{u} = 0, \tag{2.5.2}$$

$$\frac{\mathrm{d}\mathbf{X}}{\mathrm{d}t} = \mathbf{U}(\mathbf{X}, t) = \boldsymbol{S}^\star[\mathbf{X}]\mathbf{u}. \tag{2.5.3}$$

In this section, we present a second-order time-stepping scheme, similar to the ones developed previously [16], that evolves the spatially-discretized system Eqs. (2.5.1) to (2.5.3). Let $\mathbf{u}^n, \mathbf{X}^n$ denote the approximations of the fluid velocity and Lagrangian marker velocities at time $t_n = n\Delta t$. To advance the solutions to $\mathbf{u}^{n+1}$ and $\mathbf{X}^{n+1}$, we perform the following steps:

1. First, update the Lagrangian markers to the intermediate time step $n + \frac{1}{2}$ using the

interpolated velocity,

$$\widetilde{\mathbf{X}}^{n+\frac{1}{2}} = \mathbf{X}^n + \frac{\Delta t}{2}\boldsymbol{S}^{\star}\left[\mathbf{X}^n\right]\mathbf{u}^n. \tag{2.5.4}$$

2. Evaluate the intermediate Lagrangian force density at $\widetilde{\mathbf{X}}^{n+\frac{1}{2}}$ from the force density functional or the energy functional, and spread it to the Eulerian grid using the force-spreading scheme to get

$$\mathbf{f}^{n+\frac{1}{2}} = \boldsymbol{S}\left[\widetilde{\mathbf{X}}^{n+\frac{1}{2}}\right]\mathbf{F}^{n+\frac{1}{2}}. \tag{2.5.5}$$

3. Solve the fluid equations on the periodic grid [35],

$$\begin{cases} \rho\left(\dfrac{\mathbf{u}^{n+1}-\mathbf{u}^n}{\Delta t} + \widetilde{\mathbf{N}}^{n+\frac{1}{2}}\right) + \mathbf{G}^h p^{n+\frac{1}{2}} = \mu\mathbf{L}^h\left(\dfrac{\mathbf{u}^{n+1}+\mathbf{u}^n}{2}\right) + \mathbf{f}^{n+\frac{1}{2}}, \\ \\ \mathbf{D}^h\cdot\mathbf{u}^{n+1} = 0, \end{cases} \tag{2.5.6}$$

   where the second-order Adams-Bashforth (AB2) method is applied to approximate the nonlinear advection term

$$\widetilde{\mathbf{N}}^{n+\frac{1}{2}} = \frac{3}{2}\mathbf{N}^n - \frac{1}{2}\mathbf{N}^{n-1}, \tag{2.5.7}$$

   and $\mathbf{N}^n = \boldsymbol{N}(\mathbf{u}^n)$.

4. In the last step, update the Lagrangian markers $\mathbf{X}^{n+1}$ by using the mid-point approximation

$$\mathbf{X}^{n+1} = \mathbf{X}^n + \Delta t\ \boldsymbol{S}^{\star}\left[\widetilde{\mathbf{X}}^{n+\frac{1}{2}}\right]\left(\frac{\mathbf{u}^{n+1}+\mathbf{u}^n}{2}\right). \tag{2.5.8}$$

Note that the time-stepping scheme described above requires two starting values because of the treatment of the nonlinear advection term using the AB2. To get the starting value at

	# of scalar Poisson solves				# of scalar interpolation/spreading			
	2D		3D		2D		3D	
	DFIB	IBMAC	DFIB	IBMAC	DFIB	IBMAC	DFIB	IBMAC
$\boldsymbol{S}^\star$ in Eq. (2.5.4)	1	-	3	-	2	2	6	3
$\boldsymbol{S}$ in Eq. (2.5.5)	2	-	3	-	2	2	6	3
Fluid solver	3	3	4	4	-	-	-	-
$\boldsymbol{S}^\star$ in Eq. (2.5.8)	1	-	3	-	2	2	6	3
Total	7	3	13	4	6	6	18	9

Table 2.3: Cost of DFIB versus IBMAC in terms of the number of scalar Poisson solves and interpolation/spreading of a scalar from/to the Eulerian grid.

$t = \Delta t$, we can use the second-order Runge-Kutta (RK2) scheme described in [10, 35].

In Table 2.3 we compare the cost of DFIB and IBMAC for the above IB scheme in terms of the number of the two cost-dominating procedures: the scalar Poisson solver which costs $\mathcal{O}(N^d \log N)$ using FFT on the periodic domain, where $d \in \{2, 3\}$ is the spatial dimension, and spreading/interpolation of a scalar field between the Eulerian grid and the Lagrangian mesh which costs $\mathcal{O}(M)$. In summary, DFIB is only more expensive than IBMAC by 4 scalar Poisson solves for two-dimensional (2D) problems, and is more expensive by 9 scalar Poisson solves and 9 scalar interpolation and spreading for three dimensional (3D) problems. Therefore, the DFIB method is about two times slower than IBMAC per time step in 3D. We point out that if the RK2 scheme [10, 35] is employed rather than the scheme above, then we can save one interpolation step per time step, but the fluid equations need to be solved twice.

2.6 Numerical Results

This section presents numerical results of the DFIB method for various benchmark problems in 2D and 3D. We first consider in 2D a thin elastic membrane subject to surface tension of the membrane only. The continuum solution of this simple 2D problem has the special feature that the tangential component of the elastic force vanishes, and therefore, the normal derivative of the tangential fluid velocity does not suffer any jump across the immersed boundary. This has the effect that second-order convergence in the fluid velocity $\boldsymbol{u}$ and the Lagrangian deformation map $\boldsymbol{\mathcal{X}}$ can be achieved [25]. In the second set of tests, we compare volume conservation in 2D, i.e., area conservation of DFIB and IBMAC by applying them to a circular membrane under tension, and we discuss the connection between area conservation and the choice of Lagrangian marker spacing relative to the Eulerian grid size. In the third set of computations, we apply the DFIB method to a problem in which a 2D elastic membrane actively evolves in a parametrically-forced system. In the last set of numerical experiments, we extend the surface tension problem to 3D, and compare volume conservation of DFIB with that of IBMAC.

2.6.1 A thin elastic membrane with surface tension in 2D

It is well-known that the solutions to problems involving an infinitely thin massless membrane interacting with a viscous incompressible fluid possess jump discontinuities across the interface in the pressure and in the normal derivative of the velocity due to singular forcing at the interface [28, 47]. These sharp jump discontinuities cannot be fully resolved by the conventional IB method because of the use of the regularized delta function at the interface. Consequently, the numerical convergence rate for the Lagrangian deformation map $\boldsymbol{\mathcal{X}}$ is generally only first order even if the discretization is carried out with second-order accuracy. To achieve the expected rate of convergence, we consider problems with solutions that possess sufficient smoothness.

As a simple benchmark problem with a sufficiently smooth continuum solution we consider a thin elastic membrane that deforms in response to surface tension only. Suppose that the elastic interface Γ is discretized by a collection of Lagrangian markers $\mathbf{X} = \{\mathbf{X}_1, \ldots, \mathbf{X}_M\}$.

The discrete elastic energy functional associated with the surface tension of the membrane is the total (polygonal) arc-length of the interface [20],

$$E[\mathbf{X}_1, \ldots, \mathbf{X}_M] = \gamma \sum_{m=1}^{M} \left|\mathbf{X}_m - \mathbf{X}_{m-1}\right|, \tag{2.6.1}$$

where $\mathbf{X}_0 = \mathbf{X}_M$ and γ is the surface tension constant (energy per unit length). The Lagrangian force generated by the energy functional at the marker $\mathbf{X}_m$ is

$$\mathbf{F}_m \Delta s = -\frac{\partial E}{\partial \mathbf{X}_m} = \gamma \left(\frac{\mathbf{X}_{m+1} - \mathbf{X}_m}{|\mathbf{X}_{m+1} - \mathbf{X}_m|} - \frac{\mathbf{X}_m - \mathbf{X}_{m-1}}{|\mathbf{X}_m - \mathbf{X}_{m-1}|} \right). \tag{2.6.2}$$

In our tests, we set the initial configuration of the membrane to be the ellipse

$$\boldsymbol{\mathcal{X}}(s,0) = L \cdot \left(\frac{1}{2} + \frac{5}{28}\cos(s),\ \frac{1}{2} + \frac{7}{20}\sin(s) \right), \quad s \in [0, 2\pi]. \tag{2.6.3}$$

The Eulerian fluid domain $\Omega = [0, L]^2$ is discretized by a uniform $N \times N$ Cartesian grid with meshwidth $h = \frac{L}{N}$ in each direction. The elastic interface Γ is discretized by a uniform Lagrangian mesh of size $M = \lceil \pi N \rceil$ in the Lagrangian variable s, so that the Lagrangian markers $\mathbf{X} = \{\mathbf{X}_1, \ldots, \mathbf{X}_M\}$ are physically separated by a distance of approximately $\frac{h}{2}$ in the equilibrium circular configuration. In all of our tests, we set $L = 5,\ \rho = 1,\ \gamma = 1,\ \mu = 0.1$. The time step size is chosen to be $\Delta t = \frac{h}{2}$ to ensure the stability of all simulations up to $t = 20$ when the elastic interface is empirically observed to be in equilibrium.

We denote by $\mathbf{u}^N(t)$ the computed fluid velocity field and by $\mathcal{I}^{2N \to N}$ a restriction operator from the finer grid of size $2N \times 2N$ to the coarser grid of size $N \times N$. The discrete l_p-norm of the successive error in the velocity component u_i is defined by

$$\varepsilon^N_{p,u,i}(t) = \left\| u_i^N(t) - \mathcal{I}^{2N \to N} u_i^{2N}(t) \right\|_p. \tag{2.6.4}$$

To avoid artifacts in the error-norm computation because of Lagrangian markers getting too clustered during the simulation, we reparametrize the interface (for the purpose of the error computation only) from the computed markers using periodic cubic splines after each time step, and compute the l_p-norm error of $\mathbf{X}$ based on a collection of M' uniformly sampled markers $\widetilde{\mathbf{X}}$ from the reparametrized interface, that is,

$$\varepsilon_{p,\mathbf{X}}^{N}(t) = \left\| \widetilde{\mathbf{X}}^{N}(t) - \widetilde{\mathbf{X}}^{2N}(t) \right\|_{p}, \tag{2.6.5}$$

where M' does not change with N. We emphasize that the resampled markers are only used to compute the error norm and are discarded after each time step. In Fig. 2.6 the successive l_∞-norm and l_2-norm errors of the x, y-component of the fluid velocity and of the deformation map are plotted as a function of time from $t = 0$ to $t = 20$ for grid resolution $N = 64, 128$ and 256. The number of resampled markers for computing $\varepsilon_{p,\mathbf{X}}^{N}(t)$ is $M' = 128$. To clearly visualize that second-order convergence is achieved by our scheme, we multiply the computed errors for the finer grid resolution $N = 128$ and 256 by a factor of 4 and 4^2 respectively, and plot them along with errors for the coarser grid resolution $N = 64$ in Fig. 2.6. The observation that all three error curves almost align with each other (as shown in Fig. 2.6) confirms that second-order convergence in $\mathbf{u}$ and $\mathbf{X}$ is achieved.

vo

2.6.2 Area conservation and IB marker spacing

As an immediate consequence of fluid incompressibility, the volume enclosed by a closed immersed boundary should be exactly conserved as it deforms and moves with the fluid. However, it is observed that even in the simplest scenario of a pressurized membrane in its circular equilibrium configuration [31], the volume error of an IB method with conventional interpolation and spreading systematically grows at a rate proportional to the pressure jump

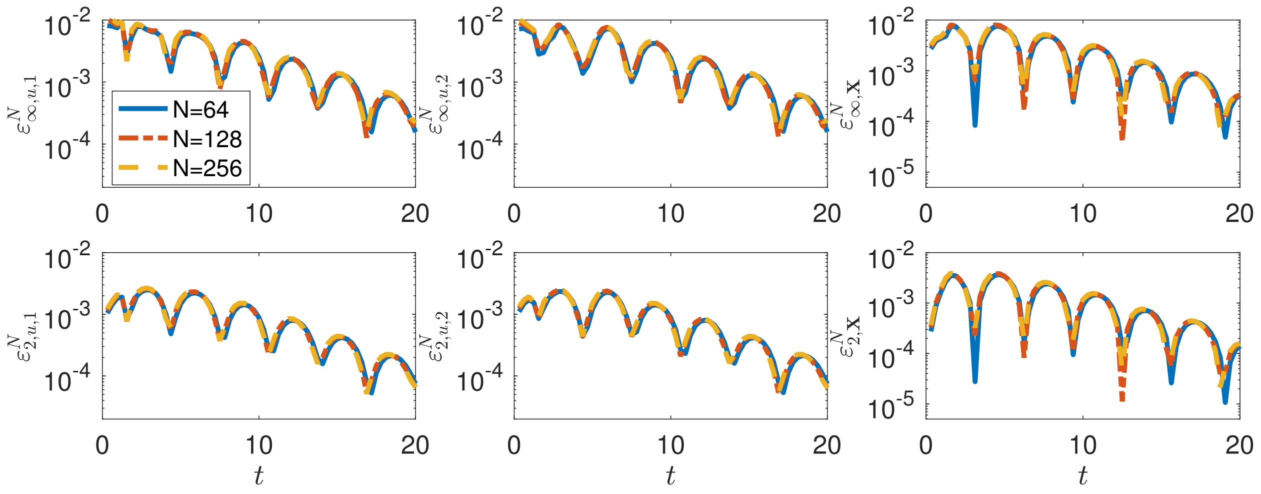


Fig. 2.6: Rescaled l_∞-norm (top panel) and l_2-norm (bottom panel) errors of the x, y-component of the fluid velocity defined by Eq. (2.6.4), and errors of the Lagrangian deformation map defined by Eq. (2.6.5) for the 2D surface tension problem are plotted as a function of time from $t = 0$ to $t = 20$. The left and middle columns show errors of the fluid velocity components and the right column shows errors of the Lagrangian deformation map. The Eulerian grid sizes are $N = 64, 128, 256$ and the corresponding Lagrangian mesh sizes are $M = 202, 403, 805$, so that the spacing between two Lagrangian markers is kept at a distance of approximately $\frac{h}{2}$ in the equilibrium configuration. For the finer grid resolution $N = 128, 256$, the errors in each norm are multiplied by a factor of 4 and 4^2 respectively. After rescaling, the error curves of the finer grid resolution almost align with the error curves of grid resolution $N = 64$, which indeed confirms that second-order convergence in $\mathbf{u}$ and $\mathbf{X}$ is achieved. For this set of computations, we use the $\mathscr{C}^3$ 6-point IB kernel in the discrete delta function, and the time step size is chosen to be $\Delta t = \frac{h}{2}$.

across the elastic interface [26]. In this set of tests, we demonstrate that, the "volume" or area enclosed by a 2D membrane is well-conserved by the DFIB method when the Lagrangian interface is sufficiently resolved.

We follow the same problem setup as in the test described in [31]. A thin elastic membrane $\boldsymbol{\mathcal{X}}(s,t)$, initially in a circular equilibrium configuration,

$$\boldsymbol{\mathcal{X}}(s,0) = \left(\frac{1}{2} + \frac{1}{4}\cos(s), \frac{1}{2} + \frac{1}{4}\sin(s)\right), \quad s \in [0, 2\pi], \tag{2.6.6}$$

is immersed in a periodic unit cell $\Omega = [0,1]^3$ with zero initial background flow. The Lagrangian force density on the interface is described by

$$\boldsymbol{F}(s,t) = \kappa \frac{\partial^2 \boldsymbol{\mathcal{X}}}{\partial s^2}, \tag{2.6.7}$$

in which κ is the uniform stiffness coefficient. The elastic membrane is discretized by a uniform Lagrangian mesh of M points in the variable s. We approximate the Lagrangian force density by

$$\mathbf{F}_m = \frac{\kappa}{(\Delta s)^2} (\mathbf{X}_{m+1} - 2\mathbf{X}_m + \mathbf{X}_{m-1}), \tag{2.6.8}$$

which corresponds to a collection of Lagrangian markers connected by linear springs of zero rest length with stiffness κ. For this problem, since the elastic interface is initialized in the equilibrium configuration with zero background flow, any spurious fluid velocity and area loss incurred in the simulation are regarded as numerical errors.

In our simulations, we set $\rho = 1,\ \mu = 0.1,\ \kappa = 1$. The size of the Eulerian grid is fixed at 128×128 with meshwidth $h = \frac{1}{128}$. The size of the Lagrangian mesh M is chosen so that two adjacent Lagrangian markers are separated by a physical distance of h_s in the equilibrium configuration, that is, $M \approx 2\pi R/h_s$, where R is the radius of the circular membrane. In addition to the Lagrangian markers, we also include a dense collection of passive tracers with

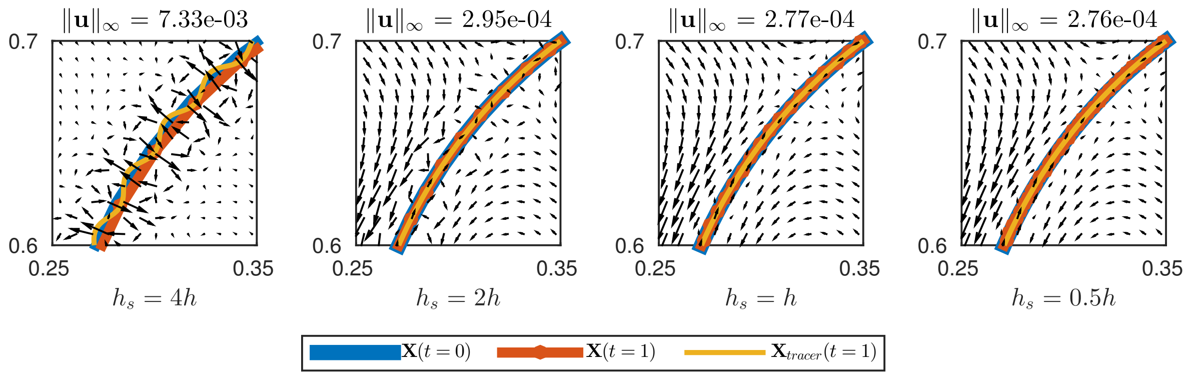
$N_{\text{tracer}} = 20M$ to address the limiting case of moving the entire interface. These tracers are initially in the same configuration as the circular membrane in Eq. (2.6.6), and they move passively with the interpolated velocity according to Eqs. (2.5.4) and (2.5.8). The time step size is set to be $\Delta t = \frac{h}{4}$ for stability. In all computations, we use the $\mathscr{C}^3$ 6-point IB kernel ϕ_{6h}^{new} to form the regularized delta function δ_h.

In Fig. 2.7 we compare the computational results of DFIB with those of IBMAC for different $h_s = 4h$, $2h$, h and $\frac{h}{2}$ (from left to right in Fig. 2.7). Each subplot of Fig. 2.7 shows a magnified view of the same arc of the circular interface along with its nearby spurious fluid velocity field. The interface represented by the Lagrangian markers $\mathbf{X}(t = 1)$ is shown in red and the initial configuration $\mathbf{X}(t = 0)$ is shown in the blue curve. The interface represented by the passive tracers $\mathbf{X}_{\text{tracer}}(t = 1)$ is shown in the yellow curve. In the first column of Fig. 2.7 in which $h_s = 4h$, we see that the maximum spurious velocity $\|\mathbf{u}\|_\infty$ of IBMAC is of the same magnitude as that of DFIB. At such coarse resolution in the Lagrangian mesh, fluid apparently leaks through the gap between two adjacent markers, as can be observed by the wiggly pattern in the passive tracers. As the the Lagrangian mesh is refined gradually from $h_s = 4h$ to $\frac{h}{2}$ (from left to right in Fig. 2.7), we see that $\|\mathbf{u}\|_\infty$ decreases from 10^{-3} to 10^{-7} in the DFIB method, whereas $\|\mathbf{u}\|_\infty$ stops improving around 10^{-4} in IBMAC. Moreover, in the columns where $h_s = 2h$, h, $\frac{h}{2}$, we see a clear global pattern in the spurious velocity field in IBMAC, while the spurious velocity field of DFIB appears to be much smaller in magnitude and random in pattern.

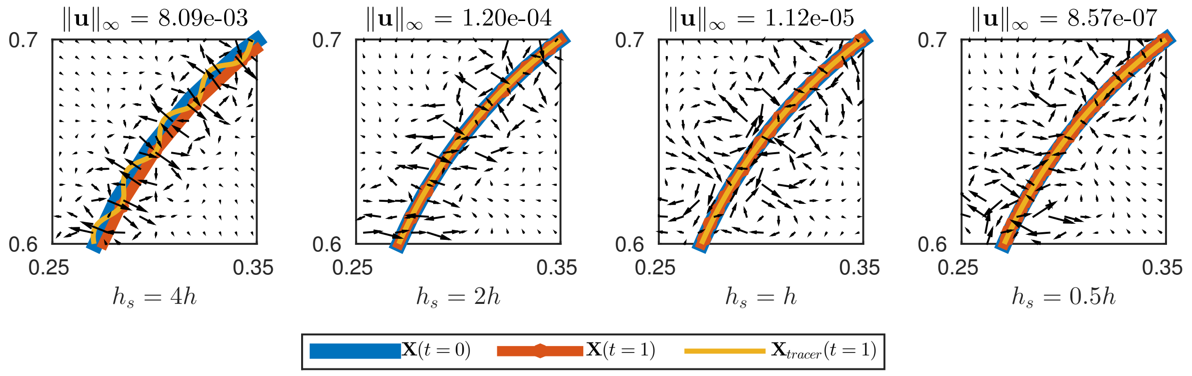
We define the normalized area error with respect to the initial configuration

$$\Delta A(t; \mathbf{X}) := \frac{|A(t; \mathbf{X}) - A(0; \mathbf{X})|}{A(0; \mathbf{X})}, \tag{2.6.9}$$

where the area enclosed by the Lagrangian markers $A(t; \mathbf{X})$ is approximated by the area of the polygon formed by the Lagrangian markers $\mathbf{X} = \{\mathbf{X}_1, \ldots, \mathbf{X}_M\}$ at time t. Fig. 2.8



(a) IBMAC



(b) DFIB

Fig. 2.7: A magnified view of the quasi-static circular membrane and its nearby spurious velocity field for different Lagrangian mesh spacing $h_s = 4h$, $2h$, h and $\frac{h}{2}$, as indicated below each figure panel, while keeping $h = \frac{1}{128}$ fixed. The top panel (a) shows the computational results from IBMAC, and the bottom panel (b) shows the results from DFIB. The interface represented by the Lagrangian markers $\mathbf{X}(t = 1)$ is shown in red, the initial configuration $\mathbf{X}(t = 0)$ is shown in blue, and the interface represented by $N_{\text{tracer}} = 20M$ passive tracers is shown in yellow, where M is the number of Lagrangian markers. The time step size is set to be $\Delta t = \frac{h}{4}$ for stability. In the above computations, the $\mathscr{C}^3$ 6-point IB kernel ϕ_{6h}^{new} is used in IBMAC and DFIB.

and Fig. 2.9 show the normalized area errors defined by Eq. (2.6.9) for DFIB and IBMAC with different choices of the IB kernels: $\phi_{4h} \in \mathscr{C}^1$, $\phi_{4h}^{\mathrm{B}} \in \mathscr{C}^2$, $\phi_{5h}^{\mathrm{new}} \in \mathscr{C}^3$, $\phi_{6h}^{\mathrm{new}} \in \mathscr{C}^3$ and $\phi_{6h}^{\mathrm{B}} \in \mathscr{C}^4$. For the coarse Lagrangian marker spacings, for example, when $h_s = 2h$, $4h$, the area errors for IBMAC and DFIB have similar orders of magnitude (compare Fig. 2.8a, 2.8b to Fig. 2.8d, 2.8e). As the Lagrangian marker spacing is reduced from $2h$ to h, we see a decrease in $\Delta A(t; \mathbf{X})$ for IBMAC by approximately a factor of 10 (see Fig. 2.8b, 2.8c) for all the IB kernels we consider in this set of tests. In contrast, the area errors for DFIB improve by at least a factor of 10^3 for the IB kernels that are at least $\mathscr{C}^2$ (see Fig. 2.8e, 2.8f), and in the best scenario, $\Delta A(t; \mathbf{X})$ for ϕ_{6h}^{B} decreases from 10^{-4} to 10^{-9}. Moreover, as the Lagrangian mesh is refined from h to $\frac{h}{8}$, area errors for DFIB keep improving, even approaching the machine epsilon in double precision for ϕ_{6h}^{B} at $h_s = \frac{h}{4}, \frac{h}{8}$ and for ϕ_{6h}^{new} at $h_s = \frac{h}{8}$ (see Fig. 2.9e, 2.9f). For a moderate Lagrangian marker spacing, such as $h_s = h$ and $\frac{h}{2}$, area errors for DFIB are several orders of magnitude smaller than those of IBMAC. On the other hand, area errors for IBMAC stop improving around 10^{-5} for $h_s \leq h$, no matter how densely the Lagrangian mesh is refined (see Fig. 2.8c, 2.9c). We remark that the smoothness of the IB kernel appears to play an important role in volume conservation of DFIB. In this study DFIB achieves the best volume conservation result for $h_s \leq h$ with ϕ_{6h}^{B}, and this kernel also has the highest regularity of the kernel functions considered in this work.

The area errors of DFIB shown in Fig. 2.8 and Fig. 2.9 can be attributed to two sources of error. The first source of error is the time-stepping error from the temporal integrator, which is relatively small in the quasi-static circle test. The second source of area loss comes from discretizing the continuous curve (circle) as a polygon whose vertices are the IB markers. This kind of error can be substantially reduced by using a high-order representation of the interface, such as a periodic cubic spline. We define a normalized area error with respect to the true initial area of the interface A_{true} by using the tracers,

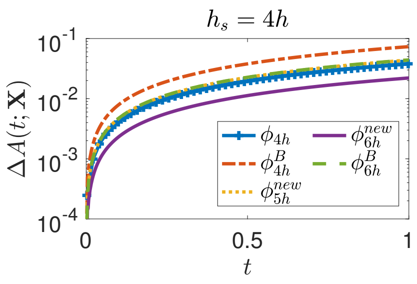
$$\Delta A(t;\mathbf{X}_{\text{tracer}}) := \frac{|A(t;\mathbf{X}_{\text{tracer}}) - A_{\text{true}}|}{A_{\text{true}}}, \tag{2.6.10}$$

where we compute $A(t;\mathbf{X}_{\text{tracer}})$ via exact integration of the cubic spline interpolant. Fig. 2.10 shows that the area enclosed by the passive tracers using the cubic spline approximation is far more accurately preserved than the polygonal approximation. Furthermore, the area error approaches zero as the discretization of the tracer interface is refined, even if the marker and grid spacings are held fixed, as shown in Fig. 2.10. Indeed, the area of the spline interpolant through the tracers of spacing $h_s/20 = h/20$ is conserved to the machine epsilon in double precision.

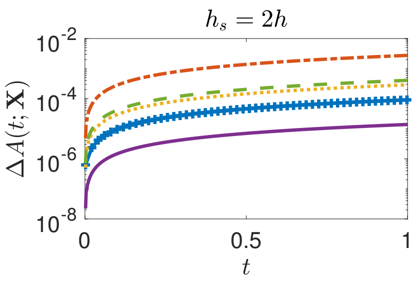
It is well-known that the traditional IB method produces non-smooth surface tractions, and a number of improvements have been proposed [48, 49, 44, 32, 50]. Somewhat unexpectedly, the divergence-free force spreading used in our DFIB method proposed here offers smoother and more accurate tractions without any post-processing such as filtering [48]. This is inherently linked to the reduced spurious flows compared to traditional methods [32]. In Fig. 2.11, we compare the errors of the tangential and normal components of $\mathbf{F}(s, t = 1)$ for DFIB and IBMAC for the quasi-static circle problem. We observe that the DFIB method dramatically improves the accuracy of Lagrangian forces by only refining the Lagrangian mesh, keeping the Eulerian grid fixed. By contrast, in the IBMAC method, the tractions do not improve as the Lagrangian grid is refined.

2.6.3 A thin elastic membrane with parametric resonance in 2D

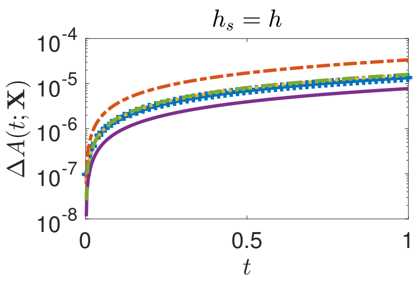
In many biological applications, the immersed structure is an active material, interacting dynamically with the surrounding fluid and generating time-dependent motion. It has been reported that the simulation of active fluid-structure interactions using the conventional



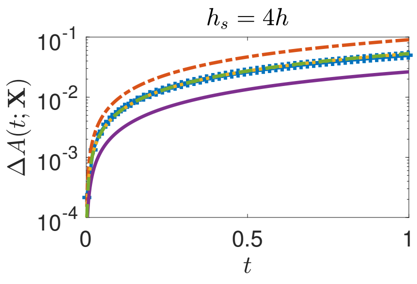
(a)



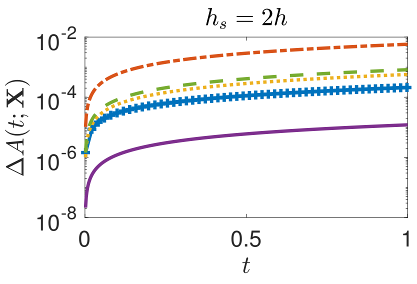
(b)



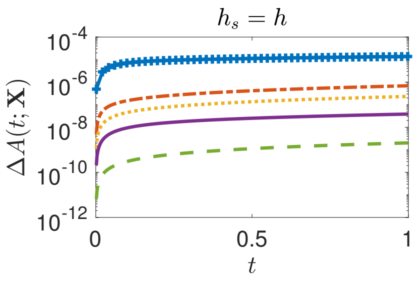
(c)



(d)

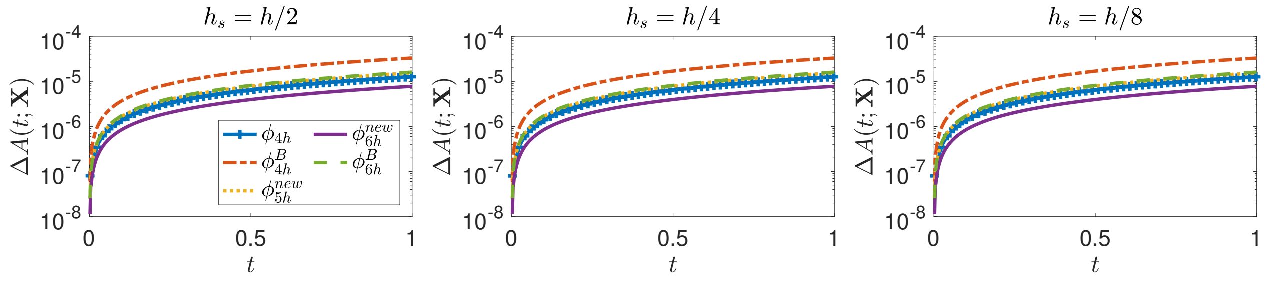


(e)

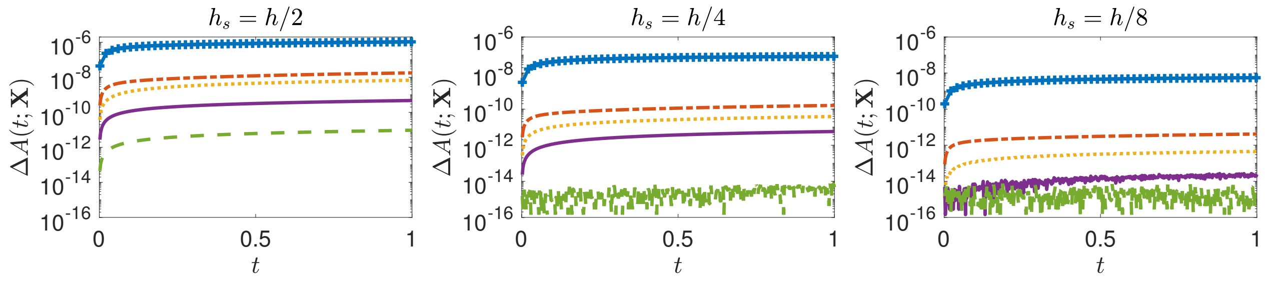


(f)

Fig. 2.8: Normalized area errors of the pressurized circular membrane (relative to the initial area, see (2.6.9)) simulated by IBMAC (top panel) and DFIB (bottom panel) with the IB kernels: $\phi_{4h} \in \mathscr{C}^1$, $\phi_{4h}^{\mathrm{B}} \in \mathscr{C}^2$, $\phi_{5h}^{\mathrm{new}} \in \mathscr{C}^3$, $\phi_{6h}^{\mathrm{new}} \in \mathscr{C}^3$ and $\phi_{6h}^{\mathrm{B}} \in \mathscr{C}^4$, and with Lagrangian marker spacings $h_s \in \{4h, 2h, h\}$ indicated above each figure panel.



(a) (b) (c)



(d) (e) (f)

Fig. 2.9: Normalized area errors of the pressurized circular membrane (relative to the initial area, see (2.6.9)) simulated by IBMAC (top panel) and DFIB (bottom panel) with the IB kernels: $\phi_{4h} \in \mathscr{C}^1$, $\phi_{4h}^{\mathrm{B}} \in \mathscr{C}^2$, $\phi_{5h}^{\mathrm{new}} \in \mathscr{C}^3$, $\phi_{6h}^{\mathrm{new}} \in \mathscr{C}^3$ and $\phi_{6h}^{\mathrm{B}} \in \mathscr{C}^4$, and with Lagrangian marker spacings, $h_s \in \left\{\frac{h}{2}, \frac{h}{4}, \frac{h}{8}\right\}$ indicated above each figure panel. As the Lagrangian mesh is refined, area errors for DFIB keep improving, even approaching the machine precision for ϕ_{6h}^{B} at $h_s = \frac{h}{4}, \frac{h}{8}$ and for ϕ_{6h}^{new} at $h_s = \frac{h}{8}$.

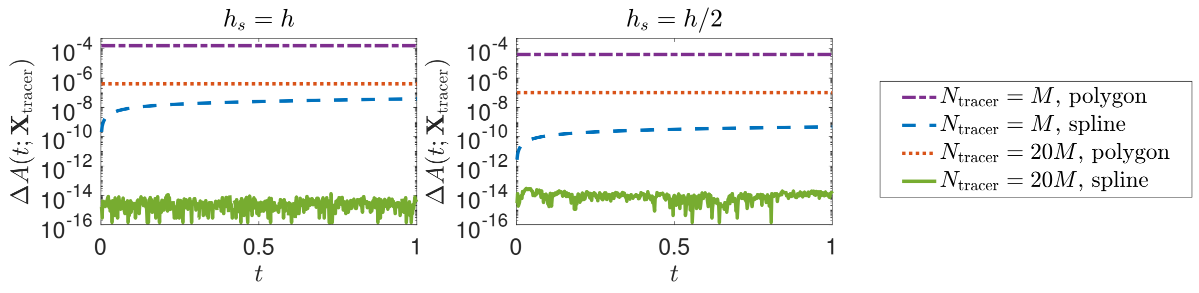
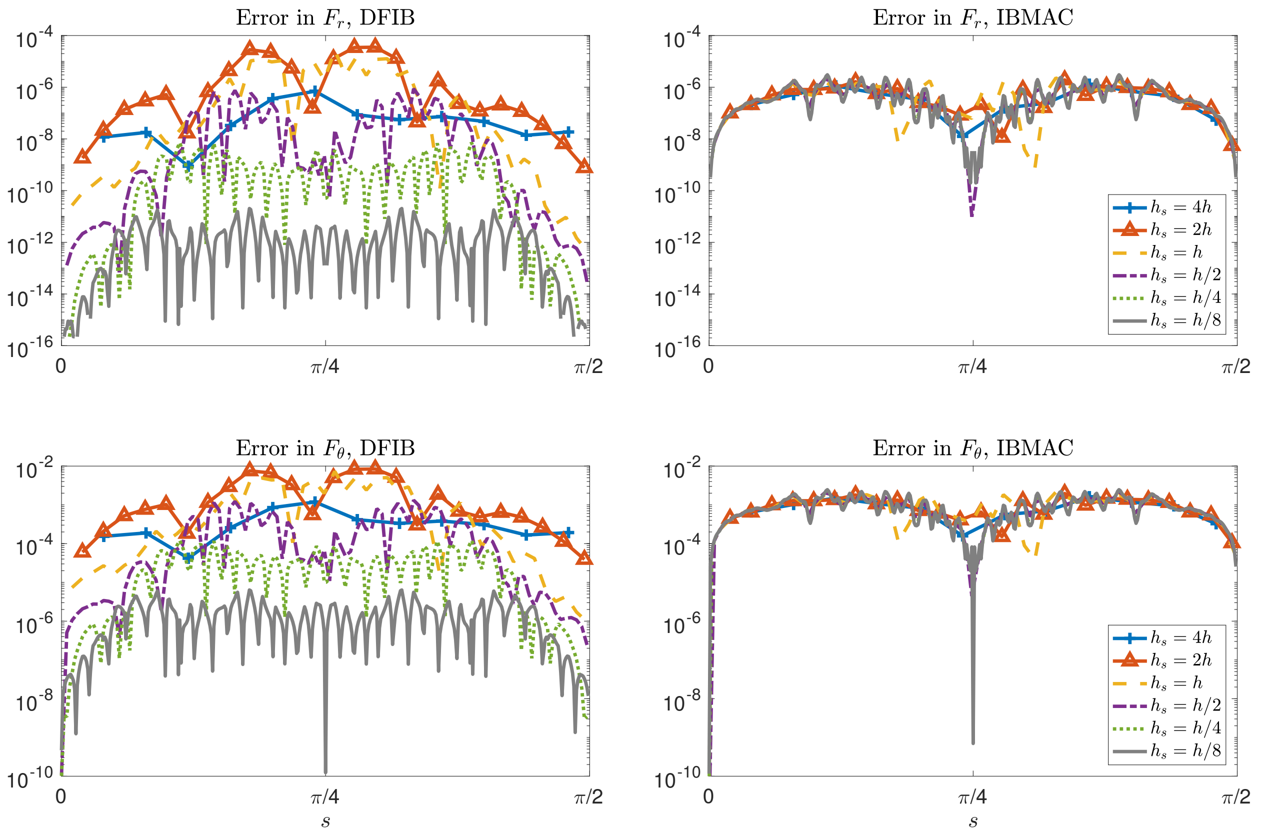


Fig. 2.10: Normalized area errors of the interface enclosed by the tracers that move passively with the interpolated velocity of the DFIB method (relative to the true area of the circle, see Eq. (2.6.10)). The initial configuration of the interface is given by Eq. (2.6.6), and ϕ_{6h}^{new} is used for this computation. From left to right, the Lagrangian marker spacing is $h_s \in \left\{h, \frac{h}{2}\right\}$, as shown in each figure panel. In each case, the area error enclosed by the tracers is computed for tracer resolution $N_{\mathrm{tracer}} = M$ and $20M$ in two ways: (1) by the area of the polygon formed by the tracers, and (2) by the exact integration of the cubic spline interpolant of the tracer interface.



(a) DFIB (b) IBMAC

Fig. 2.11: Normalized errors of the normal F_r (top panels) and tangential F_θ (bottom panels) components of the Lagrangian force $\mathbf{F}(s, t = 1)$ of the circular membrane for $s \in [0, \frac{\pi}{2}]$. The computations are performed using DFIB (left panels) and IBMAC (right panels) with ϕ_{6h}^{new}, and with Lagrangian marker spacings $h_s \in \left\{4h, 2h, h, \frac{h}{2}, \frac{h}{4}, \frac{h}{8}\right\}$.

IB method may suffer from significant loss in the volume enclosed by the structure [26]. A simple prototype problem for active fluid-structure interaction is a thin elastic membrane that dynamically evolves in a fluid in response to elastic forcing with periodic variation in the stiffness parameter [51, 52], that is,

$$\boldsymbol{F}(s,t) = \kappa(t)\frac{\partial^2 \boldsymbol{\mathcal{X}}}{\partial s^2}, \tag{2.6.11}$$

where $\kappa(t)$ is a periodic time-dependent stiffness coefficient of the form

$$\kappa(t) = K_c(1 + 2\tau \sin(\omega_0 t)). \tag{2.6.12}$$

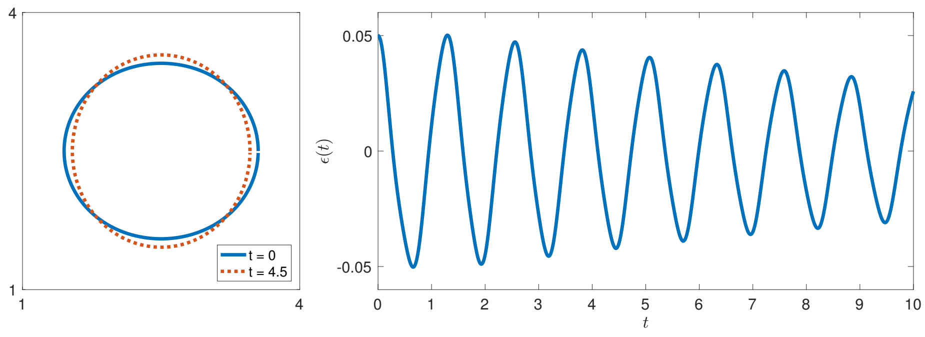
It is quite remarkable that such a purely temporal parameter variation can result in the emergence of spatial patterns, but that is indeed the case. We assume that the immersed structure is initially in a configuration that has a small-amplitude perturbation from a circle of radius R,

$$\boldsymbol{\mathcal{X}}(s,0) = R(1 + \epsilon_0 \cos(ps))\ \hat{\boldsymbol{r}}(s), \tag{2.6.13}$$

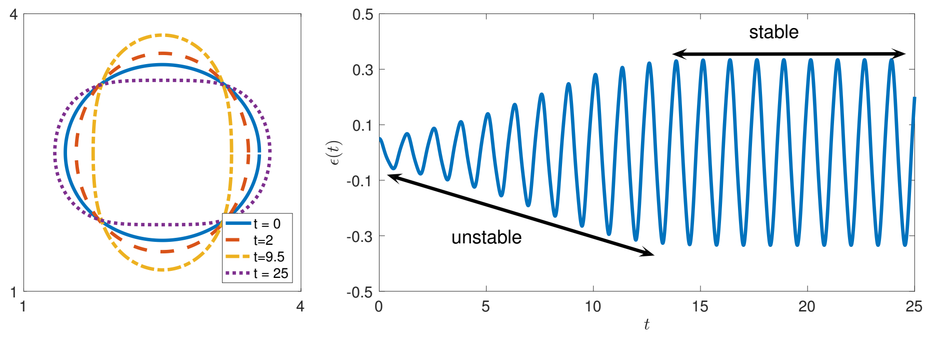
where $\hat{\boldsymbol{r}}(s)$ denotes the position vector pointing radially from the origin. For certain choices of parameters, the perturbed mode in the initial configuration may resonate with the driving frequency ω_0 in the periodic forcing, leading to large-amplitude oscillatory motion in the membrane. The stability of the parametric resonance has been studied in the IB framework using Floquet linear stability analysis for a thin elastic membrane in 2D [51, 52], and recently for an elastic shell in 3D [53]. Motivated by the linear stability analysis of [51, 52], we consider two sets of parameters listed in Table 2.4 for our simulations. The first set of parameters with $\tau = 0.4$ leads to a stable configuration in which the membrane undergoes damped oscillations (Fig. 2.12a), and the second set with $\tau = 0.5$ leads to an unstable configuration in which the membrane oscillates with growing amplitude (Fig. 2.12b).

ρ	μ	L	R	K_c	ω_0	p	ϵ_0	τ
1	0.15	5	1	10	10	2	0.05	0.4 (damped oscillation)
								0.5 (growing oscillation)

Table 2.4: Parameters used to simulate the motion of the 2D membrane with parametric resonance.



(a)



(b)

Fig. 2.12: Left panel: snapshots of the 2D membrane with parametric resonance. Right panel: the time-dependent amplitude $\epsilon(t)$ of the perturbed mode in Eq. (2.6.14). (a) Damped oscillation (b) Growing oscillation.

The computational domain $\Omega = [0, L]^2$ is discretized by a 128×128 uniform Cartesian grid with meshwidth $h = \frac{L}{128}$. The number of Lagrangian markers is determined so that the distance between the Lagrangian markers is $h_s \approx \frac{h}{2}$ in the initial configuration. The discretization of the Lagrangian force density Eq. (2.6.11) is constructed in the same way as Eq. (2.6.8). The time step size is $\Delta t = \frac{h}{10}$ to ensure the stability of computation. On the left panel of Fig. 2.12 we show snapshots of the membrane configuration for each case, and on the right panel we plot the time-dependent amplitude $\epsilon(t)$ of the ansatz

$$\boldsymbol{\mathcal{X}}(s,t) = R(1 + \epsilon(t)\cos(ps))\ \hat{\boldsymbol{r}}(s) \tag{2.6.14}$$

by applying the FFT to the Lagrangian marker positions $\mathbf{X}$. In the case of growing oscillation (Fig. 2.12b), the amplitude of the perturbed mode increases from 0.05 to 0.3 until nonlinearities eventually stabilize the growing mode and the membrane starts to oscillate at a fixed amplitude.

We next give a direct comparison of area conservation of IBModified (with $\phi_{4h}^{\cos}$ [26]), IBMAC, and DFIB with the IB kernels $\phi_{4h} \in \mathscr{C}^1$, $\phi_{4h}^{\mathrm{B}} \in \mathscr{C}^2$, $\phi_{5h}^{\mathrm{new}} \in \mathscr{C}^3$, $\phi_{6h}^{\mathrm{new}} \in \mathscr{C}^3$ and $\phi_{6h}^{\mathrm{B}} \in \mathscr{C}^4$. In this test, the area enclosed the Lagrangian markers is computed by the cubic spline approximation discussed in Sec. 2.6.2. In Fig. 2.13 and Fig. 2.14 we show time-dependent area errors enclosed by the parametric membrane for the damped-oscillation and the growing-amplitude cases respectively. For the damped-oscillation case (Fig. 2.13), we see that area errors for DFIB are at least two orders of magnitude smaller than those of IBMAC and IBModified for IB kernels that are at least $\mathscr{C}^2$. The volume conservation of IBModified and IBMAC was not directly compared in the previous work [31], but it was anticipated that they are similar. In our comparison, we find that IBModified is only slightly better than IBMAC in volume conservation, yet IBMAC is much simpler to use in practice.

In this set of tests, the choice of IB kernel also plays a role in affecting area conservation.

In particular, the area errors for DFIB using ϕ_{6h}^{new} and ϕ_{6h}^{B} are smaller than those of ϕ_{4h}^{B} and ϕ_{5h}^{new} by approximately one order of magnitude. Additionally, the error curves of DFIB with ϕ_{6h}^{new} and ϕ_{6h}^{B} remain oscillating below 10^{-7} while apparent growth of error in time is observed with ϕ_{4h}^{B} and ϕ_{5h}^{new}, and in the other IB methods. Unlike the quasi-static circle test in which the time-stepping error is negligible compared to the area loss due to moving a finite collection of Lagrangian markers, the time-stepping error in this example can be observed by considering a dense collection of tracers with different time step sizes. With $N_{\text{tracer}} = 4M$, we first confirm that the area error of the tracer interface cannot be reduced by further including more tracers, but as we reduce the time step size $\Delta t \in \left\{\frac{h}{10}, \frac{h}{20}, \frac{h}{40}\right\}$, we observe an improvement in the area error, as shown in the bottom panel of Fig. 2.13. The improvements in area conservation of DFIB is consistently more than 10^4 times over IBCollocated and about 10^3 times over IBMAC. Similar results are obtained for the growing-amplitude case (see Fig. 2.14) except that the parametrically-unstable membrane has experienced some area loss due to the growing-amplitude oscillation before its motion is stabilized by the nonlinearities.

2.6.4 A 3D thin elastic membrane with surface tension

In our final test problem, we examine volume conservation of the DFIB method by extending the surface tension problem to 3D. We consider in 3D a thin elastic membrane that is initially in its spherical equilibrium configuration. The spherical surface of the membrane is discretized by a triangulation consisting of approximately equilateral triangles with edge length approximately equal to h_s, constructed from successive refinement of a regular icosahedron by splitting each facet into four smaller equilateral triangles and projecting the vertices onto the sphere to form the refined mesh (see Fig. 2.15 for the first two levels of refinement). We use $\{\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_M\}$ and $\{\mathbf{T}_1, \mathbf{T}_2, \dots, \mathbf{T}_P\}$ to denote the vertices (Lagrangian markers) and the triangular facets of the mesh respectively. The generalization of discrete elastic energy

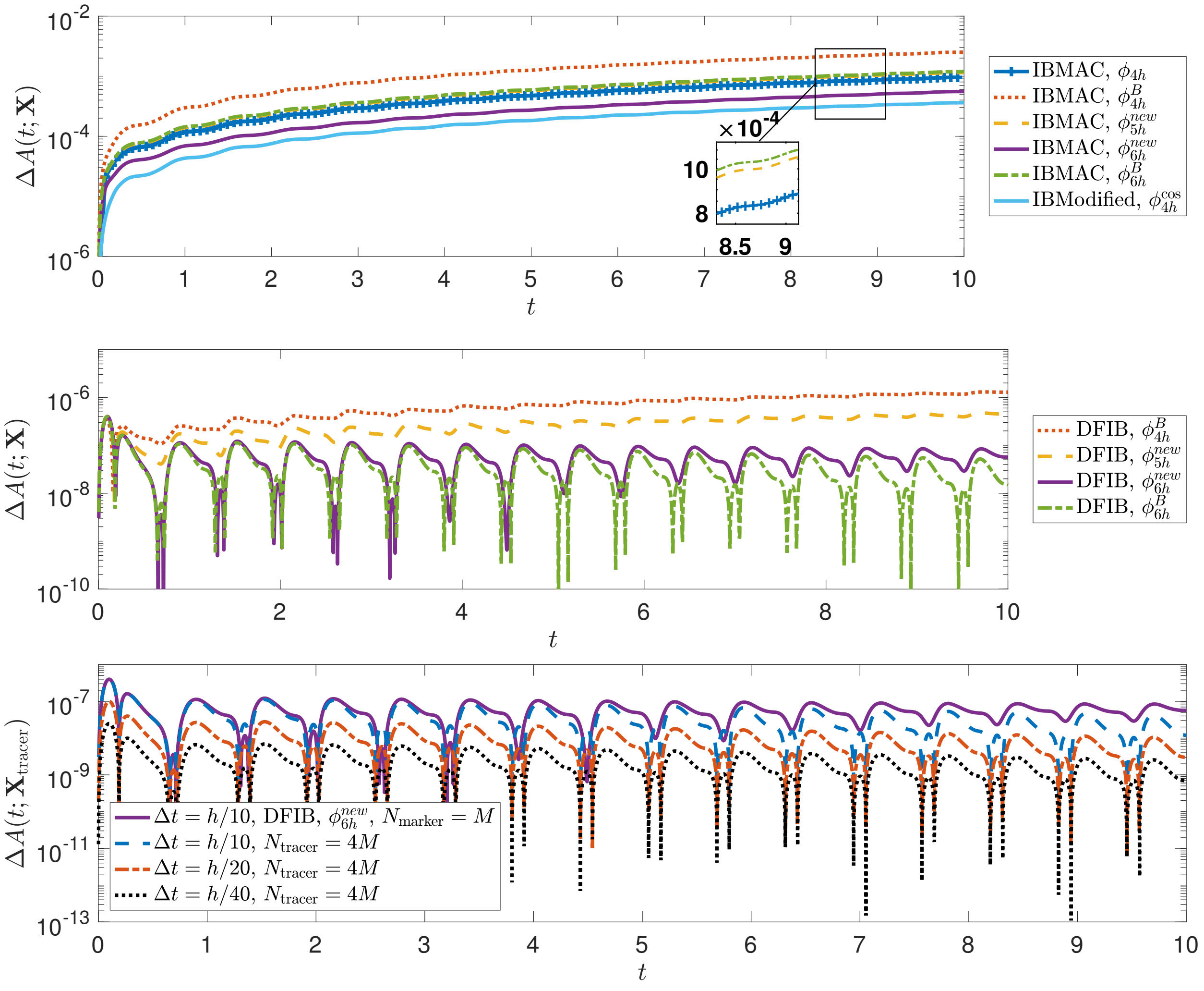


Fig. 2.13: Top and middle panels: normalized area errors $\Delta A(t;\mathbf{X})$ (relative to the initial area, see (2.6.9)) of the 2D parametric membrane undergoing damped oscillatory motion (corresponding to the motion shown in Fig. 2.12a) are plotted on the semi-log scale. The area enclosed by the markers is computed by the exact integration of a cubic spline approximation. The computations are performed using IBMAC and DFIB with the IB kernels: $\phi_{4h} \in \mathscr{C}^1$, $\phi_{4h}^{\mathrm{B}} \in \mathscr{C}^2$, $\phi_{5h}^{\mathrm{new}} \in \mathscr{C}^3$, $\phi_{6h}^{\mathrm{new}} \in \mathscr{C}^3$ and $\phi_{6h}^{\mathrm{B}} \in \mathscr{C}^4$, and IBModified with $\phi_{4h}^{\cos} \in \mathscr{C}^1$. The top panel shows area errors for IBMAC and IBModified, and the middle panel shows area errors for DFIB. The bottom panel shows improvement in area errors (relative to the true area of the ellipse, see (2.6.10)) enclosed by the interface of $N_{\mathrm{tracer}} = 4M$ tracers by reducing the time step size: $\Delta t \in \left\{\frac{h}{10}, \frac{h}{20}, \frac{h}{40}\right\}$.

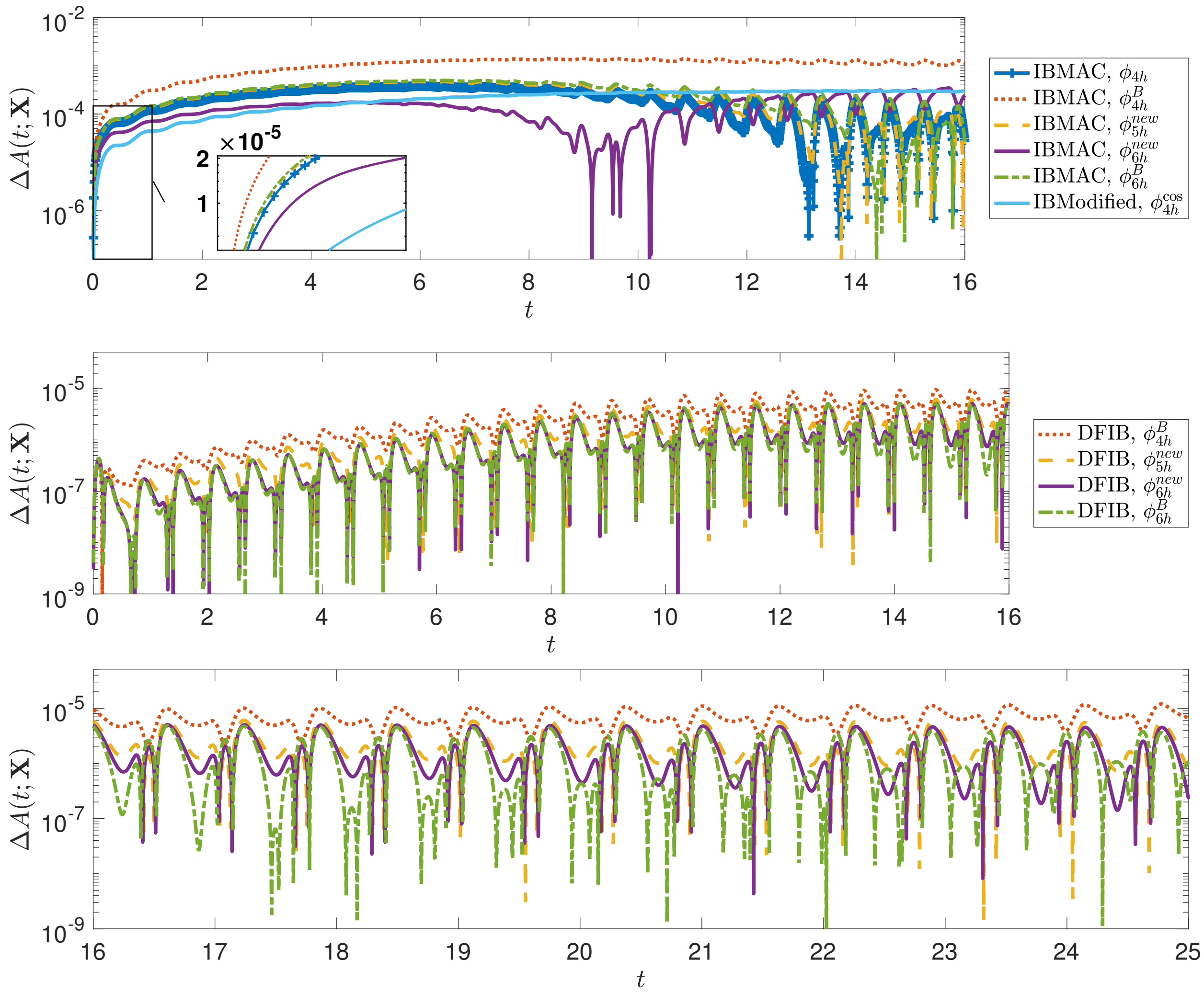


Fig. 2.14: Normalized area errors $\Delta A(t; \mathbf{X})$ of the 2D parametric membrane undergoing growing-amplitude oscillatory motion (corresponding to the motion shown in Fig. 2.12b) are plotted on the semi-log scale, as done in Fig. 2.13 for damped motion. The computations are performed using IBMAC and DFIB with the IB kernels: $\phi_{4h} \in \mathscr{C}^1$, $\phi^{\mathrm{B}}_{4h} \in \mathscr{C}^2$, $\phi^{\mathrm{new}}_{5h} \in \mathscr{C}^3$, $\phi^{\mathrm{new}}_{6h} \in \mathscr{C}^3$ and $\phi^{\mathrm{B}}_{6h} \in \mathscr{C}^4$, and IBModified with $\phi^{\cos}_{4h} \in \mathscr{C}^1$. The top panel shows area errors for IBMAC and IBModified, the middle panel shows the area errors for DFIB for $t = 0$ to 16, and the bottom panel extends the middle panel for $t = 16$ to 25.

functional of surface tension in 3D is the product of surface tension constant γ (energy per unit area) and the total surface area of the triangular mesh [21], that is,

$$E[\mathbf{X}_1, \ldots \mathbf{X}_M] = \gamma \sum_{p=1}^{P} |\mathbf{T}_p| \,, \tag{2.6.15}$$

where $|\mathbf{T}_p|$ is the area of the p^{th} triangle. The Lagrangian force $\mathbf{F}_k \Delta \mathbf{s}$ at the k^{th} vertex is minus the partial derivative of $E[\mathbf{X}_1, \ldots, \mathbf{X}_M]$ with respect to $\mathbf{X}_k$,

$$\mathbf{F}_k \Delta \mathbf{s} = -\frac{\partial E}{\partial \mathbf{X}_k} = -\gamma \sum_{l \,\in \text{nbor}(k)} \frac{\partial |\mathbf{T}_l|}{\partial \mathbf{X}_k} \,, \tag{2.6.16}$$

where nbor(k) denotes the set of indices of triangles that share $\mathbf{X}_k$ as a vertex[4]. Each component of the term $\partial|\mathbf{T}_l|/\partial \mathbf{X}_k$ in Eq. (2.6.16) can be computed analytically [21],

$$\begin{aligned} \left(\frac{\partial|\mathbf{T}_l|}{\partial \mathbf{X}_k}\right)_\alpha &= \frac{\partial}{\partial \mathbf{X}_{k,\alpha}} \left(\frac{1}{2}\left|(\mathbf{X}_k - \mathbf{X}'_k) \times (\mathbf{X}'_k - \mathbf{X}''_k)\right|\right) \\ &= \frac{1}{2}\left((\mathbf{X}'_k - \mathbf{X}''_k) \times \hat{\mathbf{n}}_l\right)_\alpha \,, \quad \alpha = 1, 2, 3, \end{aligned} \tag{2.6.17}$$

where $\mathbf{X}_k,\ \mathbf{X}'_k,\ \mathbf{X}''_k$ denote the three vertices of the triangle $\mathbf{T}_l$ ordered in the counterclockwise direction and $\hat{\mathbf{n}}$ is the unit outward normal vector of $\mathbf{T}_l$.

The computation is performed in the periodic box $\Omega = [0,1]^3$ with Eulerian meshwidth $h = \frac{1}{128}$ using DFIB with ϕ_{6h}^{new}. For the quasi-static test, the initial fluid velocity is set to be zero, and for the dynamic test, we set $\boldsymbol{u}(\boldsymbol{x}, 0) = (0,\ \sin(4\pi x),\ 0)$. In the computational results shown in Fig. 2.16, the spherical membrane is discretized by triangulation (as shown in Fig. 2.15) with 5 successive levels of refinement from the regular icosahedron (Fig. 2.15a), which results in a triangular mesh with $M = 10242$ vertices and $P = 20480$ facets. The radius

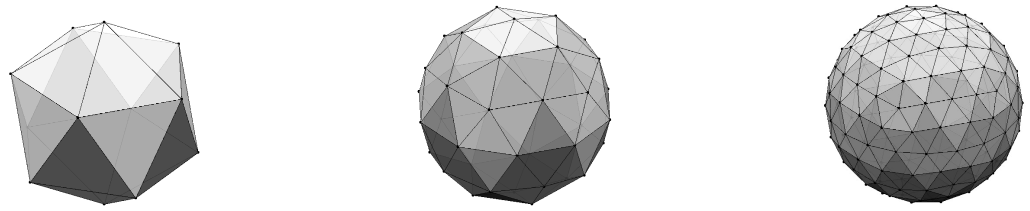
[4]Here $\Delta \mathbf{s}$ is the Lagrangian area associated with each node and $\mathbf{F}_k$ is the Lagrangian force density with respect to Lagrangian area, but note that we do not need $\mathbf{F}_k$ and $\Delta \mathbf{s}$ separately; only their product is used in the numerical scheme.

of the spherical membrane is set to be $R \approx 0.1$ which corresponds to $h_s \approx \frac{h}{2}$. The remaining parameters in the computation are $\rho = 1,\ \mu = 0.05,\ \gamma = 1$ and the time step size $\Delta t = \frac{h}{4}$. In Fig. 2.16 we show snapshots of the 3D elastic membrane at $t = 0,\ \frac{1}{32},\ \frac{1}{4}$ and $\frac{1}{2}$ for the dynamic case. The elastic interface is instantaneously deformed by the fluid flow in the y-direction, and due to surface tension, the membrane eventually relaxes back to the spherical equilibrium configuration. Colored markers that move passively with the divergence-free interpolated fluid velocity are added for visualizing the fluid flow in the vicinity of the interface.

The volume enclosed by the triangular surface mesh is approximated by the total volume of tetrahedra formed by each facet and one common reference point (e.g. the origin) using the scalar triple product. To study volume conservation of the DFIB method in 3D, we compare the normalized volume error defined by

$$\Delta V(t;\mathbf{X}) := \frac{|\,\mathrm{Vol}(t;\mathbf{X}) - \mathrm{Vol}(0;\mathbf{X})|}{\mathrm{Vol}(0;\mathbf{X})} \tag{2.6.18}$$

using IBMAC and DFIB with $h_s = h, \frac{h}{2}, \frac{h}{4}$, which correspond to triangular meshes with 4,5,6 levels of refinement from the regular icosahedron respectively. For the quasi-static case (Fig. 2.17a), volume errors for DFIB are at least 2 orders of magnitude smaller than those of IBMAC. Further, volume errors for DFIB keep decreasing as the Lagrangian mesh is refined from $h_s = h$ to $\frac{h}{4}$. For the dynamic case (Fig. 2.17b), both methods suffer a significant amount of volume loss arising from the rapid deformation at the beginning of simulation. The volume error of DFIB with $h_s = h$ is similar to those of IBMAC in magnitude, but the volume error of DFIB decreases as the Lagrangian mesh is refined for $h_s = \frac{h}{2}, \frac{h}{4}$. It appears that the behavior of volume error changes in nature from $h_s = h$ to $\frac{h}{2}$, which coincides with the conventional recommendation that the best choice of Lagrangian mesh spacing in the IB method is $h_s = \frac{h}{2}$ in practice. Similar to the two sources of error that contribute to the area loss in 2D, the volume error observed in Fig. 2.17 can also be explained by contribution from



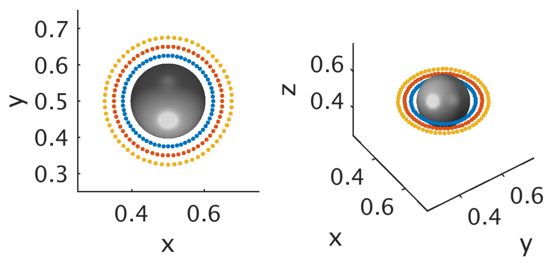
(a) (b) (c)

Fig. 2.15: Triangulation of a spherical surface mesh via refinement of a regular icosahedron. (a) Regular icosahedron (b) Refined mesh after one level of refinement (c) Refined mesh after two levels of refinement.

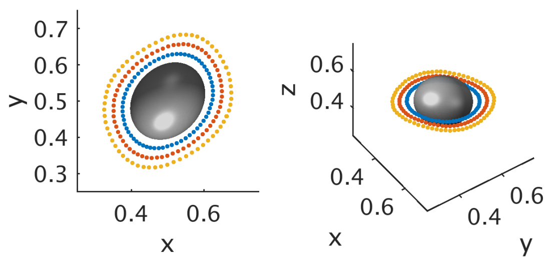
the time-stepping error, and the volume loss due to only moving the vertices (Lagrangian markers) that constitute the triangular mesh. This kind of error in volume conservation decreases as the discretization of the surface is refined, even on a fixed Eulerian grid (as shown in Fig. 2.17). Finally, we remark that the improvement in volume conservation does not seem to be as substantial as the improvement in area conservation in 2D. We suspect that this may be attributed to the larger approximation error in computing the volume using the tetrahedral approximation (after the triangular mesh is deformed), whereas in two dimensions we use a higher-order representation of the interface (cubic splines). Nevertheless, the reduction in volume error from the Lagrangian mesh-refinement experiments indeed confirms that the DFIB method can generally achieve better volume conservation if the immersed boundary is sufficiently resolved ($h_s \leq \frac{h}{2}$).

2.7 Conclusions

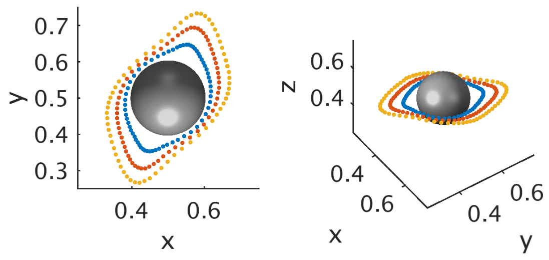
In this chapter, we introduce an IB method with divergence-free velocity interpolation and force spreading. Our IB method makes use of staggered-grid discretization to define an edge-centered discrete vector potential. By interpolating the discrete vector potential



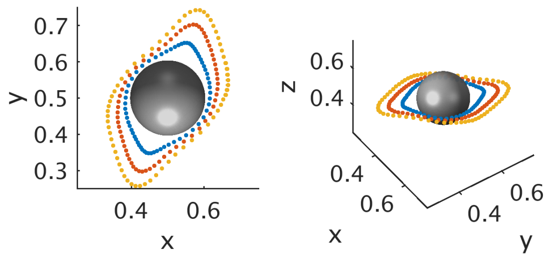
(a) $t = 0$



(b) $t = \frac{1}{32}$

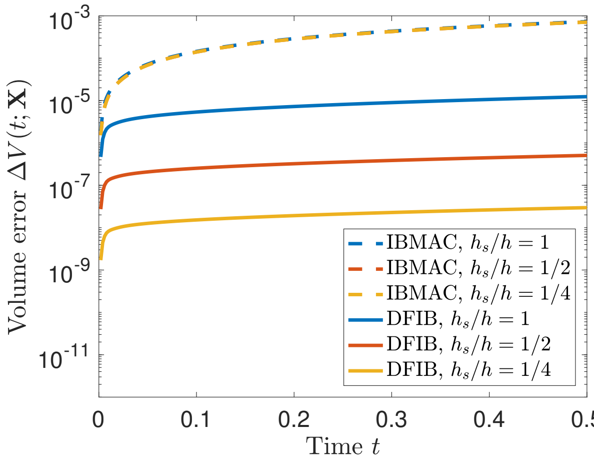


(c) $t = \frac{1}{4}$

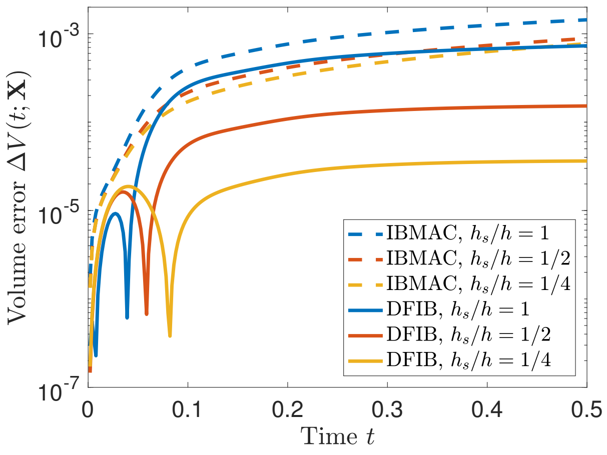


(d) $t = \frac{1}{2}$

Fig. 2.16: Deformation of a 3D elastic membrane immersed in a viscous fluid with initial velocity $\boldsymbol{u}(\boldsymbol{x}, t) = (0,\ \sin(4\pi x),\ 0)$ at $t = 0, \frac{1}{32}, \frac{1}{4}$ and $\frac{1}{2}$. The computation is performed using DFIB with ϕ_{6h}^{new} in the periodic box $\Omega = [0, 1]^3$ with Eulerian meshwidth $h = \frac{1}{128}$. The elastic membrane, initially in spherical configuration with radius $R \approx 0.1$, is discretized by a triangular surface mesh with $M = 10242$ vertices and $P = 20480$ facets so that $h_s = \frac{h}{2}$ in the initial configuration. Colored markers that move passively with the divergence-free interpolated fluid velocity are added for visualizing the fluid flow in the vicinity of the membrane interface.



(a) Quasi-static test



(b) Dynamic test

Fig. 2.17: Normalized volume error $\Delta V(t;\mathbf{X})$ of a 3D elastic membrane using IBMAC and DFIB with $h_s = h, \frac{h}{2}, \frac{h}{4}$, where $h = \frac{1}{128}$. For (a) the quasi-static test, $\Delta V(t,\mathbf{X})$ of DFIB decreases with mesh refinement, while there is no improvement in volume error for IBMAC. For (b) the dynamic test, the volume error in DFIB remains (almost) steady in time for $h_s = \frac{h}{2}, \frac{h}{4}$ as the membrane rests, whereas we see no substantial improvement in volume conservation with mesh refinement for IBMAC, and the volume loss keeps increasing in time. For this set of computations, the $\mathscr{C}^3$ 6-point kernel ϕ_{6h}^{new} is used.

in the conventional IB fashion, we obtain a continuum vector potential whose curl directly yields a continuum Lagrangian velocity field that is exactly divergence-free by default. The corresponding force-spreading operator is constructed to be the adjoint of velocity interpolation so that energy is preserved in the interaction between the fluid and the immersed boundary. Both the new interpolation and spreading schemes require solutions of discrete vector Poisson equations which can be efficiently solved by a variety of algorithms. The transfer of information from the Eulerian grid to the Lagrangian mesh (and vice versa) is performed using $\nabla\delta_h$ on the edge-centered staggered grid $\mathbb{E}$. We have found that volume conservation of DFIB improves with the smoothness of the IB kernel used to construct δ_h, and we have numerically tested that IB kernels that are at least $\mathscr{C}^2$ are good candidate kernels that can be used to construct the regularized delta function in the DFIB method.

We have incorporated the divergence-free interpolation and spreading operators in a second-order time-stepping scheme, and applied it to several benchmark problems in two and three spatial dimensions. First, we have tested that our method achieves second-order convergence in both the fluid velocity and the Lagrangian deformation map for the 2D surface tension problem, which is admittedly a special case, since its continuum solution has a continuous normal derivative of the tangential velocity across the immersed boundary. The highlight of the DFIB is its capability of substantially reducing volume error in the immersed structure as it moves and deforms in the process of fluid-structure interaction. Through numerical simulations of quasi-static and dynamic membranes, we have confirmed that the DFIB method improves volume conservation by several orders of magnitude compared to IBMAC and IBModified. Furthermore, owing to the divergence-free nature of its velocity interpolation, the DFIB method reduces volume error with Lagrangian mesh refinement while keeping the Eulerian grid fixed. A similar refinement study would not yield improved volume conservation when using the conventional IB method. Although the numerical examples considered in this chapter only involve thin elastic structures, we note that the

DFIB method can also be directly applied to model thick elastic structures without any modification to the method, other than representing the thick elastic structure by a curvilinear mesh of Lagrangian points [25, 54, 55, 56]. We also remark that the current version of the DFIB method is accompanied with a single-fluid Navier-Stokes fluid solver. This is not a fundamental limitation in our approach, and as a direction of future research, the DFIB method may be extended to work with variable-viscosity and variable-density fluid solvers [19].

Unlike other improved IB methods that either use non-standard finite-difference operators (IBModified [26]) that complicate the implementation of the fluid solver, or rely on analytically-computed correction terms (IIM [27, 28] or Blob-Projection method [38]) that may not be readily accessible in many applications, the DFIB method is generally applicable, and it is straightforward to implement in both 2D and 3D from an existing IB code that is based on the staggered-grid discretization. Moreover, the additional costs of performing the new interpolation and spreading do not increase the overall complexity of computation and are modest compared to the existing IB methods.

We point out two limitations of our present work. A first limitation of the current version of DFIB method is based on the assumption of periodic boundary conditions. Extending the method to include physical boundary conditions at the boundaries of the computational domain is one possible direction of future work, but there are several challenges to overcome. First, a special treatment of spreading and interpolation is required near the boundaries since the support of the IB kernel can extend outside of the physical domain [15, 22]. Second, with non-periodic BCs, instead of FFTs, the resulting linear system needs to be solved by geometric or algebraic multigrid method to achieve high performance. For unbounded domains, a lattice Green's function technique was recently proposed as an alternative approach [50]. Third, for domains with physical boundaries, the use of projection-based fluid solvers to eliminate pressure introduces splitting errors near the physical boundaries, and instead one ought to solve a coupled velocity-pressure system at every time step [37]. In the DFIB method, it is also nontrivial to specify boundary conditions for the vector potential $\mathbf{a}(\mathbf{x})$ at physical boundaries, which would lead to a Poisson equation with non-periodic BCs. It may be that volume conservation as the structure passes near a boundary requires solving a coupled velocity-potential system. A second limitation of our DFIB method is that the pressure gradient generated by the Lagrangian forces is part of the resulting Eulerian force density because force spreading is also constructed to be discretely divergence-free. However, this may also be an important advantage of our method from the standpoint of accuracy, since it means that jumps in pressure across the interface do not require any explicit representation. We do not yet see an obvious way to extract the pressure from the Eulerian force density in case it is needed for output purposes, or for the purposes of imposing physical boundary conditions involving tractions or avoiding splitting errors near boundaries [37].

2.8 Appendix: Vector identities of discrete differential operators

Suppose $\varphi(\mathbf{x})$ is a scalar grid function defined on $\mathbb{C}$, and $\mathbf{u}(\mathbf{x})$ and $\mathbf{a}(\mathbf{x})$ are vector grid functions defined on $\mathbb{F}$ and $\mathbb{E}$ respectively. The following discrete vector identities are valid on the periodic staggered grid just as in the continuum case,

$$\mathbf{D}^h \times \mathbf{G}^h \varphi = 0, \tag{2.8.1}$$

$$\mathbf{D}^h \cdot (\mathbf{D}^h \times \mathbf{u}) = 0, \tag{2.8.2}$$

$$\mathbf{D}^h \times (\mathbf{D}^h \times \mathbf{u}) = \mathbf{G}^h(\mathbf{D}^h \cdot \mathbf{u}) - \mathbf{L}^h \mathbf{u}, \tag{2.8.3}$$

$$\sum_{\mathbf{x}\in\mathbb{F}} \mathbf{u}(\mathbf{x}) \cdot (\mathbf{G}^h \varphi)(\mathbf{x}) h^3 = -\sum_{\mathbf{x}\in\mathbb{C}} (\mathbf{D}^h \cdot \mathbf{u})(\mathbf{x})\, \varphi(\mathbf{x}) h^3, \tag{2.8.4}$$

$$\sum_{\mathbf{x}\in\mathbb{E}} \mathbf{a}(\mathbf{x}) \cdot (\mathbf{D}^h \times \mathbf{u})(\mathbf{x}) h^3 = \sum_{\mathbf{x}\in\mathbb{F}} (\mathbf{D}^h \times \mathbf{a})(\mathbf{x}) \cdot \mathbf{u}(\mathbf{x}) h^3. \tag{2.8.5}$$

Eqs. (2.8.1) and (2.8.3) are merely discrete analogues of well-known vector identities involving gradient, divergence and curl. These identities can be proved in the same manner as their continuous counterparts. Eqs. (2.8.4) and (2.8.5) can be verified via "summation by parts". Note that Eqs. (2.8.2) and (2.8.3) also hold if we replace $\mathbf{u}$ (which lives on $\mathbb{F}$) by $\mathbf{a}$ (which lives on $\mathbb{E}$).

2.9 Appendix: Existence of discrete vector potential

Lemma 1. *Suppose $\mathbf{D}^h \cdot \mathbf{u} = 0$ and $\mathbf{D}^h \times \mathbf{u} = 0$ for $\mathbf{x} \in \mathbb{F}$, then $\mathbf{u}(\mathbf{x})$ is a constant function on $\mathbb{F}$.*

Proof. To prove this statement, we use Eqs. (2.8.3) to (2.8.5),

$$\begin{aligned}
\sum_{\mathbf{x}\in\mathbb{E}} (\mathbf{D}^h \times \mathbf{u})(\mathbf{x}) \cdot (\mathbf{D}^h \times \mathbf{u})(\mathbf{x}) h^3 &= \sum_{\mathbf{x}\in\mathbb{F}} \mathbf{u}(\mathbf{x}) \cdot (\mathbf{D}^h \times (\mathbf{D}^h \times \mathbf{u})) h^3 \\
&= \sum_{\mathbf{x}\in\mathbb{F}} \mathbf{u}(\mathbf{x}) \cdot \mathbf{G}^h (\mathbf{D}^h \cdot \mathbf{u}) h^3 - \sum_{\mathbf{x}\in\mathbb{F}} \mathbf{u}(\mathbf{x}) \cdot (\mathbf{L}^h \mathbf{u}) h^3 \\
&= -\sum_{\mathbf{x}\in\mathbb{C}} (\mathbf{D}^h \cdot \mathbf{u})^2 h^3 + \sum_{\substack{\mathbf{x}\in\mathbb{E}, i\neq j \\ \mathbf{x}\in\mathbb{C}, i=j}} \left(D_j^h u_i\right)^2 h^3.
\end{aligned}$$

Thus,

$$\sum_{\substack{\mathbf{x}\in\mathbb{E}, i\neq j \\ \mathbf{x}\in\mathbb{C}, i=j}} \left(D_j^h u_i\right)^2 h^3 = \sum_{\mathbf{x}\in\mathbb{E}} \left|(\mathbf{D}^h \times \mathbf{u})\right|^2 h^3 + \sum_{\mathbf{x}\in\mathbb{E}} (\mathbf{D}^h \cdot \mathbf{u})^2 h^3. \tag{2.9.1}$$

Since $\mathbf{D}^h \cdot \mathbf{u} = 0$ and $\mathbf{D}^h \times \mathbf{u} = 0$ by hypothesis, the left-hand side of Eq. (2.9.1) is also zero. But this implies u_i is constant for $i = 1, 2, 3$. □

Lemma 2. *If ψ is a scalar grid function that lives on one of the staggered grids, such that*

$$\sum_{\mathbf{x}} \psi(\mathbf{x}) h^3 = 0, \tag{2.9.2}$$

then there exists a grid function φ such that

$$L^h \varphi = \psi. \tag{2.9.3}$$

Proof. This lemma states the solvability of the discrete Poisson problem Eq. (2.9.3). Since $L^h = \mathbf{D}^h \cdot \mathbf{G}^h$ is symmetric with respect to the inner product on the periodic grid

$$(\varphi, \psi) = \sum_{\mathbf{x}} \varphi(\mathbf{x}) \psi(\mathbf{x}) h^3, \tag{2.9.4}$$

what we have to show is that any ψ satisfying Eq. (2.9.2) is orthogonal to any φ_0 in the null space of $\mathbf{L}^h$. But the null space of $\mathbf{L}^h$ with periodic boundary conditions contains only the constant function, and hence $(\psi, \varphi_0) = 0$ because of Eq. (2.9.2) as required. □

Now we are ready to state the theorem that guarantees the existence of a discrete vector potential $\mathbf{a}(\mathbf{x})$ for $\mathbf{x} \in \mathbb{E}$ given a discretely divergence-free velocity field $\mathbf{u}(\mathbf{x})$ for $\mathbf{x} \in \mathbb{F}$.

Theorem 3. *Suppose $\mathbf{u}(\mathbf{x})$ is a periodic grid function for $\mathbf{x} \in \mathbb{F}$, and $\mathbf{u}(\mathbf{x})$ satisfies*

$$\sum_{\mathbf{x} \in \mathbb{F}} \mathbf{u}(\mathbf{x}) h^3 = 0 \quad \textit{and} \quad \mathbf{D}^h \cdot \mathbf{u} = 0, \tag{2.9.5}$$

then there exists a grid function $\mathbf{a}(\mathbf{x})$ for $\mathbf{x} \in \mathbb{E}$ such that

$$\mathbf{u} = \mathbf{D}^h \times \mathbf{a}. \tag{2.9.6}$$

Proof. We choose $\mathbf{a}(\mathbf{x})$ to be any solution of

$$-\mathbf{L}^h\,\mathbf{a} = \mathbf{D}^h \times \mathbf{u}. \tag{2.9.7}$$

Such an $\mathbf{a}(\mathbf{x})$ exists by Lemma 2, because

$$\begin{aligned}\sum_{\mathbf{x}\in\mathbb{E}} \left(\mathbf{D}^h \times \mathbf{u}\right)_i(\mathbf{x}) &= \epsilon_{ijk} \sum_{\mathbf{x}\in\mathbb{E}} 1 \cdot D_j u_k\, h^3 \\ &= -\epsilon_{ijk} \sum_{\mathbf{x}\in\mathbb{F}} (D_j 1) u_k\, h^3 \\ &= 0.\end{aligned}$$

By applying $\mathbf{D}^h\cdot$ to Eq. (2.9.7) and using the property that $\mathbf{L}^h$ and $\mathbf{D}^h\cdot$ commute, we also have

$$-\mathbf{L}^h(\mathbf{D}^h \cdot \mathbf{a}) = \mathbf{D}^h \cdot (\mathbf{D}^h \times \mathbf{u}) = 0. \tag{2.9.8}$$

Because the null space of $\mathbf{L}^h$ contains only the constant function, it follows that

$$\mathbf{G}^h(\mathbf{D}^h \cdot \mathbf{a}) = 0. \tag{2.9.9}$$

If we use Eq. (2.9.9) and Eq. (2.8.3) for $\mathbf{a}$, we can rewrite Eq. (2.9.7) as

$$\mathbf{D}^h \times (\mathbf{D}^h \times \mathbf{a}) = \mathbf{D}^h \times \mathbf{u}, \tag{2.9.10}$$

or

$$\mathbf{D}^h \times (\mathbf{D}^h \times \mathbf{a} - \mathbf{u}) = 0. \tag{2.9.11}$$

But we also know from Eq. (2.8.2) and the requirement that $\mathbf{D}^h \cdot \mathbf{u} = 0$ that

$$\mathbf{D}^h \cdot (\mathbf{D}^h \times \mathbf{a} - \mathbf{u}) = 0. \tag{2.9.12}$$

From Eqs. (2.9.11) and (2.9.12) and Lemma 1, it follows that

$$\mathbf{D}^h \times \mathbf{a} - \mathbf{u} = \text{constant}. \tag{2.9.13}$$

The constant must be zero, however, since $\mathbf{D}^h \times \mathbf{a}$ has zero sum by "summation by parts", and $\mathbf{u}$ has zero sum by assumption. This completes the proof of the existence of a vector potential satisfying $\mathbf{u} = \mathbf{D}^h \times \mathbf{a}$. $\square$

Chapter 3

A Fluctuating Boundary Integral Method for Brownian Suspensions

We present a fluctuating boundary integral method (FBIM) for overdamped Brownian Dynamics (BD) of two-dimensional periodic suspensions of rigid particles of complex shape immersed in a Stokes fluid. We develop a novel approach for generating Brownian displacements that arise in response to the thermal fluctuations in the fluid. Our approach relies on a first-kind boundary integral formulation of a mobility problem in which a random surface velocity is prescribed on the particle surface, with zero mean and covariance proportional to the Green's function for Stokes flow (Stokeslet). This approach yields an algorithm that scales linearly in the number of particles for both deterministic and stochastic dynamics, handles particles of complex shape, achieves high order of accuracy, and can be generalized to three dimensions and other boundary conditions. We show that Brownian displacements generated by our method obey the discrete fluctuation-dissipation balance relation (DFDB). Based on a recently-developed Positively Split Ewald method [57], near-field contributions to the Brownian displacements are efficiently approximated by iterative methods in real space, while far-field contributions are rapidly generated by fast Fourier-space methods based on fluctuating hydrodynamics. FBIM provides the key ingredient for time integration of the overdamped Langevin equations for Brownian suspensions of rigid particles. We demonstrate that FBIM obeys DFDB by performing equilibrium BD simulations of suspensions of starfish-shaped bodies using a random finite difference temporal integrator.

3.1 Introduction

Complex fluids containing colloidal particles are ubiquitous in science and industrial applications. Colloidal particles span length scales from several nanometers, such as magnetic nano-propellers [58] and molecular motors [59, 60], to a few microns, such as self-phoretic Janus particles [61] and motile microorganisms [62]. In the last decade, increasing attention has been given to the emerging field of *active* colloidal suspensions [63, 64, 65, 66, 67], in which particles move autonomously or in response to external forces.

Despite the advances in the theory and experimental design of passive and active colloids, developing accurate and efficient computational methods that are capable of simulating tens or hundreds of thousands of particles, as well as handling particles of complex shape, still remains a formidable challenge. The main purpose of this chapter is to present a novel computational framework for Brownian suspensions in two-dimensional periodic domains, which scales linearly in the number of particles for both deterministic and stochastic dynamics, handles particles of complex shape, and achieves high order of accuracy.

The two key aspects that need to be included in a computational method for colloidal suspensions are the long-ranged hydrodynamic interactions (HIs) and the correlated Brownian motion of the particles. In the absence of active and Brownian motion, describing the dynamics of Stokesian suspensions requires the accurate solution of mobility problems [68, 1], i.e., computing the linear and angular velocities of the particles in response to applied (external) forces and torques. This defines the action of a *body mobility matrix* $\boldsymbol{\mathcal{N}}$, which encodes all of the hydrodynamic interactions among the particles. In addition, fluctuation-dissipation balance requires the Brownian (stochastic) displacements of particles to have zero mean and covariance proportional to $\boldsymbol{\mathcal{N}}$, i.e., hydrodynamic interactions are synonymous with hydrodynamic correlations among the particles' random velocities. Computing the Brownian displacements over a time step requires generating Gaussian random variables whose covariance is $\boldsymbol{\mathcal{N}}$. In other words, computing Brownian increments necessitates the computation of the action of a matrix $\boldsymbol{\mathcal{N}}^{\frac{1}{2}}$, the "square root" of $\boldsymbol{\mathcal{N}}$ (not unique), on a vector of white noise processes. Developing numerical methods for performing BD simulations for many-particle systems requires applying the action of $\boldsymbol{\mathcal{N}}$ and $\boldsymbol{\mathcal{N}}^{\frac{1}{2}}$ efficiently. In this work we develop a new linear-scaling method for simulating Brownian suspensions of particles of complex shape in two-dimensional periodic domains using a boundary integral formulation. Because of its close connection to fluctuating hydrodynamics, we refer to our method as the *Fluctuating Boundary Integral Method* (FBIM).

For passive suspensions of spherical particles, the methods of Brownian [69, 70, 71, 72, 73] and Stokesian Dynamics (SD) [74, 75, 76] have dominated the chemical engineering community. These methods are tailored to sphere suspensions and utilize a multipole hierarchy truncated at either the monopole (BD) or dipole (SD) level in order to capture the far-field behavior of the hydrodynamic interactions. Modern fast algorithms can apply the action of the truncated mobility matrix with linear-scaling by using the Fast Multipole Method (FMM) for an unbounded domain [73], and using Ewald-like methods for periodic [77, 76] and confined domains [71]. The Brownian (stochastic) displacements are typically generated iteratively by a Chebyshev polynomial approximation method [78], or by the Lanczos algorithm for application of the matrix square root [79]. However, since the hydrodynamic interactions among particles decay slowly like the inverse of distance in three dimensions and diverge logarithmically in two dimensions, the condition number of the mobility matrix grows as the number of particles increases (keeping the packing fraction fixed, see [57, Fig. 1]). Therefore, the overall computational scaling of directly applying iterative methods on $\boldsymbol{\mathcal{N}}$ to generate the action of $\boldsymbol{\mathcal{N}}^{\frac{1}{2}}$ is only super-linear in general.

The fluctuating Lattice Boltzmann (FLB) method has been used for Brownian suspensions for some time [80, 81]. This is an explicit solvent method which includes fluid inertia and thus operates at finite Schmidt number instead of in the overdamped limit we are interested in; furthermore, FLB relies on artificial fluid compressibility to avoid solving Poisson problems for the pressure. While the cost of each time step is linear in the number of particles (more precisely, the number of fluid cells) N, in three dimensions $\mathcal{O}\left(N^{2/3}\right)$ time steps are required for vorticity to diffuse throughout the system volume, leading to superlinear $\mathcal{O}\left(N^{5/3}\right)$ overall complexity [57].

As an alternative, methods such as the Fluctuating Immersed Boundary method (FIB) [82] and the Fluctuating Force Coupling Method (FFCM) [83, 84] utilize fluctuating hydrodynamics to generate the Brownian increments in linear time by solving the fluctuating steady Stokes equation on a grid. This ensures that the computational cost of Brownian simulation is only marginally larger than the cost of deterministic simulations, in stark contrast to traditional BD approaches. The FIB/FCM approach to generating the Brownian displacements is further improved in the recently-developed Positively Split Ewald (PSE) method [57]. In PSE, the Rotne-Prager-Yamakawa (RPY) tensor [85] is used to capture the long-ranged hydrodynamic interactions, and its action is computed with spectral accuracy by extending the Spectral Ewald [77] method for the RPY tensor. The key idea in PSE is to use the Hasimoto splitting [86] to decompose the RPY tensor into near-field (short-ranged) and far-field (long-ranged) contributions, which guarantees that *both* contributions are independently symmetric and positive-definite (SPD). This makes it possible to apply a Lanczos algorithm [79] to generate the near-field contribution with only a small ($O(1)$) number of iterations, while the far-field contribution is computed by fast Fourier-space methods based on the fluctuating hydrodynamics using only a few FFTs. Later in Sec. 3.3.3, we will apply the same SPD splitting idea to the Green's function of steady Stokes flow to achieve linear-scaling in the FBIM.

Essentially all commonly-used methods for suspension flows are limited to spherical particles (with some extensions to spheroids [87]), and generalization to include particles of complex shape is generally difficult. Further, these methods employ an *uncontrolled* truncation of a multipole expansion hierarchy and therefore become inaccurate when particles get close to one another as in dense suspensions.

For deterministic Stokes problems, the Boundary Integral Method (BIM) [1] is very well-developed [2, 3, 4] and allows one to handle complex particle shapes and achieve *controlled accuracy* even for dense suspensions [5, 6]. In the boundary integral framework, the steady Stokes equation are reformulated as an integral equation of unknown densities that are defined on the boundary, using a first-kind (single layer densities) or second-kind (double layer densities) formulation, or a mixture of both. Suspended particles of complex geometry can be directly discretized by a surface mesh, and by a suitable choice of surface quadrature higher-order, or even spectral accuracy, can be achieved. The key difficulty is handling the singularity of the Green's functions appearing in the boundary integral formulation. One possibility is to regularize the singularity, as done in the method of regularized Stokeslets [88] and the recently-developed linear-scaling rigid multiblob method [23], both of which rely on a regularized first-kind formulation. Regularization, however, comes at a drastic loss of accuracy, and to resolve near-field hydrodynamic interactions accurately one must make use of high-order accurate *singular quadratures* for the singular or near-singular kernel in the first- and second-kind integral operators, such as quadrature-by-expansion (QBX) [89, 90].

Discretizing the boundary integral equation typically leads to a dense linear system, and fast algorithms for performing the dense matrix-vector product are required to achieve linear-scaling, such as the Fast Multipole Method (FMM) [2, 4], and Spectral Ewald methods [5, 6]. Much of the state-of-the-art BIM work on Stokes mobility problems [5, 6, 4, 91] uses (completed) *second-kind* (double layer) formulations, and we will not review it in detail here since we will rely on a first-kind formulation. The current development of BIM for particle suspensions is limited to the deterministic case only. In this work, we combine singular quadrature (namely, Alpert quadrature in two dimensions) and the idea of SPD splitting from PSE method to develop a linear-scaling method for generating the Brownian displacements based on a *first-kind* boundary integral formulation.

The outline of this chapter is as follows. First, in the continuum formulation (Sec. 3.2), we show that the action of $\boldsymbol{\mathcal{N}}$ and $\boldsymbol{\mathcal{N}}^{\frac{1}{2}}$ can be formulated equivalently as the solution of a Stokes boundary value problem, herein referred to as the *Stochastic Stokes Boundary Value Problem* (SSBVP). The key idea is that, instead of adding a stochastic stress to the fluid equations as done in fluctuating hydrodynamics, we can prescribe a random surface velocity (distribution) on the particle surface that has zero mean and covariance proportional to the (singular) periodic Green's function of the Stokes flow (the periodic *Stokeslet*). Reformulating the SSBVP as a first-kind boundary integral equation reveals that the random surface velocity has covariance proportional to the single-layer integral operator, which suggests that one ought to handle the singularity of the covariance using the same machinery used to handle the singularity of the Green's function in the first-kind BIM. This allows us to develop a numerical method that satisfies discrete fluctuation-dissipation balance (DFDB) to within solver and roundoff errors.

In our two dimensional discrete formulation (Sec. 3.3), the first-kind integral equation is discretized using Alpert quadrature [92]. The resulting dense *saddle-point* linear system of equations contains on the right hand side a random surface velocity whose covariance is proportional to $\mathbf{M}$, the matrix discretizing the single-layer operator. We solve the saddle point problem iteratively by the generalized minimal residual method (GMRES), using fast (near) *linear-scaling* methods to apply the action of $\mathbf{M}$ and $\mathbf{M}^{1/2}$ on vectors. To deal with the inherent ill-conditioning of the linear system due to the first-kind formulation, we use a simple block-diagonal preconditioning technique [23, 6] for both GMRES and the Lanczos method. Our fast method heavily relies on the set of ideas developed for the PSE method [57], which in turn relies heavily on ideas developed for the SE method [77]. Specifically, we use Ewald decomposition to split the action of $\mathbf{M}$ and $\mathbf{M}^{1/2}$ into a far-field (long-ranged) part that can be computed efficiently in the wave space by FFTs, and a near-field (short-ranged) part that can be computed directly in the real space using a preconditioned Krylov (Lanczos) method [93]. The key requirement is to guarantee that *both* the far-field and near-field contributions are *symmetric positive definite* (SPD), so that taking the square root of each part is well-defined. One choice of splitting that ensures the SPD property of both parts [57] is the Hasimoto splitting [86].

We apply the FBIM to several benchmark problems in two dimensions (Sec. 3.4), and assess the effectiveness of the method by its accuracy and convergence, robustness, and scalability for suspensions of many rigid disks. We also couple the FBIM with stochastic temporal integrators based on the idea of random finite differences [82, 94, 95] to perform BD simulations with starfish-shaped particles, and confirm that DFBD is obtained for sufficiently small time step sizes by comparing the numerical equilibrium distribution to the correct Gibbs-Boltzmann distribution.

3.2 Continuum formulation

This section presents the continuum formulation of the equations of motion for Brownian suspensions. We consider a suspension of N_b rigid Brownian particles of complex shape immersed in a viscous incompressible fluid with constant density ρ, viscosity η, and temperature T in a domain $\mathcal{V}$ with periodic boundary conditions. Each particle or body, indexed by β, is described by the position of a chosen "tracking point", denoted by $\boldsymbol{q}_\beta$, and its rotation relative to a chosen reference configuration, denoted by $\boldsymbol{\theta}_\beta$. In this chapter, we restrict the formulation and implementation to two spatial dimensions, and $\theta_\beta \in \mathbb{R}$ is simply the angle of rotation. The generalization to three dimensional systems is intellectually straightforward but technically complicated by the fact that orientation in three dimensions cannot be represented by a vector in $\mathbb{R}^3$; instead, one can use normalized quaternions [94]. We denote the particle surface by Γ_β which encloses an interior domain denoted by D_β. The fluid domain exterior to the particles is defined by $E = \mathcal{V} \backslash \left\{ \cup_{\beta=1}^{N_b} \overline{D}_\beta \right\}$, where $\overline{D}_\beta = D_\beta \cup \Gamma_\beta$.

A number of prior works [96, 97, 98, 99] have arrived at a fluctuating hydrodynamics formulation of the equations of motion when fluid and particle inertia (and perhaps compressibility [100]) is accounted for [1]. The fluid velocity $\boldsymbol{v}(\boldsymbol{r}, t)$ and the pressure π follows the time-dependent fluctuating Stokes equations for all $\boldsymbol{r} \in E$,

$$\rho\, \partial_t \boldsymbol{v} + \nabla \pi = \eta \nabla^2 \boldsymbol{v} + \sqrt{2\eta k_B T}\ \boldsymbol{\nabla} \cdot \boldsymbol{\mathcal{Z}}, \tag{3.2.1a}$$

$$\boldsymbol{\nabla} \cdot \boldsymbol{v} = 0, \tag{3.2.1b}$$

where k_B is Boltzmann's constant and $\boldsymbol{\mathcal{Z}}(\boldsymbol{r}, t)$ is a random Gaussian tensor field whose components are white in space and time with mean zero and covariance

$$\langle \boldsymbol{\mathcal{Z}}_{ij}(\boldsymbol{r}, t)\ \boldsymbol{\mathcal{Z}}_{kl}(\boldsymbol{r}', t) \rangle = (\delta_{ik}\delta_{jl} + \delta_{il}\delta_{jk})\delta(\boldsymbol{r} - \boldsymbol{r}')\delta(t - t'). \tag{3.2.2}$$

On the surface of Γ_β, we assume that *no-slip* boundary condition (BC) holds,

$$\boldsymbol{v}(\boldsymbol{x}, t) = \boldsymbol{u}_\beta + \boldsymbol{\omega}_\beta \times (\boldsymbol{x} - \boldsymbol{q}_\beta), \quad \forall \boldsymbol{x} \in \Gamma_\beta, \tag{3.2.3}$$

[1] Note that Hauge and Martin-Lof [99] explain that there is some ambiguity in whether the stochastic traction is taken to be zero or nonzero on the particle surface; this choice does not, however, affect the resulting equations of motion for the bodies. We consider the formulation given here to be the more physically meaningful and follow Hinch [97], see in particular Section 3 of the comprehensive work of Roux [98].

where $\boldsymbol{u}_\beta$ and $\boldsymbol{\omega}_\beta$ are the translational and rotational velocities of the particle. Each particle is also subject to applied force $\boldsymbol{f}_\beta$ and torque $\boldsymbol{\tau}_\beta$, which are related to the fluid stress tensor $\boldsymbol{\sigma}$ and stochastic stress tensor $\boldsymbol{\sigma}^{(s)}$ by

$$\begin{aligned} m_\beta \frac{d\boldsymbol{u}_\beta}{dt} &= \boldsymbol{f}_\beta - \int_{\Gamma_\beta} (\boldsymbol{\lambda}_\beta + \boldsymbol{\lambda}_\beta^{(s)})(\boldsymbol{x})\, \mathrm{d}S_{\boldsymbol{x}}, \\ \mathbf{I}_\beta \cdot \frac{d\boldsymbol{\theta}_\beta}{dt} &= \boldsymbol{\tau}_\beta - \int_{\Gamma_\beta} (\boldsymbol{x} - \boldsymbol{x}_\beta) \times (\boldsymbol{\lambda}_\beta + \boldsymbol{\lambda}_\beta^{(s)})(\boldsymbol{x})\, \mathrm{d}S_{\boldsymbol{x}}, \end{aligned} \tag{3.2.4}$$

where m_β and $\mathbf{I}_\beta$ are mass and moment of inertia tensor of body β. Here, $\boldsymbol{\lambda}_\beta(\boldsymbol{x}) = (\boldsymbol{\sigma} \cdot \boldsymbol{n}_\beta)(\boldsymbol{x})$ and $\boldsymbol{\lambda}_\beta^{(s)}(\boldsymbol{x}) = \left(\boldsymbol{\sigma}^{(s)} \cdot \boldsymbol{n}_\beta\right)(\boldsymbol{x})$ are the normal components of the stress tensors on the outside of the surface of the body,

$$\begin{aligned} \boldsymbol{\sigma} &= -\pi \boldsymbol{I} + \eta(\nabla \boldsymbol{v} + \nabla^\top \boldsymbol{v}), \\ \boldsymbol{\sigma}^{(s)} &= \sqrt{2\eta k_B T}\boldsymbol{\mathcal{Z}}, \end{aligned} \tag{3.2.5}$$

and $\boldsymbol{n}_\beta$ is the unit outward normal vector of Γ_β pointing into the fluid domain.

In the overdamped (large Schmidt number) limit where the fluid velocity is eliminated as a fast variable through an adiabatic mode elimination procedure [101], the diffusive motion of the rigid bodies can be described by the stochastic differential equation (SDE) of *Brownian Dynamics* (BD) [2] [98]:

$$\frac{d\boldsymbol{Q}}{dt} = \boldsymbol{\mathcal{N}}\boldsymbol{F} + \sqrt{2k_B T}\boldsymbol{\mathcal{N}}^{\frac{1}{2}}\boldsymbol{\mathcal{W}} + (k_B T)(\partial_{\boldsymbol{Q}} \cdot \boldsymbol{\mathcal{N}}), \tag{3.2.6}$$

where $\boldsymbol{Q}_\beta = \left\{\boldsymbol{q}_\beta, \boldsymbol{\theta}_\beta\right\}$ and $\boldsymbol{Q} = \left\{\boldsymbol{Q}_\beta\right\}_{\beta=1}^{N_b}$ is a composite vector that collects the positions and orientations of particles (in two dimensions, $\boldsymbol{Q}_\beta \in \mathbb{R}^3$). The first term on the right-hand-side of Eq. (3.2.6) is the deterministic motion of rigid bodies, where $\boldsymbol{\mathcal{N}}(\boldsymbol{Q}) \succeq \mathbf{0}$ is the symmetric positive semidefinite (SPD) *body mobility* matrix that converts applied forces and torques

[2]We use the differential notation of SDEs common in the physics literature.

$\boldsymbol{F}(\boldsymbol{Q}) = \left\{\boldsymbol{f}_\beta(\boldsymbol{Q}), \boldsymbol{\tau}_\beta(\boldsymbol{Q})\right\}_{\beta=1}^{N_b}$ to rigid body motion, and can be obtained by solving the standard mobility problem for rigid body motion,

$$\begin{cases} -\boldsymbol{\nabla}\cdot\boldsymbol{\sigma} = \nabla\pi - \eta\nabla^2\boldsymbol{v} = \boldsymbol{0}, \\ \boldsymbol{\nabla}\cdot\boldsymbol{v} = 0, \\ \boldsymbol{v}(\boldsymbol{x}) = \boldsymbol{u}_\beta + \boldsymbol{\omega}_\beta \times (\boldsymbol{x} - \boldsymbol{q}_\beta), \quad \forall\, \boldsymbol{x} \in \Gamma_\beta, \\ \int_{\Gamma_\beta} \boldsymbol{\lambda}_\beta(\boldsymbol{x})\,\mathrm{d}S_{\boldsymbol{x}} = \boldsymbol{f}_\beta \quad \text{and} \quad \int_{\Gamma_\beta} (\boldsymbol{x} - \boldsymbol{x}_c) \times \boldsymbol{\lambda}_\beta(\boldsymbol{x})\,\mathrm{d}S_{\boldsymbol{x}} = \boldsymbol{\tau}_\beta. \end{cases} \tag{3.2.7}$$

The random Brownian motion of the particles involves computing the "square root" of the body mobility matrix, denoted by $\boldsymbol{\mathcal{N}}^{\frac{1}{2}}(\boldsymbol{Q})$ acting on a vector of independent white noise processes $\boldsymbol{\mathcal{W}}(t)$. In order for fluctuation-dissipation balance to hold, the matrix $\boldsymbol{\mathcal{N}}^{\frac{1}{2}}$ must satisfy $\boldsymbol{\mathcal{N}}^{\frac{1}{2}}\left(\boldsymbol{\mathcal{N}}^{\frac{1}{2}}\right)^\top = \boldsymbol{\mathcal{N}}$, and does not necessarily have to be square. The last term on the right-hand-side of Eq. (3.2.6) is the stochastic drift term due to the Ito interpretation of the SDEs, where the divergence operator $(\partial_{\boldsymbol{x}}\cdot)$ for a matrix-valued function $\boldsymbol{A}(\boldsymbol{x})$ is defined by $(\partial_{\boldsymbol{x}}\cdot\boldsymbol{A})_i = \sum_j \partial A_{ij}/\partial x_j$.

Developing numerical schemes to integrate Eq. (3.2.6) has two main challenges. The first challenge is that, at every time step, one needs to generate the random velocity

$$\boldsymbol{U} = \{\boldsymbol{u}_\beta, \boldsymbol{\omega}_\beta\}_{\beta=1}^{N_\beta} = \bar{\boldsymbol{U}} + \tilde{\boldsymbol{U}} = \boldsymbol{\mathcal{N}}\boldsymbol{F} + \boldsymbol{\mathcal{N}}^{\frac{1}{2}}\boldsymbol{W}, \tag{3.2.8}$$

where $\bar{\boldsymbol{U}}$ is the deterministic particle velocity due to applied forces and torques, $\tilde{\boldsymbol{U}}$ is the random velocity due to the stochastic stress tensor, and $\boldsymbol{W}$ is a vector of independent and identically distributed (i.i.d.) Gaussian random variables with mean zero and covariance

$$\langle W_i\, W_j\rangle = \frac{2k_BT}{\Delta t}\delta_{ij}, \tag{3.2.9}$$

where Δt is the time step size. More precisely, our task is to efficiently and accurately apply the action of $\boldsymbol{\mathcal{N}}$ and $\boldsymbol{\mathcal{N}}^{\frac{1}{2}}$. The second challenge is to compute or approximate the stochastic drift term $(k_B T)(\partial_{\boldsymbol{Q}} \cdot \boldsymbol{\mathcal{N}})$, which is conventionally handled by developing specialized stochastic temporal integrators [102, 103, 82, 104, 84]. The main focus of this chapter is to tackle the first challenge by developing schemes that generate the action of $\boldsymbol{\mathcal{N}}$ and $\boldsymbol{\mathcal{N}}^{\frac{1}{2}}$ based on a boundary integral formulation.

3.2.1 Boundary value problem formulation

For simplicity, we now consider only a single particle Γ described by $\{\boldsymbol{q}, \boldsymbol{\theta}\}$ immersed in a periodic domain $\mathcal{V} = [0, L]^2$, and we drop the subscript β. The generalization to account for many-body interaction is straightforward. In this section, we show that the random velocity given by Eq. (3.2.8) can be obtained by solving the *Stochastic Stokes Boundary Value Problem* (SSBVP):

$$\begin{cases} -\boldsymbol{\nabla} \cdot \boldsymbol{\sigma} = \nabla \pi - \eta \nabla^2 \boldsymbol{v} = 0, \quad \boldsymbol{r} \in \mathcal{V} \backslash \overline{D}, \\ \boldsymbol{\nabla} \cdot \boldsymbol{v} = 0, \\ \boldsymbol{v}(\boldsymbol{x}) = \boldsymbol{u} + \boldsymbol{\omega} \times (\boldsymbol{x} - \boldsymbol{q}) - \breve{\boldsymbol{v}}(\boldsymbol{x}), \quad \boldsymbol{x} \in \Gamma, \\ \int_\Gamma \boldsymbol{\lambda}(\boldsymbol{x})\, \mathrm{d}S_{\boldsymbol{x}} = \boldsymbol{f} \quad \text{and} \quad \int_\Gamma (\boldsymbol{x} - \boldsymbol{q}) \times \boldsymbol{\lambda}(\boldsymbol{x})\, \mathrm{d}S_{\boldsymbol{x}} = \boldsymbol{\tau}, \end{cases} \tag{3.2.10}$$

where $\breve{\boldsymbol{v}}(\boldsymbol{x})$ is a random surface velocity prescribed on the particle, that has zero mean and covariance

$$\langle \breve{\boldsymbol{v}}(\boldsymbol{x})\, \breve{\boldsymbol{v}}(\boldsymbol{y}) \rangle = \frac{2 k_B T}{\Delta t} \mathbb{G}(\boldsymbol{x}, \boldsymbol{y}), \quad \text{for all } (\boldsymbol{x} \neq \boldsymbol{y}) \in \Gamma, \tag{3.2.11}$$

Here $\mathbb{G}(\boldsymbol{x}, \boldsymbol{y})$ is the Green's function for steady Stokes flow with viscosity η, and includes the specified boundary conditions (periodic BCs in our case). For $\boldsymbol{x} = \boldsymbol{y}$, Eq. (3.2.11) is not well-defined since $\mathbb{G}$ is singular, which implies that $\breve{\boldsymbol{v}}$ is a distribution and not a function; a more precise definition is given later in Eq. (3.2.28).

By linearity of Stokes flow, the solution of Eq. (3.2.10) is the superposition of

$$\boldsymbol{v} = \bar{\boldsymbol{v}} + \tilde{\boldsymbol{v}}\ ,\ \boldsymbol{\sigma} = \bar{\boldsymbol{\sigma}} + \tilde{\boldsymbol{\sigma}}\ ,\ \boldsymbol{U} = \bar{\boldsymbol{U}} + \tilde{\boldsymbol{U}}, \tag{3.2.12}$$

where $\bar{\boldsymbol{U}} = \{\bar{\boldsymbol{u}}, \bar{\boldsymbol{\omega}}\}$ and $\tilde{\boldsymbol{U}} = \{\tilde{\boldsymbol{u}}, \tilde{\boldsymbol{\omega}}\}$ with $\{\bar{\boldsymbol{v}}, \bar{\boldsymbol{\sigma}}, \bar{\boldsymbol{u}}, \bar{\boldsymbol{\omega}}\}$ satisfying a Stokes BVP *without* random surface velocity,

$$\begin{cases} -\boldsymbol{\nabla}\cdot\bar{\boldsymbol{\sigma}} = \nabla\bar{\pi} - \eta\nabla^2\bar{\boldsymbol{v}} = \boldsymbol{0}, \\ \boldsymbol{\nabla}\cdot\bar{\boldsymbol{v}} = 0, \\ \bar{\boldsymbol{v}}(\boldsymbol{x}) = \bar{\boldsymbol{u}} + \bar{\boldsymbol{\omega}} \times (\boldsymbol{x} - \boldsymbol{q}), \quad \boldsymbol{x} \in \Gamma \\ \displaystyle\int_\Gamma \bar{\boldsymbol{\lambda}}(\boldsymbol{x})\,\mathrm{d}S_{\boldsymbol{x}} = \boldsymbol{f} \quad \text{and} \quad \int_\Gamma (\boldsymbol{x} - \boldsymbol{q}) \times \bar{\boldsymbol{\lambda}}(\boldsymbol{x})\,\mathrm{d}S_{\boldsymbol{x}} = \boldsymbol{\tau}, \end{cases} \tag{3.2.13}$$

and $\{\tilde{\boldsymbol{v}}, \tilde{\boldsymbol{\sigma}}, \tilde{\boldsymbol{u}}, \tilde{\boldsymbol{\omega}}\}$ satisfying a force- and torque-free Stokes BVP *with* a random surface velocity $\breve{\boldsymbol{v}}$ with zero mean and covariance (3.2.11),

$$\begin{cases} -\boldsymbol{\nabla}\cdot\tilde{\boldsymbol{\sigma}} = \nabla\tilde{\pi} - \eta\nabla^2\tilde{\boldsymbol{v}} = \boldsymbol{0}, \\ \boldsymbol{\nabla}\cdot\tilde{\boldsymbol{v}} = 0, \\ \tilde{\boldsymbol{v}}(\boldsymbol{x}) = \tilde{\boldsymbol{u}} + \tilde{\boldsymbol{\omega}} \times (\boldsymbol{x} - \boldsymbol{q}) - \breve{\boldsymbol{v}}(\boldsymbol{x}), \quad \boldsymbol{x} \in \Gamma, \\ \displaystyle\int_\Gamma \tilde{\boldsymbol{\lambda}}(\boldsymbol{x})\,\mathrm{d}S_{\boldsymbol{x}} = \boldsymbol{0} \quad \text{and} \quad \int_\Gamma (\boldsymbol{x} - \boldsymbol{q}) \times \tilde{\boldsymbol{\lambda}}(\boldsymbol{x})\,\mathrm{d}S_{\boldsymbol{x}} = \boldsymbol{0}. \end{cases} \tag{3.2.14}$$

The BVP given by Eq. (3.2.13) is the standard mobility problem for rigid body motion that solves for the deterministic part of the particle velocity $\bar{\boldsymbol{U}} = \{\bar{\boldsymbol{u}}, \bar{\boldsymbol{\omega}}\} = \boldsymbol{\mathcal{N}}\boldsymbol{F}$. The BVP given by Eq. (3.2.14) generates its stochastic part $\tilde{\boldsymbol{U}} = \{\tilde{\boldsymbol{u}}, \tilde{\boldsymbol{\omega}}\}$.

To show that the random particle velocity $\tilde{\boldsymbol{U}}$ determined by the mobility problem (3.2.14) indeed obeys the fluctuation-dissipation balance,

$$\langle \tilde{\boldsymbol{U}} \tilde{\boldsymbol{U}}^\top \rangle = \frac{2k_BT}{\Delta t} \boldsymbol{\mathcal{N}}, \tag{3.2.15}$$

we will invoke the *Lorentz Reciprocal Theorem* (LRT) [1, see Eq. (1.4.5)],

$$\int_\Gamma \boldsymbol{u} \cdot \boldsymbol{\lambda}' \, \mathrm{d}S = \int_\Gamma \boldsymbol{u}' \cdot \boldsymbol{\lambda} \, \mathrm{d}S, \tag{3.2.16}$$

where $\{\boldsymbol{u}, \boldsymbol{\lambda}\}$ and $\{\boldsymbol{u}', \boldsymbol{\lambda}'\}$ are two arbitrary velocity-traction pairs corresponding to solutions of the homogeneous Stokes equations. To apply the LRT, we substitute $\{\boldsymbol{u}, \boldsymbol{\lambda}\}$ by $\{\tilde{\boldsymbol{v}}, \tilde{\boldsymbol{\lambda}}\}$ from Eq. (3.2.14), and $\{\boldsymbol{u}', \boldsymbol{\lambda}'\}$ by $\{\boldsymbol{v}^{(i)}, \boldsymbol{\lambda}^{(i)}\}$, where $\boldsymbol{v}^{(i)}$ and $\boldsymbol{\lambda}^{(i)}$ are the velocity and traction of the standard mobility problem (3.2.13) with applied force and torque $\boldsymbol{F} = \{\boldsymbol{f}, \boldsymbol{\tau}\} = \boldsymbol{e}^{(i)}$, where $\boldsymbol{e}^{(i)} \in \mathbb{R}^3$ is the i^{th} column of the identity matrix. By using the associated BCs in Eq. (3.2.13) and Eq. (3.2.14) to express $\tilde{\boldsymbol{v}}$ and $\boldsymbol{v}^{(i)}$ on Γ, and then making use of the force and torque balance conditions for $\tilde{\boldsymbol{\lambda}}$ and $\boldsymbol{\lambda}^{(i)}$, we can rewrite Eq. (3.2.16) as

$$\tilde{\boldsymbol{U}} \cdot \boldsymbol{e}^{(i)} = \tilde{U}_i = \int_\Gamma \breve{\boldsymbol{v}}(\boldsymbol{x}) \cdot \boldsymbol{\lambda}^{(i)}(\boldsymbol{x}) \, \mathrm{d}S_{\boldsymbol{x}}. \tag{3.2.17}$$

This allows to express the covariance of $\tilde{\boldsymbol{U}}$ as:

$$\begin{aligned} \langle \tilde{U}_i \tilde{U}_j \rangle &= \int_\Gamma \int_\Gamma \boldsymbol{\lambda}^{(j)}(\boldsymbol{x}) \cdot \langle \breve{\boldsymbol{v}}(\boldsymbol{x}) \breve{\boldsymbol{v}}(\boldsymbol{y}) \rangle \cdot \boldsymbol{\lambda}^{(i)}(\boldsymbol{y}) \, \mathrm{d}S_{\boldsymbol{y}} \, \mathrm{d}S_{\boldsymbol{x}} \\ &= \frac{2k_BT}{\Delta t} \int_\Gamma \int_\Gamma \boldsymbol{\lambda}^{(j)}(\boldsymbol{x}) \cdot \mathbb{G}(\boldsymbol{x}, \boldsymbol{y}) \cdot \boldsymbol{\lambda}^{(i)}(\boldsymbol{y}) \, \mathrm{d}S_{\boldsymbol{y}} \, \mathrm{d}S_{\boldsymbol{x}}. \end{aligned} \tag{3.2.18}$$

It can be shown (see 3.6) that the last integral in Eq. (3.2.18) is equal to the $(i, j)^{th}$ element of the body mobility matrix $\boldsymbol{\mathcal{N}}$, giving the desired result,

$$\langle \tilde{U}_i \tilde{U}_j \rangle = \frac{2k_BT}{\Delta t} \mathcal{N}_{ij}. \tag{3.2.19}$$

This shows that solving the SSBVP (3.2.10) gives the desired deterministic and stochastic particle velocity

$$\boldsymbol{U} = \bar{\boldsymbol{U}} + \tilde{\boldsymbol{U}} = \boldsymbol{\mathcal{N}}\boldsymbol{F} + \boldsymbol{\mathcal{N}}^{\frac{1}{2}}\boldsymbol{W}. \tag{3.2.20}$$

While the LRT has been used in the past to analyze the nonhomogeneous Stokes BVP involving the fluctuating stress [96], here we employ it to establish that the homogeneous Stokes BVP given by Eq. (3.2.14) and Eq. (3.2.11) yields the correct statistics for the rigid body motion of the immersed particles. The removal of the fluctuating stress driving the surrounding fluid allows for the eventual application of boundary integral techniques to solve the SSBVP.

3.2.2 First-kind integral formulation

For rigid particles moving in a Stokes fluid, we observe that the details of what happens inside the particle do not actually matter for its motion and its hydrodynamic interactions with other particles or boundaries. Therefore, it is possible to extend the fluid to the entire domain so that the fluid inside the body moves with a velocity that is continuous across the boundary of the body. Once we extend the fluid to the interior of bodies, we may write down an alternative formulation of the SSBVP (3.2.10) as a *first-kind* boundary integral equation [1],

$$\boldsymbol{v}(\boldsymbol{x} \in \Gamma) = \boldsymbol{u} + \boldsymbol{\omega} \times (\boldsymbol{x} - \boldsymbol{q}) - \breve{\boldsymbol{v}}(\boldsymbol{x}) = \int_\Gamma \mathbb{G}(\boldsymbol{x}, \boldsymbol{y})\, \boldsymbol{\psi}(\boldsymbol{y})\, \mathrm{d}S_{\boldsymbol{y}}, \tag{3.2.21}$$

along with the force and torque balance conditions

$$\int_\Gamma \boldsymbol{\psi}(\boldsymbol{x})\, \mathrm{d}S_{\boldsymbol{x}} = \boldsymbol{f} \quad \text{and} \quad \int_\Gamma (\boldsymbol{x} - \boldsymbol{q}) \times \boldsymbol{\psi}(\boldsymbol{x})\, \mathrm{d}S_{\boldsymbol{x}} = \boldsymbol{\tau}. \tag{3.2.22}$$

Equations (3.2.21) and (3.2.22) together define a linear system of equations to be solved for the *single-layer density* $\boldsymbol{\psi}(\boldsymbol{x} \in \Gamma)$ and particle velocity $\boldsymbol{U} = \{\boldsymbol{u}, \boldsymbol{\omega}\}$. We remark that $\boldsymbol{\psi}$ is the jump in the normal component of the stress when going across the body surface from the "interior" flow to the "exterior" flow. If $\breve{\boldsymbol{v}} = 0$, then $\boldsymbol{\psi} = \boldsymbol{\lambda}$ is the traction.

In operator notation, we can write the system formed by Eqs. (3.2.21) and (3.2.22) as a *saddle-point* problem,

$$\begin{bmatrix} \boldsymbol{\mathcal{M}} & -\boldsymbol{\mathcal{K}} \\ -\boldsymbol{\mathcal{K}}^* & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\psi} \\ \boldsymbol{U} \end{bmatrix} = - \begin{bmatrix} \breve{\boldsymbol{v}} \\ \boldsymbol{F} \end{bmatrix}, \tag{3.2.23}$$

where $\boldsymbol{\mathcal{M}}$ denotes the *single-layer* integral operator defined by

$$(\boldsymbol{\mathcal{M}}\boldsymbol{\psi})(\boldsymbol{x} \in \Gamma) = \int_\Gamma \mathbb{G}(\boldsymbol{x}, \boldsymbol{y})\, \boldsymbol{\psi}(\boldsymbol{y})\, \mathrm{d}S_{\boldsymbol{y}}, \tag{3.2.24}$$

and $\boldsymbol{\mathcal{K}}$ is a geometric operator that relates particle velocity to surface velocities,

$$(\boldsymbol{\mathcal{K}}\boldsymbol{U})(\boldsymbol{x} \in \Gamma) = \boldsymbol{u} + \boldsymbol{\omega} \times (\boldsymbol{x} - \boldsymbol{q}), \tag{3.2.25}$$

and its adjoint $\boldsymbol{\mathcal{K}}^\star$ is an integral operator that converts the single-layer density $\boldsymbol{\psi}$ to a force and torque,

$$\boldsymbol{\mathcal{K}}^\star \boldsymbol{\psi} = \left(\int_\Gamma \boldsymbol{\psi}(\boldsymbol{x})\, \mathrm{d}S_{\boldsymbol{x}}\ ,\ \int_\Gamma (\boldsymbol{x} - \boldsymbol{q}) \times \boldsymbol{\psi}(\boldsymbol{x})\, \mathrm{d}S_{\boldsymbol{x}} \right) = (\boldsymbol{f}, \boldsymbol{\tau}). \tag{3.2.26}$$

The covariance of the random surface velocity $\breve{\boldsymbol{v}}$ can be written as

$$\langle \breve{\boldsymbol{v}} \breve{\boldsymbol{v}} \rangle = \frac{2k_B T}{\Delta t} \boldsymbol{\mathcal{M}}, \tag{3.2.27}$$

by which we mean that $\boldsymbol{\mathcal{M}}\boldsymbol{\psi}' = \langle (\breve{\boldsymbol{v}}, \boldsymbol{\psi}')\, \breve{\boldsymbol{v}} \rangle$, for all $\boldsymbol{\psi}'$ in L^2-space, and $(\cdot, \cdot)$ denotes the L^2- inner product defined by $(\boldsymbol{f}, \boldsymbol{g}) = \int_\Gamma \boldsymbol{f}(\boldsymbol{x}) \cdot \boldsymbol{g}(\boldsymbol{x})\, \mathrm{d}S_{\boldsymbol{x}}$. If the random surface velocity $\breve{\boldsymbol{v}}$ were a function and could therefore be evaluated pointwise, Eq. (3.2.27) would simply be a formal rewriting of Eq. (3.2.11). However, we reminder the reader again that Eq. (3.2.11) is also formal (in the same way that Eq. (3.2.2) is) and $\breve{\boldsymbol{v}}$ is a distribution and therefore *cannot* be evaluated pointwise. We can define $\breve{\boldsymbol{v}}$ more precisely as follows. Since $\boldsymbol{\mathcal{M}}$ is a compact, self-adjoint and positive-semidefinite operator in the L_2 sense, it has countably infinitely many eigenvalues $\lambda_i \geq 0$ and orthonormal eigenfunctions $\boldsymbol{w}_i$, so we may write $\breve{\boldsymbol{v}}$ in the Karhunen-Loève expansion

$$\breve{\boldsymbol{v}} \overset{\text{d.}}{=} \sum_{i=1}^{\infty} \sqrt{\lambda_i}\, W_i\, \boldsymbol{w}_i, \tag{3.2.28}$$

where as before W_i are independent Gaussian random variables with mean zero and variance $2k_BT/\Delta t$. In formal operator notation, we will write Eq. (3.2.28) as $\breve{\boldsymbol{v}} = \boldsymbol{\mathcal{M}}^{\frac{1}{2}}\boldsymbol{W}$, where $\left(\boldsymbol{\mathcal{M}}^{\frac{1}{2}}\right)\left(\boldsymbol{\mathcal{M}}^{\frac{1}{2}}\right)^{\top} = \boldsymbol{\mathcal{M}}$. We have found Eq. (3.2.27) (equivalently, Eq. (3.2.28)), rather than the deceptively simple Eq. (3.2.11), to be a suitable starting point for a finite-dimensional discretization of $\breve{\boldsymbol{v}}$, as we explain shortly.

In this formulation, we require that $\breve{\boldsymbol{v}}(\boldsymbol{x})$ is consistent with a divergence-free velocity field in the extended domain, i.e.,

$$\int_{\Gamma} \breve{\boldsymbol{v}}(\boldsymbol{x}) \cdot \boldsymbol{n}(\boldsymbol{x})\, \mathrm{d}S_{\boldsymbol{x}} = 0, \tag{3.2.29}$$

which is required for Eq. (3.2.23) to be solvable since the single-layer operator $\boldsymbol{\mathcal{M}}$ has a nontrivial null space consisting of single-layer densities that are normal to the boundary,

$$(\boldsymbol{\mathcal{M}}\boldsymbol{n})(\boldsymbol{x} \in \Gamma) = \int_{\Gamma} \mathbb{G}(\boldsymbol{x}, \boldsymbol{y})\, \boldsymbol{n}(\boldsymbol{y})\, \mathrm{d}S_{\boldsymbol{y}} = \boldsymbol{0}. \tag{3.2.30}$$

From Eq. (3.2.28), we note that $\breve{\boldsymbol{v}} = \boldsymbol{\mathcal{M}}^{\frac{1}{2}}\boldsymbol{W}$ is perpendicular to the null space of $\boldsymbol{\mathcal{M}}$, and hence, the solvability condition Eq. (3.2.29) is fulfilled.

Formally[3], taking the Schur complement of Eq. (3.2.23) to eliminate $\boldsymbol{\psi}$ and solving for

[3]For the operator $\boldsymbol{\mathcal{M}}^{-1}$ to be well-defined, we need to resort to in its precise definition either the space of

the body motion $\boldsymbol{U}$, we obtain

$$\begin{aligned}\boldsymbol{U} = \bar{\boldsymbol{U}} + \tilde{\boldsymbol{U}} &= (\boldsymbol{\mathcal{K}}^{\star}\boldsymbol{\mathcal{M}}^{-1}\boldsymbol{\mathcal{K}})^{-1}\boldsymbol{F} + (\boldsymbol{\mathcal{N}}\boldsymbol{\mathcal{K}}^{\star}\boldsymbol{\mathcal{M}}^{-1})\breve{\boldsymbol{v}} \\ &= (\boldsymbol{\mathcal{K}}^{\star}\boldsymbol{\mathcal{M}}^{-1}\boldsymbol{\mathcal{K}})^{-1}\boldsymbol{F} + (\boldsymbol{\mathcal{N}}\boldsymbol{\mathcal{K}}^{\star}\boldsymbol{\mathcal{M}}^{-1}\boldsymbol{\mathcal{M}}^{\frac{1}{2}})\boldsymbol{W} \\ &= \boldsymbol{\mathcal{N}}\boldsymbol{F} + \boldsymbol{\mathcal{N}}^{\frac{1}{2}}\boldsymbol{W},\end{aligned} \tag{3.2.31}$$

which allows us to formally define $\boldsymbol{\mathcal{N}}$ and $\boldsymbol{\mathcal{N}}^{\frac{1}{2}}$ explicitly as,

$$\begin{aligned}\boldsymbol{\mathcal{N}} &= (\boldsymbol{\mathcal{K}}^{\star}\boldsymbol{\mathcal{M}}^{-1}\boldsymbol{\mathcal{K}})^{-1}, \\ \boldsymbol{\mathcal{N}}^{\frac{1}{2}} &= \boldsymbol{\mathcal{N}}\boldsymbol{\mathcal{K}}^{\star}\boldsymbol{\mathcal{M}}^{-1}\boldsymbol{\mathcal{M}}^{\frac{1}{2}}.\end{aligned} \tag{3.2.32}$$

Note that the random surface velocity $\breve{\boldsymbol{v}}$ is in the range of $\boldsymbol{\mathcal{M}}$ by the construction of Eq. (3.2.28), so that $\boldsymbol{\mathcal{M}}^{-1}\breve{\boldsymbol{v}}$ is well-defined. We can therefore formally show that fluctuation-dissipation balance holds in the continuum operator sense,

$$\begin{aligned}\boldsymbol{\mathcal{N}}^{\frac{1}{2}}\left(\boldsymbol{\mathcal{N}}^{\frac{1}{2}}\right)^{\top} &= \boldsymbol{\mathcal{N}}\boldsymbol{\mathcal{K}}^{\star}\boldsymbol{\mathcal{M}}^{-1}\boldsymbol{\mathcal{M}}^{\frac{1}{2}}\left(\boldsymbol{\mathcal{M}}^{\frac{1}{2}}\right)^{\top}\boldsymbol{\mathcal{M}}^{-1}\boldsymbol{\mathcal{K}}\boldsymbol{\mathcal{N}} \\ &= \boldsymbol{\mathcal{N}}(\boldsymbol{\mathcal{K}}^{\star}\boldsymbol{\mathcal{M}}^{-1}\boldsymbol{\mathcal{K}})\boldsymbol{\mathcal{N}} \\ &= \boldsymbol{\mathcal{N}}(\boldsymbol{\mathcal{N}})^{-1}\boldsymbol{\mathcal{N}} = \boldsymbol{\mathcal{N}}.\end{aligned} \tag{3.2.33}$$

This shows that the desired random velocity $\boldsymbol{U}$ in Eq. (3.2.8) can be generated by solving Eq. (3.2.23), which is the first-kind boundary integral formulation of the SSBVP (3.2.10). While at first sight it may appear that we have simply formally rederived Eq. (3.2.19) by appealing to a first-kind BVP formulation, we will demonstrate next that the formal continuum formulation presented here has a well-defined finite-dimensional truncation that is very suitable for numerical computations.

band-limited functions or the fractional Sobolov space, but here, we will simply use this formal computation to inform our discretization and show later in Sec. 3.3.1 that our finite-dimensional discretization converges numerically.

3.3 Fluctuating boundary integral method

This section presents the fluctuating boundary integral method (FBIM) for suspensions of Brownian rigid particles in two dimensions. Our discrete formulation closely follows the (formal) continuum first-kind formulation presented in Sec. 3.2.2. We have found this to be much more effective than following the more general BVP formulation presented in Sec. 3.2.1. Specifically, one approach to discretizing the SSBVP Eq. (3.2.10) is to first generate a (smooth) random surface velocity *function* $\breve{\boldsymbol{v}}$ using a *regularized*[4] variant of Eq. (3.2.11), and then to use a standard spectrally-accurate second-kind boundary integral formulation to solve the resulting boundary-value problem. Our preliminary investigations of such an approach have revealed that regularization (truncation) of the covariance in Eq. (3.2.11) leads to a drastic loss of accuracy in numerical fluctuation-dissipation balance. Instead, by relating the covariance of the random surface velocity to the first-kind operator as in Eq. (3.2.27), the regularization of the *distribution* $\breve{\boldsymbol{v}}$ becomes directly connected to the singular quadrature used to discretize $\boldsymbol{\mathcal{M}}$. As we demonstrate here, this leads to a first-kind formulation that satisfies discrete fluctuation-dissipation (DFDB) to within solver tolerances, while preserving the underlying accuracy of the singular quadrature.

We begin by presenting a discrete formulation of the mobility problem (3.2.23) by first discretizing the continuum operators $\boldsymbol{\mathcal{M}}$, $\boldsymbol{\mathcal{K}}$ and $\boldsymbol{\mathcal{K}}^{\star}$, to obtain a discrete saddle-point linear system, whose solution strictly obeys DFDB without any approximation. To efficiently solve the saddle-point linear system with Krylov iterative methods, we present the two key components of FBIM: a fast routine for computing matrix-vector product of the single-layer matrix $\mathbf{M}$ (i.e., the discretized operator $\boldsymbol{\mathcal{M}}$), and a fast method for generating the random surface velocity $\breve{\mathbf{v}} = \mathbf{M}^{1/2}\mathbf{W}$. To address the inherent ill-conditioning of the linear system arising from the first-kind integral formulation, we will also discuss block-diagonal

[4]The most direct way to regularize the singular Green's function is to represent it in Fourier space and then simply truncate the finite-dimensional sum to a finite number of Fourier modes.

preconditioning for the iterative solvers.

3.3.1 Discrete formulation of the mobility problem

We present the discrete formulation of the mobility problem (3.2.23) by first discretizing the continuum operators $\boldsymbol{\mathcal{K}}$, $\boldsymbol{\mathcal{K}}^\star$ and $\boldsymbol{\mathcal{M}}$. In the discrete formulation, for generality, we describe our method for many-body suspensions. Let us assume that Γ_β is parametrized by $\boldsymbol{\gamma}_\beta : [0, 2\pi] \to \mathbb{R}^2$. We introduce a collection of N_p equispaced points $s_j = j\Delta s$, $j \in \{1, \ldots, N_p\}$, where $\Delta s = 2\pi/N_p$, and Γ_β is discretized by the collection of nodes $\mathbf{x}_\beta = \{\mathbf{x}_{\beta,j}\}_{j=1}^{N_p}$, where $\mathbf{x}_{\beta,j} = \boldsymbol{\gamma}_\beta(s_j)$. We also denote the composite vector of all nodes or points by $\mathbf{X} = \{\mathbf{x}_\beta\}_{\beta=1}^{N_b}$.

The discrete operator $\mathbf{K}$ is a geometric matrix defined by

$$(\boldsymbol{\mathcal{K}}\boldsymbol{U})(\mathbf{x}_{\beta,j}) = (\mathbf{K}_\beta \boldsymbol{U}_\beta)_j = \boldsymbol{u}_\beta + \boldsymbol{\omega}_\beta \times (\mathbf{x}_{\beta,j} - \boldsymbol{q}_\beta), \tag{3.3.1}$$

where $\mathbf{K}_\beta$ is the sub-block of $\mathbf{K}$ that maps the particle velocity $\boldsymbol{U}_\beta = \{\boldsymbol{u}_\beta, \boldsymbol{\omega}_\beta\}$ to the velocity at the node $\mathbf{x}_{\beta,j}$ on Γ_β.

The adjoint operator $\boldsymbol{\mathcal{K}}^\star$ defined by Eq. (3.2.26) can be discretized by the periodic trapezoidal rule for each body,

$$\int_{\Gamma_\beta} \boldsymbol{\psi}(\boldsymbol{x})\, \mathrm{d}S_{\boldsymbol{x}} \;\approx\; \sum_{j=1}^{N_p} \boldsymbol{\mu}_{\beta,j}, \tag{3.3.2a}$$

$$\int_{\Gamma_\beta} (\boldsymbol{x} - \boldsymbol{q}_\beta) \times \boldsymbol{\psi}(\boldsymbol{x})\, \mathrm{d}S_{\boldsymbol{x}} \;\approx\; \sum_{j=1}^{N_p} (\mathbf{x}_{\beta,j} - \boldsymbol{q}_\beta) \times \boldsymbol{\mu}_{\beta,j}, \tag{3.3.2b}$$

where $\boldsymbol{\mu}_{\beta,j} = \boldsymbol{\psi}(\mathbf{x}_{\beta,j})\, \Delta s$ denotes an unknown discrete boundary force at the node $\mathbf{x}_{\beta,j}$ on Γ_β. We can therefore write

$$(\boldsymbol{\mathcal{K}}^\star \boldsymbol{\psi})(\mathbf{X}) \approx \mathbf{K}^\top \boldsymbol{\mu}, \tag{3.3.3}$$

where $\boldsymbol{\mu} = \left\{\boldsymbol{\mu}_\beta\right\}_{\beta=1}^{N_b}$ is a composite vector that collects all the boundary forces.

To approximate the single-layer operator $\boldsymbol{\mathcal{M}}$, we need to employ a singular quadrature that can handle the singularity of $\mathbb{G}$. The development of quadrature rules for singular or near-singular integrals is an active research area of its own [92, 105, 106, 89, 90, 107]. In two dimensions, for simplicity, we will use the Alpert quadrature which is based on a modification of the trapezoidal rule with auxiliary nodes whose weights are configured to achieve the desired order of accuracy [92].

In matrix notation, the approximation of the single-layer integral operator evaluated on the vector of quadrature nodes $\mathbf{X}$ can be compactly written as

$$(\boldsymbol{\mathcal{M}}\boldsymbol{\psi})(\mathbf{X}) \approx \mathbf{M}\boldsymbol{\mu}, \tag{3.3.4}$$

where $\mathbf{M}$ is the discretized matrix operator of $\boldsymbol{\mathcal{M}}$, hereinafter referred to as the single-layer matrix. The details of constructing $\mathbf{M}$ will be discussed in Sec. 3.3.2. We have chosen to keep track of surface forces in the discrete representation (rather than surface tractions as in the continuous case) as this yields both a symmetric saddle-point system in the usual sense, as well as the desirable property that $\mathbf{M}_{ij} \to \mathbb{G}(\mathbf{x}_i, \mathbf{x}_j)$ as $|\mathbf{x}_i - \mathbf{x}_j| \to \infty$. The matrix $\mathbf{M}$ has a physical interpretation of a mobility matrix relating surface forces to surface velocities, which in turn suggests that $\mathbf{M}$ should be symmetric (self-adjoint) in the standard L_2 sense.

Following the continuum formulation (see Eq. (3.2.27)), we require the discrete random surface velocity[5] to satisfy

$$\langle \check{\mathbf{v}}\check{\mathbf{v}}^{\top} \rangle = \frac{2k_BT}{\Delta t}\mathbf{M}. \tag{3.3.5}$$

More precisely, we need to generate a vector of Gaussian random variables $\check{\mathbf{v}}$ whose covariance is given by Eq. (3.3.5), and we denote it by $\check{\mathbf{v}} = \mathbf{M}^{1/2}\mathbf{W}$, where $\mathbf{W}$ is a finite-dimensional

[5]The discrete $\check{\mathbf{v}}$ can be thought of as being a suitably scaled finite-volume representation of the distribution $\check{\boldsymbol{v}}$.

vector of Gaussian random variables with mean zero and covariance

$$\langle W_i W_j \rangle = \frac{2k_B T}{\Delta t}\delta_{ij}, \tag{3.3.6}$$

and $\mathbf{M}^{1/2}$ satisfies

$$\left(\mathbf{M}^{1/2}\right)\left(\mathbf{M}^{1/2}\right)^{\top} = \mathbf{M}. \tag{3.3.7}$$

The discrete formulation of Eq. (3.2.23) is a saddle-point linear system for the boundary forces $\boldsymbol{\mu}$ and the rigid body motion $\boldsymbol{U}$,

$$\begin{bmatrix} \mathbf{M} & -\mathbf{K} \\ -\mathbf{K}^{\top} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\mu} \\ \boldsymbol{U} \end{bmatrix} = - \begin{bmatrix} \mathbf{M}^{1/2}\mathbf{W} \\ \boldsymbol{F} \end{bmatrix}. \tag{3.3.8}$$

By taking the Schur complement of $\mathbf{M}$ and eliminating $\boldsymbol{\mu}$, we obtain

$$\begin{aligned} \boldsymbol{U} &= \boldsymbol{N}\boldsymbol{F} + \boldsymbol{N}^{\frac{1}{2}}\mathbf{W} \\ &= (\mathbf{K}^{\top}\mathbf{M}^{\dagger}\mathbf{K})^{-1}\boldsymbol{F} + (\boldsymbol{N}\mathbf{K}^{\top}\mathbf{M}^{\dagger}\mathbf{M}^{1/2})\mathbf{W}, \end{aligned} \tag{3.3.9}$$

from which we can define

$$\begin{aligned} \boldsymbol{N} &= (\mathbf{K}^{\top}\mathbf{M}^{\dagger}\mathbf{K})^{-1}, \\ \boldsymbol{N}^{\frac{1}{2}} &= \boldsymbol{N}\mathbf{K}^{\top}\mathbf{M}^{\dagger}\mathbf{M}^{1/2}, \end{aligned} \tag{3.3.10}$$

where $\mathbf{M}^{\dagger}$ is the pseudo-inverse of $\mathbf{M}$, and $\boldsymbol{N}$ is an approximation of $\boldsymbol{\mathcal{N}}$ up to the order of accuracy of Alpert quadrature. Under the definition of $\boldsymbol{N}$ and $\boldsymbol{N}^{\frac{1}{2}}$, we can verify that our discrete formulation satisfies DFDB without any approximation,

$$
\begin{aligned}
\boldsymbol{N}^{\frac{1}{2}}\left(\boldsymbol{N}^{\frac{1}{2}}\right)^{\top} &= \boldsymbol{N}\mathbf{K}^{\top}\mathbf{M}^{\dagger}\mathbf{M}^{1/2}\left(\mathbf{M}^{1/2}\right)^{\top}\mathbf{M}^{\dagger}\mathbf{K}\boldsymbol{N}, \\
&= \boldsymbol{N}(\mathbf{K}^{\top}\mathbf{M}^{\dagger}\mathbf{K})\boldsymbol{N}, \\
&= \boldsymbol{N}(\boldsymbol{N})^{-1}\boldsymbol{N} = \boldsymbol{N}.
\end{aligned}
\tag{3.3.11}
$$

Note that these relations are well-defined finite-dimensional versions of the formal continuum equations (3.2.32) and (3.2.33).

To generate the random velocity given by Eq. (3.3.9) efficiently, we solve the saddle-point linear system (3.3.8) by GMRES. In the remaining sections, we present efficient numerical that compute the matrix-vector product $\mathbf{M}\boldsymbol{\mu}$ and generate the random surface velocity $\breve{\mathbf{v}} = \mathbf{M}^{1/2}\mathbf{W}$.

3.3.2 Fast matrix-vector multiplication for the single-layer matrix

In this section we develop a fast method to efficiently perform the matrix-vector product $\mathbf{M}\boldsymbol{\mu}$. The key idea for achieving linear-scaling is to use Ewald splitting to decompose the periodic Stokeslet as

$$
\mathbb{G} = \mathbb{G}_{\xi}^{(w)} + \mathbb{G}_{\xi}^{(r)} = H * \mathbb{G} + (\mathbb{G} - H * \mathbb{G}),
\tag{3.3.12}
$$

where "$*$" denotes convolution, and $\mathbb{G}_{\xi}^{(w)}$ is a far-field (long-ranged) smooth kernel that decays exponentially in Fourier space, and $\mathbb{G}_{\xi}^{(r)}$ is a near-field (short-ranged) singular kernel that decays exponentially in real space. The choice of splitting function $H(r;\xi)$ by Hasimoto [86] is defined in Fourier space as

$$
\widehat{H}(k;\xi) = \left(1 + \frac{k^2}{4\xi^2}\right) e^{-k^2/4\xi^2},
\tag{3.3.13}
$$

where ξ is the splitting parameter that controls the rate of exponential decay.

Using the Fourier representation of the periodic Green's function of Stokes flow we can compute the far-field kernel in wave space as

$$\mathbb{G}_\xi^{(w)}(\boldsymbol{r}) = \frac{1}{\eta V} \sum_{\mathbf{k}\neq 0} \frac{\widehat{H}(k;\xi)}{k^2} (\mathbf{I} - \hat{\mathbf{k}}\hat{\mathbf{k}}^\top)\, e^{-i\mathbf{k}\cdot\boldsymbol{r}}, \tag{3.3.14}$$

where $V = |\mathcal{V}| = L^2$, $\mathbf{k} \in \{2\pi\kappa_i/L : \kappa_i \in \mathbb{Z},\ i = 1, 2\}$, and $\hat{\mathbf{k}} = \mathbf{k}/k$ for $k = |\mathbf{k}|$. The near-field kernel is analytically computed in real space using the inverse Fourier transform[6],

$$\mathbb{G}_\xi^{(r)}(\boldsymbol{r}) = \frac{1}{4\pi\eta} \left[\frac{1}{2} E_1(\xi^2 r^2)\mathbf{I} + \left(\frac{\boldsymbol{r}\otimes\boldsymbol{r}}{r^2} - \mathbf{I} \right) e^{-\xi^2 r^2} \right], \tag{3.3.15}$$

where $E_1(z)$ is the exponential integral defined by

$$E_1(z) = \int_1^\infty \frac{e^{-zt}}{t} dt = \int_z^\infty \frac{e^{-t}}{t} dt. \tag{3.3.16}$$

We observe from Eq. (3.3.15) that $\mathbb{G}_\xi^{(r)}$ also has the logarithmic singularity of $\mathbb{G}$, since in the limit $z \to 0$,

$$E_1(z) = -\gamma - \log z + O(z), \tag{3.3.17}$$

where γ is the Euler-Mascheroni constant. An important remark on the Hasimoto splitting is that it ensures both $\mathbb{G}_\xi^{(r)}$ and $\mathbb{G}_\xi^{(w)}$ are SPD, because $0 \le \widehat{H}(k;\xi) \le 1$ for all $\mathbf{k}$ and ξ [57].

The splitting of $\mathbb{G}$ naturally induces the splitting of the single-layer integral operator $\boldsymbol{\mathcal{M}} = \boldsymbol{\mathcal{M}}^{(r)} + \boldsymbol{\mathcal{M}}^{(w)}$, and subsequently, the corresponding splitting of the single-layer matrix

$$\mathbf{M} = \mathbf{M}^{(r)} + \mathbf{M}^{(w)}, \tag{3.3.18}$$

[6]We gratefully thank Anna-Karin Tornberg for sharing with us notes on the Hasimoto splitting of the Stokeslet in two dimensions.

where the elements of $\mathbf{M}^{(w)}$ are obtained by applying the regular trapezoidal rule to $\boldsymbol{\mathcal{M}}^{(w)}$, which gives

$$\left(\mathbf{M}^{(w)}\right)_{mn} = \mathbb{G}_\xi^{(w)}(\mathbf{x}_m - \mathbf{x}_n). \tag{3.3.19}$$

The elements of $\mathbf{M}^{(r)}$ are obtained by applying Alpert's hybrid Gauss-trapezoidal quadrature to $\boldsymbol{\mathcal{M}}^{(r)}$, and can be futher decomposed as

$$\mathbf{M}^{(r)} = \mathbf{M}^{(t)} + \mathbf{M}^{(a)}. \tag{3.3.20}$$

where $\mathbf{M}^{(t)}$ is the trapezoidal rule, i.e., $\left(\mathbf{M}^{(t)}\right)_{mn} = \mathbb{G}_\xi^{(r)}(\mathbf{x}_m - \mathbf{x}_n)$ for $m \neq n$, and we define it to be zero for $m = n$. The singular quadrature correction $\mathbf{M}^{(a)}$ is a banded matrix that contains Alpert weights for the logarithmic singularity of $\mathbb{G}_\xi^{(r)}$. The Alpert weights are defined on a set of auxiliary nodes that do not coincide with the trapezoidal nodes, so a local Lagrangian interpolation from the auxiliary nodes to the trapezoidal nodes is needed to obtain the elements of $\mathbf{M}^{(a)}$. Note that $\mathbf{M}^{(r)}$, $\mathbf{M}^{(w)}$ and $\mathbf{M}^{(a)}$ depend on ξ, but for conciseness of notation, the subscript ξ is omitted.

A key ingredient of FBIM is a fast method to compute the matrix-vector product

$$\mathbf{M}\boldsymbol{\mu} = \left(\mathbf{M}^{(a)} + \mathbf{M}^{(t)} + \mathbf{M}^{(w)}\right)\boldsymbol{\mu}. \tag{3.3.21}$$

We recall that the Alpert quadrature assigns only local correction weights to the trapezoidal nodes. Since $\mathbf{M}^{(a)}$ is block-diagonal and banded, matrix-vector products involving $\mathbf{M}^{(a)}$ can be computed in $O(N_b)$ operations using vector rotations and sparse matrix-vector multiplications,

$$\left(\mathbf{M}^{(a)}\boldsymbol{\mu}\right)_\beta = \mathbf{R}_\beta\, \mathbf{M}_{\text{ref}}^{(a)}\, \mathbf{R}_\beta^\top\, \boldsymbol{\mu}_\beta, \quad \beta = 1, \ldots, N_b, \tag{3.3.22}$$

where $\mathbf{M}_{\text{ref}}^{(a)}$ is a precomputed Alpert matrix for some reference configuration, and $\mathbf{R}_\beta$ is the rotation matrix from the chosen reference configuration to the configuration of Γ_β.

To accelerate matrix-vector products involving $\mathbf{M}^{(r)}$ and $\mathbf{M}^{(w)}$ which are not sparse, we rely on the Spectral Ewald method [77]. For the near-field contribution, due to the short-ranged nature of $\mathbb{G}_\xi^{(r)}$, the action of $\mathbf{M}^{(t)}$ can be computed by using the cell list algorithm, commonly-used in Molecular Dynamics [108]. First, we partition the computational domain into $N_{\text{box}} \times N_{\text{box}}$ cells and sort the points into these cells, which takes $O(N_b)$ work. The splitting parameter ξ is chosen such that the real-space sum converges to within a prescribed tolerance ϵ, at a cutoff radius $r_c = L/N_{\text{box}}$. For each target point, the real-space sum is reduced to a local interaction with source points in its own cell and in all adjacent cells (with periodicity), i.e., nine cells in two dimensions. If the density of points in each cell is approximately held fixed as the system size grows, the complexity of the direct summation in the near field is also $O(N_b)$.

In the SE method, the far-field contribution given by Eqs. (3.3.14) and (3.3.19) can be factorized as

$$\mathbf{M}^{(w)} = \mathbf{D}^\star \mathbf{B} \mathbf{D}, \tag{3.3.23}$$

where the block-diagonal matrix $\mathbf{B}$ is defined in the Fourier space as

$$\mathbf{B}(\mathbf{k}, \xi) = \frac{1}{k^2} H(k, \xi)(\mathbf{I} - \hat{\mathbf{k}}\hat{\mathbf{k}}^\top), \tag{3.3.24}$$

which essentially maps the Fourier representation of forces to velocities. The operator $\mathbf{D}$ is the non-uniform Discrete Fourier Transform (NUDFT) that converts point forces $\boldsymbol{\mu} = \{\boldsymbol{\mu}_n\}$ on a collection of non-uniform source points $\{\mathbf{x}_n\}$ to Fourier space. The operator $\mathbf{D}$ and its adjoint $\mathbf{D}^\star$ can be efficiently applied using the non-uniform Fast Fourier Transform (NUFFT) (see [34] and the references therein). In operator notation, we can express the NUFFT as

$$\mathbf{D} = \boldsymbol{\mathcal{C}}\boldsymbol{\mathcal{F}}\boldsymbol{\mathcal{S}} \quad \text{and} \quad \mathbf{D}^\star = \boldsymbol{\mathcal{S}}^\star \boldsymbol{\mathcal{F}}^\star \boldsymbol{\mathcal{C}}^\star, \tag{3.3.25}$$

where $\boldsymbol{\mathcal{S}}$ and $\boldsymbol{\mathcal{S}}^\star$ are a pair of spreading and interpolation operators using Gaussian kernels, and $\boldsymbol{\mathcal{F}}$ and $\boldsymbol{\mathcal{F}}^\star$ are the forward and inverse FFT operators on a uniform grid, and $\boldsymbol{\mathcal{C}}$ and $\boldsymbol{\mathcal{C}}^\star$ are "deconvolution" operators.

The main idea of NUFFT is to first smear (spread) the point forces to a uniform grid using a Gaussian kernel, then make use of the FFT on the uniform grid, and finally apply the deconvolution (see [34]). In the SE method, the Gaussian kernel with the Fourier representation $e^{-\eta k^2/8\xi^2}$ is used for spreading and interpolation, where η is a free parameter that sets the shape of Gaussian [77, 109]. We observe that this Gaussian with parameter η is different from the one in the Hasimoto function for Ewald splitting. The main purpose for introducing an extra parameter η is that it allows the SE method to independently control the width of spreading and interpolation, so that the accuracy of evaluating the wave-space sum with NUFFT is decoupled from the accuracy of the Ewald sum.

Next we address the choice of parameters in the SE method. The parameters that need to be set by the user include the number of partition cells N_{box} (or r_c), the splitting parameter ξ, the size of the Fourier grid M (even), the number of points P (odd) for spreading and interpolation in the NUFFT, and the Gaussian shape parameter m (which is related to η, see [77, Eq. (22)]). First, we set the number of partition cells N_{box}, primarily based on balancing the computational work between the real- and wave-space sums (see Sec. 3.4.3 for details), and set $r_c = L/N_{\text{box}}$. For a user-specified error tolerance level ϵ, the splitting parameter ξ and M are determined by the truncation error estimates in the real and wave spaces [77], respectively,

$$C_r e^{-\xi^2 r_c^2} \leq \epsilon, \tag{3.3.26a}$$

$$C_w e^{-k_{\max}^2/4\xi^2} \leq \epsilon, \tag{3.3.26b}$$

where $k_{\max}$ corresponds to the largest mode of a grid of size $M \times M$. The constants in the error estimates are estimated empirically as in [77], and we have found that $C_r \approx 100$ and $C_w \approx 1$ are suitable for the Stokeslet in two dimensions. The remaining parameters P and m (or η) together determine the accuracy of evaluating the wave-space sum using NUFFT. From the error estimation of Lindbo and Tornberg [77, 109], the optimal choice is $m \sim \sqrt{\pi P}$, so that the approximation error arising from the NUFFT (including quadrature error and Gaussian truncation error) is approximately $e^{-\pi P/2} \lesssim \epsilon$, from which P is determined.

The overall complexity of the wave-space sum is $O(N_b)$ for spreading and interpolation and $O(M^2 \log M^2)$ for the FFTs. For dense suspensions in two dimensions, we have found in our implementation that the computational work is dominated by spreading and interpolation instead of the FFTs.

3.3.3 Fast sampling of the random surface velocity

The second key component of FBIM is a fast routine for sampling the random surface velocity $\breve{\mathbf{v}} = \mathbf{M}^{1/2}\mathbf{W}$. The action of $\mathbf{M}^{1/2}$ can be computed using iterative methods such as the Chebyshev polynomial approximation [78] or the Lanczos algorithm for matrix square root [79]. For the Stokeslet in two dimensions, applying iterative methods directly for $\mathbf{M}$ are expected to scale poorly for dense suspensions because of its logarithmic growth in the far field. The condition number of $\mathbf{M}$, as well as the number of iterations required to converge for a given tolerance level, would grow with the system size. This would make the overall complexity for applying the action of $\mathbf{M}^{1/2}$ super-linear, even though the action of $\mathbf{M}$ can be applied with $O(N_b)$ work.

To improve the super-linear complexity for generating the random surface velocity with iterative methods, we use the idea of the *Positively Split Ewald* (PSE) method developed by Fiore *et al.* [57] for the RPY tensor. The main idea of PSE is to split the action of $\mathbf{M}^{1/2}$ into a near-field part $\left(\mathbf{M}^{(r)}\right)^{1/2}$ and a far-field part $\left(\mathbf{M}^{(w)}\right)^{1/2}$, and generate the random surface velocity as

$$\mathbf{M}^{1/2}\mathbf{W} \overset{\text{d.}}{=} \left(\mathbf{M}^{(r)}\right)^{1/2}\mathbf{W}^{(r)} + \left(\mathbf{M}^{(w)}\right)^{1/2}\mathbf{W}^{(w)}, \tag{3.3.27}$$

so that the near-field contribution can be rapidly generated by the Lanczos algorithm [79] in the real space, and the far-field contribution can be efficiently handled in the wave space by NUFFT. The right-hand-side of Eq. (3.3.27) defines one way of computing the action of $\mathbf{M}^{1/2}$ provided that $\mathbf{W}^{(r)}$ and $\mathbf{W}^{(w)}$ are two independent Gaussian random vectors.

For a sufficiently small cut-off radius or a sufficiently large ξ, the real-space kernel $\mathbb{G}_\xi^{(r)}$ decays exponentially, so that the near-field interaction is localized and the condition number of $\mathbf{M}^{(r)}$ does not grow with the number of bodies, while the packing fraction is held fixed. However, due to the singular nature of $\mathbb{G}_\xi^{(r)}$, the condition number of $\mathbf{M}^{(r)}$ may grow if the number of points per body increases. We have found that the Lanczos algorithm [93] with block-diagonal preconditioning (see Sec. 3.3.4) can significantly reduce the number of iterations. In the case when two bodies nearly touch, so that the problem itself becomes ill-conditioned, the block-diagonal approximation becomes worse and the number of iterations for all iterative solvers increases. Nevertheless, we have found that the block-diagonal preconditioner is still effective, and it takes a reasonable number of iterations for the Lanczos algorithm to converge, even for dense suspensions (see Fig. 3.4).

Note that the validity of Eq. (3.3.27) relies on the property that both $\mathbf{M}^{(r)}$ and $\mathbf{M}^{(w)}$ need to be SPD. We observe that, even though $\mathbb{G}_\xi^{(r)}$ is a SPD kernel, its singular contribution given by the Alpert correction matrix $\mathbf{M}^{(a)}$ is not symmetric for a general-shaped particle. We have confirmed numerically that using only the symmetric part of $\mathbf{M}^{(a)}$ does not affect the accuracy of the Alpert quadrature. In practice, we observe that the Lanczos algorithm is rather insensitive to spurious negative eigenvalues of small magnitude that may exist for $(\mathbf{M}^{(r)} + (\mathbf{M}^{(r)})^\top)/2$.

The far-field matrix $\mathbf{M}^{(w)}$ is SPD by construction because of Eq. (3.3.14), and we can rewrite Eq. (3.3.23) as

$$\mathbf{M}^{(w)} = \left(\mathbf{D}^\star \mathbf{B}^{1/2}\right)\left(\mathbf{D}^\star \mathbf{B}^{1/2}\right)^\star, \tag{3.3.28}$$

with $\mathbf{B}^{1/2}$ defined in the wave space as

$$\mathbf{B}^{1/2}(\mathbf{k}, \xi) = \frac{1}{k}\widehat{H}^{1/2}(k;\xi)(\mathbf{I} - \hat{\mathbf{k}}\hat{\mathbf{k}}^\top), \tag{3.3.29}$$

The far-field contribution of the random surface velocity can be generated by

$$\left(\mathbf{M}^{(w)}\right)^{1/2}\mathbf{W}^{(w)} = \mathbf{D}^\star \mathbf{B}^{1/2}\mathbf{W}^{(w)} = \boldsymbol{\mathcal{S}}^\star \boldsymbol{\mathcal{F}}^\star \boldsymbol{\mathcal{C}}^\star \mathbf{B}^{1/2}\mathbf{W}^{(w)}, \tag{3.3.30}$$

where $\mathbf{W}^{(w)}(\boldsymbol{\kappa})$ is a complex-valued random vector in the wave space, and $\kappa_i \in \left\{-\frac{M}{2}, \ldots, \frac{M}{2} - 1\right\}$. The sequence of operations in Eq. (3.3.30) can be interpreted as follows. We first generate random numbers in the wave space, and project onto the divergence-free subspace by the projection $\mathbf{I} - \hat{\mathbf{k}}\hat{\mathbf{k}}^\top$, then scale by $\widehat{H}^{1/2}(k;\xi)/k$, and apply deconvolution and (inverse) FFTs to obtain velocities in the real space on the grid, and finally, perform interpolation using a Gaussian kernel to obtain the random surface velocities on the particles. This is equivalent to how random velocities are generated in methods like FIB [82] and fluctuating FCM [83, 84]. We remark that the cost of applying the action of $\left(\mathbf{M}^{(w)}\right)^{1/2}$ in Eq. (3.3.30) is even cheaper than the action of $\mathbf{M}^{(w)}$, since it only requires half the work.

We note that certain complex-conjugate symmetry of $\mathbf{W}^{(w)}(\boldsymbol{\kappa})$ must be maintained to ensure its Fourier transform gives a real-valued Gaussian random vector with the correct covariance in the real space. Specifically, we require that, the zeroth mode $\boldsymbol{\kappa} = (0,0)$ is set to be zero, and the Nyquist modes $\boldsymbol{\kappa} \in \left\{(-\frac{M}{2}, 0), (0, -\frac{M}{2}), (-\frac{M}{2}, -\frac{M}{2})\right\}$ are real-valued and generated from $\mathscr{N}(\mathbf{0}, \mathbf{I}_{2\times 2})$. All the remaining modes are generated by $\mathbf{W}^{(w)}(\boldsymbol{\kappa}) = \mathbf{a} + i\mathbf{b}$ for $\mathbf{a}, \mathbf{b} \in \mathscr{N}(\mathbf{0}, \frac{1}{2}\mathbf{I}_{2\times 2})$, and have the complex-conjugate symmetry: $\mathbf{W}^{(w)}(\boldsymbol{\kappa}) = (\mathbf{W}^{(w)}(\boldsymbol{\kappa}'))^*$, where $\boldsymbol{\kappa}' = -\boldsymbol{\kappa} \mod M$.

3.3.4 Block-diagonal preconditioning

This section presents the block-diagonal preconditioning technique introduced in [23] for solving the ill-conditioned linear system (3.3.8) with GMRES, and for generating the near-field contribution of the random surface velocity with the Lanczos algorithm [79, 93]. The block-diagonal preconditioner for the linear system (3.3.8) is obtained by neglecting all hydrodynamic interactions between different bodies, i.e.,

$$\mathbf{P} = \begin{bmatrix} \widetilde{\mathbf{M}} & -\mathbf{K} \\ -\mathbf{K}^\top & \mathbf{0} \end{bmatrix}, \tag{3.3.31}$$

where $\widetilde{\mathbf{M}}$ is a block-diagonal approximation of $\mathbf{M}$ obtained by setting elements of $\mathbf{M}$ corresponding to pairs of points on *distinct* bodies to zero,

$$\widetilde{\mathbf{M}}_{\alpha\beta} = \delta_{\alpha\beta}\mathbf{M}_{\alpha\beta}, \tag{3.3.32}$$

and the subscripts with Greek letters denote the sub-block containing interactions between Γ_α and Γ_β. Applying the preconditioner to Eq. (3.3.8) amounts to solving a linear system,

$$\begin{bmatrix} \widetilde{\mathbf{M}} & -\mathbf{K} \\ -\mathbf{K}^\top & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\mu} \\ \boldsymbol{U} \end{bmatrix} = - \begin{bmatrix} \check{\mathbf{v}} \\ \boldsymbol{F} \end{bmatrix}, \tag{3.3.33}$$

which requires the action of the approximate body mobility matrix (Schur complement),

$$\widetilde{\boldsymbol{N}} = \left(\mathbf{K}^\top \widetilde{\mathbf{M}}^\dagger \mathbf{K}\right)^{-1}. \tag{3.3.34}$$

The approximate body mobility matrix $\widetilde{\boldsymbol{N}}$ is also block-diagonal, and can be efficiently applied for each body,

$$\widetilde{\boldsymbol{N}}_{\beta\beta} = \left(\mathbf{K}_{\beta}^{\top}\mathbf{M}_{\beta\beta}^{\dagger}\mathbf{K}_{\beta}\right)^{-1}. \tag{3.3.35}$$

The matrix block $\mathbf{M}_{\beta\beta}$ in Eq. (3.3.35) is a small matrix with size $2N_p \times 2N_p$, which is precomputed for the reference configuration,

$$\mathbf{M}_{\text{ref}} = \mathbf{M}_{\text{ref}}^{(a)} + \mathbf{M}_{\text{ref}}^{(t)} + \mathbf{M}_{\text{ref}}^{(w)}. \tag{3.3.36}$$

The action of $\mathbf{M}_{\beta\beta}^{\dagger}$ can be efficiently applied by using $\mathbf{M}_{\text{ref}}^{\dagger}$, which is also precomputed using a dense SVD decomposition or eigenvalue decomposition, and by using rotation matrices for different bodies because of the translational and rotational invariance of the free-space Stokeslet,

$$\mathbf{M}_{\beta\beta}^{\dagger} \approx \mathbf{R}_{\beta}\mathbf{M}_{\text{ref}}^{\dagger}\mathbf{R}_{\beta}^{\top}. \tag{3.3.37}$$

We note that the two sides of Eq. (3.3.37) do not equal exactly, since the Alpert quadrature is not rotation-invariant for a general-shaped body. This is not an issue, since the error introduced by this artifact is within the error tolerance, and the preconditioner does not need to be exactly inverted to work effectively.

For the near-field contribution of the random surface velocity, we use the preconditioned Lanczos algorithm [93] to generate

$$\breve{\mathbf{v}}^{(r)} = \mathbf{G}^{\dagger}\left(\mathbf{G}\mathbf{M}^{(r)}\mathbf{G}^{\top}\right)^{1/2}\mathbf{W}^{(r)}, \tag{3.3.38}$$

where $\mathbf{G}$ is a block-diagonal preconditioner, whose diagonal blocks can be precomputed as a dense matrix for the reference configuration using the eigenvalue decomposition,

$$\mathbf{M}_{\text{ref}}^{(r)} = \mathbf{V}\boldsymbol{\Sigma}\mathbf{V}^{*}, \tag{3.3.39}$$

and we set

$$\mathbf{G}_{\text{ref}} = (\mathbf{\Sigma}^{\dagger})^{1/2}\mathbf{V}^{*} \quad \text{and} \quad \mathbf{G}^{\dagger}_{\text{ref}} = \mathbf{V}\mathbf{\Sigma}^{1/2}, \tag{3.3.40}$$

where the diagonal elements of $\mathbf{\Sigma}$ and $\mathbf{\Sigma}^{\dagger}$ corresponding to the spurious eigenvalues of $\mathbf{\Sigma}$ are set to be zero. This gives the desired discrete fluctuation-dissipation balance,

$$\left\langle \breve{\mathbf{v}}^{(r)}_{\text{ref}} \; \breve{\mathbf{v}}^{(r)}_{\text{ref}} \right\rangle = \mathbf{G}^{\dagger}_{\text{ref}}\mathbf{G}_{\text{ref}}\mathbf{M}^{(r)}\mathbf{G}^{\top}_{\text{ref}}\left(\mathbf{G}^{\dagger}_{\text{ref}}\right)^{\top} = \mathbf{M}^{(r)}_{\text{ref}}. \tag{3.3.41}$$

Generating $\breve{\mathbf{v}}^{(r)}$ as in Eq. (3.3.38) is efficient, since we can reuse the precomputed matrices in Eq. (3.3.40) throughout the simulation, and apply rotation matrices for each body,

$$\mathbf{G}_{\beta\beta} = \mathbf{R}_{\beta}\mathbf{G}_{\text{ref}}\mathbf{R}^{\top}_{\beta}. \tag{3.3.42}$$

We remark that the use of block-diagonal preconditioners does not increase the overall complexity of FBIM since they can be applied with $O(N_b)$ work, and, furthermore, the cost is amortized over the length of the BD simulations.

3.4 Numerical Results

This section presents numerical results of applying the FBIM to a number of benchmark problems in two dimensions. We first address the effect of the Ewald splitting parameter ξ on the accuracy of the first-kind mobility solver. We then test the first-kind mobility solver by applying it to the steady Stokes flow through a square periodic array of disks, and compare the results to well-known analytical solutions. In the second test, we consider suspensions of Brownian rigid disks, and assess the effectiveness of FBIM by its accuracy and convergence, the robustness of iterative solvers, and its scalability to simulate suspensions of many-body particle systems. In the last set of numerical examples, we perform *Brownian Dynamics*

(BD) simulations by combining the FBIM with previously-developed stochastic temporal integrators [103, 82]. By simulating the free diffusion of a non-spherical (starfish-shaped) particle, we confirm that correctly handling the stochastic drift term in the temporal integrator is necessary in order to reproduce the equilibrium Gibbs-Boltzmann distribution. As a simple but nontrivial benchmark problem for many-body suspension, we investigate the dynamics of a pair of starfish-shaped particles connected by a harmonic spring, which includes interaction through the spring potential, hydrodynamic interaction between the particles and with their periodic images, as well as Brownian noises.

3.4.1 Choosing the Ewald splitting parameter

The first important aspect of FBIM that needs to be addressed is how the choice of ξ affects the accuracy of the first-kind mobility solver. In the *Spectral Ewald* (SE) method [77] the choice of ξ is only based on balancing the computational work between the real- and Fourier-space sums. In practice, we want to choose a sufficiently large ξ (i.e., a short-ranged singular kernel $\mathbb{G}_\xi^{(r)}$), so that the computational work in the real-space can be made cheap at the expense of the FFT in Fourier space. In this work, we demonstrate that the choice of ξ is also limited by the accuracy of Alpert quadrature used to resolve the logarithmic singularity of $\mathbb{G}$ (diagonal elements of $\mathbf{M}$).

The Alpert quadrature can be viewed as assigning (interpolated) local correction weights to nearby quadrature nodes of a target point $\mathbf{x}_t$, as illustrated in the left panel of Fig. 3.1. For example, the number of (one-sided) quadrature nodes with nonzero correction weights is 4 and 8 for the 4^{th}- and 8^{th}-order Alpert quadrature, respectively. Since $\mathbb{G}_\xi^{(r)}$ decays exponentially with length scale ξ^{-1}, the Alpert quadrature grid must resolve length scales smaller than ξ^{-1} in order to capture the logarithmic singularity of $\mathbb{G}_\xi^{(r)}$ at the origin. Thus for a fixed grid, as ξ increases, we expect the accuracy of the Alpert quadrature to become progressively worse.

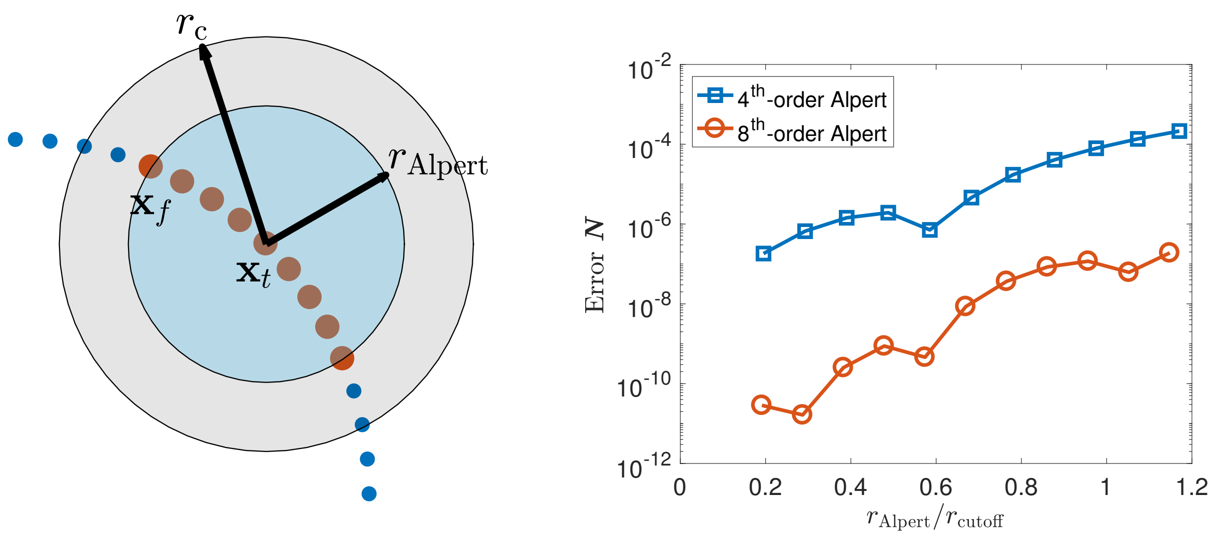


Fig. 3.1: (*Left panel*) A cartoon illustration of radius of Alpert correction r_{Alpert} and the cut-off radius of Ewald summation r_c. Both blue and red nodes represent the quadrature nodes in the trapzoidal rule, and the red nodes are the special nodes with Alpert correction weights centered a target node $\mathbf{x}_t$. The radius of Alpert correction is defined by $r_{\text{Alpert}} = |\mathbf{x}_t - \mathbf{x}_f|$, where $\mathbf{x}_f$ is the last node with Alpert correction weight for a chosen order of Alpert quadrature. (*Right panel*) Normalized error (in matrix 2-norm) of $\boldsymbol{N}$ with respect to $\boldsymbol{\mathcal{N}}$ (approximated with 12 digits of accuracy) versus the ratio r_{Alpert}/r_c for the 4^{th}- and 8^{th}-order Alpert quadratures. The mobility solver gradually loses accuracy because the singular kernel (with support r_c) is not sufficiently resolved by the Alpert quadrature as the ratio increases.

To demonstrate this issue, we study how the error of $\boldsymbol{N}$ changes with ξ as follows. We fix the packing fraction $\phi = \pi/16 \approx 0.196$ and $N_p = 64$ but use different values of ξ (i.e., different r_c). The radius of Alpert correction r_{Alpert} is defined by the distance between the target point $\mathbf{x}_t$ and the farthest quadrature node $\mathbf{x}_f$ with nonzero Alpert weight, i.e., $r_{\text{Alpert}} = |\mathbf{x}_t - \mathbf{x}_f|$. Note that r_{Alpert} is fixed once the quadrature order is fixed. The Ewald sum is computed with accuracy $\epsilon_{\text{Ewald}} = 10^{-9}$ and the tolerance level of GMRES is $\epsilon_{\text{tol}} = 10^{-9}$. The body mobility matrix $\boldsymbol{N}$ is computed by solving the deterministic mobility problem with force and torque $\boldsymbol{F}$ set to be the columns of the 3×3 identity matrix, which gives the corresponding columns of $\boldsymbol{N}$. We can also compute the exact body mobility matrix $\boldsymbol{\mathcal{N}}$ to 12 digits of accuracy with $N_p = 256$ by using the second-kind boundary integral formulation (see 3.7 for details). In two dimensions the resulting linear system of the second-kind formulation is well-conditioned and its solution is spectrally accurate. In the right panel of Fig. 3.1, we show the normalized error of $\boldsymbol{N}$ (in matrix 2-norm) with respect to the 12-digit accurate approximation of $\boldsymbol{\mathcal{N}}$ for the 4^{th}- and 8^{th}-order Alpert quadrature, and the error increases by at least two orders of magnitude for the range of r_{Alpert}/r_c considered. We conclude that choosing ξ with $r_{\text{Alpert}}/r_c \lesssim 0.6$ maintains the accuracy of Alpert quadrature sufficiently well.

3.4.2 Square lattice of disks

Steady Stokes flow around a square periodic array of fixed disks in two dimensions is one of the classical problems in fluid mechanics, and its analytic solution is a thoroughly studied topic in the literature. Notably, Hasimoto [86] obtained an analytical expression for the drag force F on a dilute array of disks moving with velocity U by solving the steady Stokes equations with Fourier series expansions. Later, Sangani and Acrivos [110] extended Hasimoto's solution to the semi-dilute regime by including higher-order correction terms, and obtained the expansion

$$\frac{F}{\eta U} = \frac{4\pi}{-\ln\sqrt{\phi} - 0.738 + \phi - 0.887\phi^2 + 2.039\phi^3 + O(\phi^4)}, \tag{3.4.1}$$

where a is the radius of disk, l is the spacing of the square lattice of disks, and $\phi = \pi a^2/l^2$ is the packing fraction. Another important regime for which there are theoretical solutions is the densely-packed regime. In this regime, the disks almost touch, so that there is flow through a narrow gap between two neighboring disks, and a lubrication correction is generally required to extend the solution to this regime. Sangani and Acrivos [111] obtained using lubrication theory the asymptotic formula

$$\frac{F}{\eta U} \approx \frac{9\pi}{2\sqrt{2}}\varepsilon^{-\frac{5}{2}} \tag{3.4.2}$$

where $\varepsilon = (l - 2a)/l = 1 - \sqrt{4\phi/\pi}$ is the relative gap between two neighboring disks.

In the following test, we numerically solve the Stokes mobility problem (3.2.23) for the rigid body motion $\boldsymbol{U}$ with applied force and torque $\boldsymbol{F} = (1, 0, 0)$ to determine the relationship between the drag force and velocity. We set $a = 1.0$ and $\epsilon_{\text{Ewald}} = \epsilon_{\text{tol}} = 10^{-9}$, while varying the length of square lattice l to achieve different packing fractions. For this simple test problem, we do not seek to optimize code performance, and we set $N_{\text{box}} = 4^2$ ($\xi \approx 20.13$) for all the test cases. In Sec. 3.4.3 we will address the choice of optimal ξ in the example of dense suspension of rigid particles. For a dilute or semi-dilute suspension ($\phi < 0.2$), the boundary Γ is discretized with $N_p = 64$ quadrature nodes, which is a sufficient resolution for the first-kind solver to give at least 6 digits of accuracy. For a higher packing fraction ($\phi > 0.2$), the number of quadrature nodes is determined by the ratio of the smallest gap between any two neighboring disks $d_g = l - 2a$ to the spacing between quadrature nodes $d_s = 2\pi a/N_p$. In a moderately-resolved computation, we require that the quadrature node

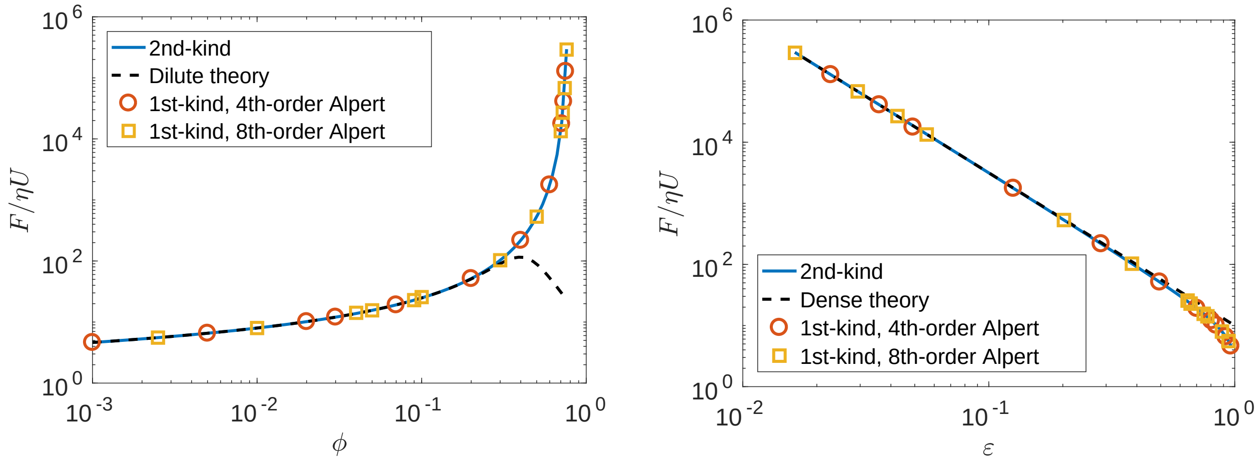


Fig. 3.2: The drag coefficients for a square periodic array of disks in steady Stokes flow, (*Left panel*) as a function of packing fraction ϕ, and compared to the dilute theory (3.4.1); (*Right panel*) as a function of the relative gap between disks ϵ, and compared to the dense theory (3.4.2). In both panels, the numerical results obtained from the first-kind solver with Alpert quadrature match very well with the spectrally-accurate results from the second-kind solver (see 3.7).

spacing is comparable or smaller than the gap spacing, i.e., $d_s/d_g \lesssim 1$. For example, for the highest packing fraction considered in this test $\phi = 0.76$ (note that for a close-packed square lattice $\phi_{\max} \approx 0.7854$), the number of quadrature nodes set by the ratio $d_s/d_g = 1$ is $N_p \approx 190$.

The linear system (3.3.8) is solved by GMRES with block-diagonal preconditioning. Numerical results for the normalized drag force over a broad range of packing fractions are shown in Fig. 3.2. We have obtained very good agreement with the dilute theory (3.4.1) ($\phi < 0.2$). For the dense packing fractions, the dilute theory no longer provides a good description for the drag, but our solutions from the first-kind solver match the dense theory (3.4.2) very well, as shown in the right panel of Fig. 3.2. The solutions from the first-kind solver are also in excellent agreement with the highly-accurate solutions (at least 9 digits of accuracy) from the second-kind solver for the range of packing fractions considered.

3.4.3 Suspension of Brownian rigid disks

In this section our primary goal is to study the performance of FBIM applying to suspensions of Brownian rigid particles in two dimensions. We will assess the effectiveness of FBIM from three aspects: accuracy and convergence, robustness of iterative solvers, and its efficiency and scalability to many-body particle systems. For simplicity, we focus on suspensions of disks only, however, there is no significant difficulty in applying the FBIM to more general bodies. In the top panel of Fig. 3.3, we show two random configurations of $N_b = 100$ disks with different packing fractions: a dilute suspension with $\phi = 0.25$ and a dense suspension with $\phi = 0.5$. These configurations are generated by using an event-driven molecular dynamics code. Since the random disks may nearly touch in the absence of (electrostatic) repulsive forces, especially when the packing fraction is high, we first generate a random configuration of disks with radius a_0 at a higher packing fraction ϕ_0, and then adjust the actual radius of disks a to achieve the desired packing fraction ϕ. In this approach, the pairs of random disks are separated by a relative minimum distance $d_{\min}/a = 2\left(\sqrt{\phi_0/\phi} - 1\right)$. For the dilute suspension ($\phi = 0.25$), we use $\phi_0 = 0.4$ so that $d_{\min}/a \approx 0.523$, and for the dense suspension ($\phi = 0.5$), we use $\phi_0 = 0.6$ so that $d_{\min}/a \approx 0.191$.

First, we investigate the accuracy and convergence of the first-kind mobility solver by applying it to the random configurations of disks shown in Fig. 3.3, subject to random forces and torques $\boldsymbol{F}$ (without random surface velocity $\breve{\boldsymbol{v}}$). In the first-kind mobility solver, we set $\epsilon_{\text{Ewald}} = \epsilon_{\text{tol}} = 10^{-9}$, and choose the splitting parameter $\xi \approx 50$ (or $N_{\text{box}} = 10^2$). Although this value of ξ does not achieve the minimum CPU time (see Fig. 3.5), it ensures $r_{\text{Alpert}}/r_c \lesssim 0.5$ for the 8^{th}-order Alpert quadrature, so that the singularity is sufficiently resolved in all test cases for convergence study purpose. For this set of computations, we consider different numbers of quadrature nodes, or degrees of freedom (DOFs) per body, $N_p \in \{16, 32, 64, 128\}$. The normalized error of $\boldsymbol{U} = \boldsymbol{N}\boldsymbol{F}$ is computed with respect to a

12-digit accurate solution computed from the second-kind solver with 256 DOFs per disk. For the dilute suspension (bottom left panel of Fig. 3.3), the second-kind solver converges spectrally fast and is more accurate than the solutions from the first-kind solver. For the dense suspension (bottom right panel of Fig. 3.3), while the second-kind solver converges faster and gives more accurate solutions for large number of DOFs per body, the first-kind solver is more accurate for low resolutions. This can be explained as follows. When two disks nearly touch, the singular kernel of the first-kind integral (Stokeslet) grows as $\log r$, but the singular kernel of the second-kind integral (stresslet) grows as r^{-1} in two dimensions. This observation implies that the first-kind mobility solver is more practical than the second-kind solver for simulating suspensions of rigid particles with high packing fractions, since it can produce sufficiently accurate solutions with a smaller number of DOFs per body.

Next, we study the robustness of iterative methods for solving the saddle-point linear system (3.3.8). For the random configurations shown in Fig. 3.3, in addition to random forces and torques, we also generate random surface velocity $\breve{\mathbf{v}} = \mathbf{M}^{1/2}\mathbf{W}$, and include it on the right-hand-side of Eq. (3.3.8). We show in the inset of Fig. 3.4 the residual versus the number of GMRES iterations for solving Eq. (3.3.8) with block-diagonal preconditioning. It would take more than 3 times the number of GMRES iterations to converge to the same tolerance level without preconditioning. In general, when the packing fraction grows or when two disks get closer, the hydrodynamic interaction between the disks becomes stronger, and hence, the condition number of the linear system grows. As a result, it requires more GMRES iterations for the dense suspensions, as expected. The number of GMRES iteration is independent of ξ, since the choice of ξ does not change $\mathbf{M} = \mathbf{M}^{(r)} + \mathbf{M}^{(w)}$ in the linear system (3.3.8). We expect that GMRES will converge faster for three-dimensional problems because the Stokeslet decays faster (r^{-1} in the far field) in three dimensions. We also show in Fig. 3.4 the residual versus the number of Lanczos iterations for generating $(\mathbf{M}^{(r)})^{1/2}\mathbf{W}^{(r)}$ with block-diagonal preconditioning, for different values of ξ. The number of Lanczos iterations decreases with ξ

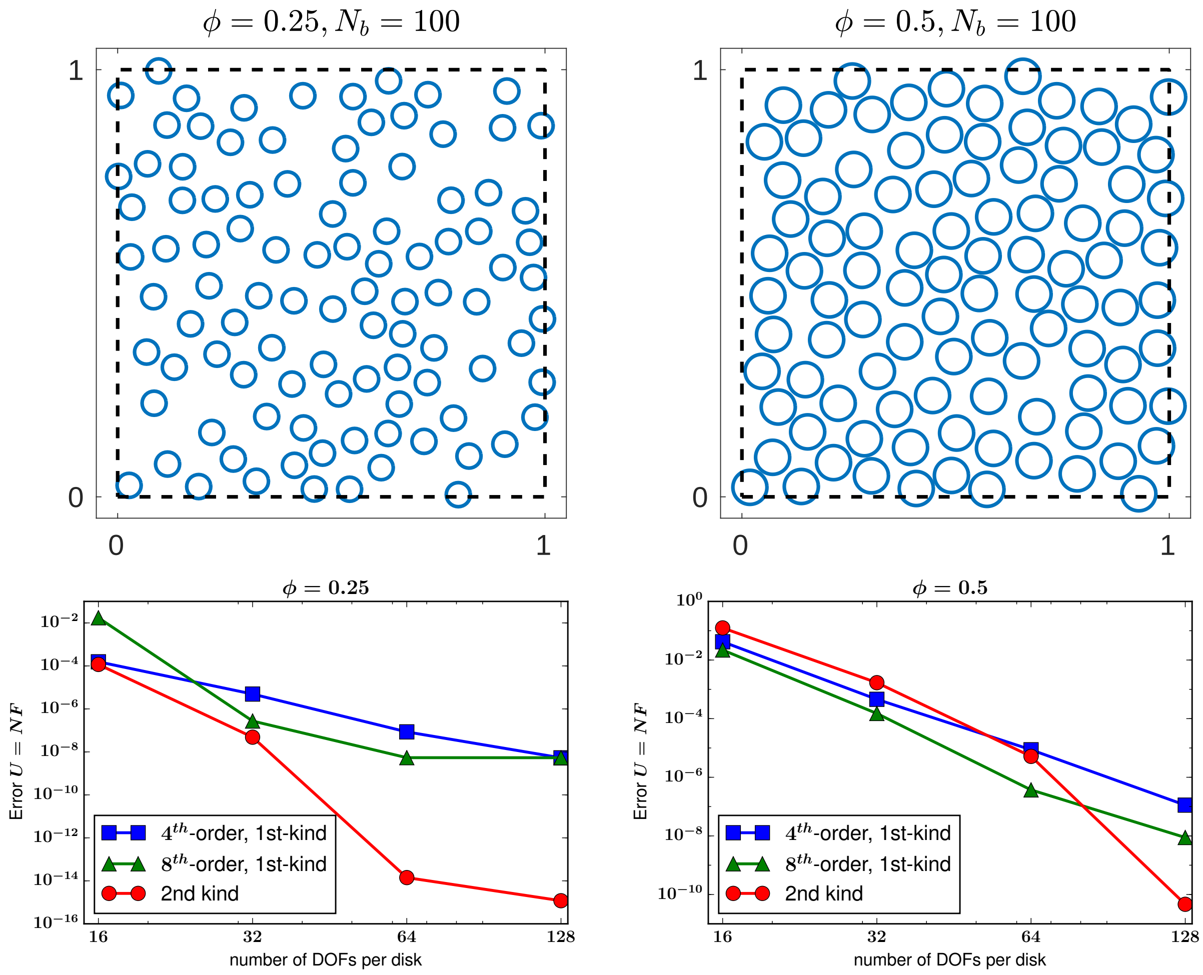
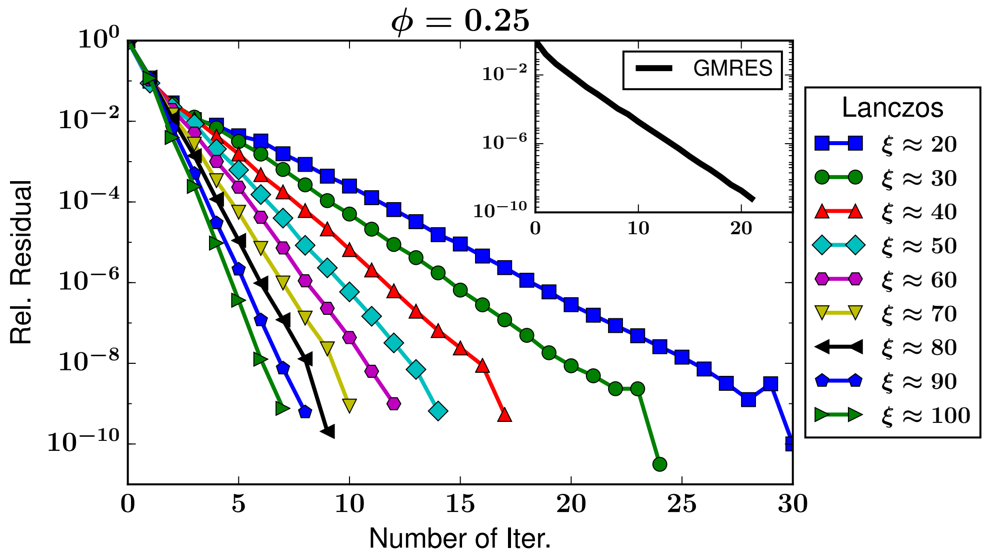


Fig. 3.3: (*Top panel*) Two random configurations of 100 rigid disks with packing fractions $\phi = 0.25$ (dilute) and $\phi = 0.5$ (dense) in a periodic unit cell. (*Bottom panel*) Normalized error of the mobility $\boldsymbol{U} = \boldsymbol{NF}$ versus number of degrees of freedom (DOFs) per disk. For the dilute suspension, the second-kind solver converges spectrally fast and is generally more accurate than the first-kind solver. For the dense suspension, while the second-kind solver converges faster, the first-kind solver is actually more accurate at lower DOFs per disk.

for both the dilute and dense suspensions. This can be explained by the observation that $\mathbb{G}_\xi^{(r)}$ becomes more short-ranged as ξ increases, and therefore, the block-diagonal preconditioner gets progressively better in approximating the inverse of $(\mathbf{M}^{(r)})^{1/2}$.

We now present profiling and scaling analysis of FBIM. First, we present the profiling results of FBIM and its algorithmic components in Fig. 3.5, and analyze the optimal choice of the splitting parameter ξ. We focus on the densely-packed configuration with 64 DOFs per disk ($d_s/d_g \approx 0.5$), and use the 4^{th}-order Alpert quadrature. The accuracy of the first-kind mobility solver with these parameters is about 10^{-5} (see bottom right panel of Fig. 3.3), so we set $\epsilon_{\text{Ewald}} = \epsilon_{\text{tol}} = 10^{-6}$. We note that the execution time depends heavily on the choice of programming language, implementation and hardware. Our proof-of-concept serial implementation of FBIM in two dimensions is written in MATLAB with some subroutines accelerated by C with the aid of MEX files. As previously discussed in Sec. 3.3, the main ingredients of FBIM are evaluating the matrix-vector products $\mathbf{M}^{(r)}\boldsymbol{\mu}$ and $\mathbf{M}^{(w)}\boldsymbol{\mu}$ in GMRES, and generating $(\mathbf{M}^{(r)})^{1/2}\mathbf{W}^{(r)}$ and $(\mathbf{M}^{(w)})^{1/2}\mathbf{W}^{(w)}$ using the Lanczos iteration. In our implementation, we found it optimal to export and store $\mathbf{M}^{(r)}$ sparsely for rapid matrix-vector multiplication in Lanczos and GMRES. In our profiling analysis, we profile the time to export $\mathbf{M}^{(r)}$ sparsely, the cumulative time to evaluate $\mathbf{M}^{(r)}\boldsymbol{\mu}$ and $\mathbf{M}^{(w)}\boldsymbol{\mu}$ in GMRES separately, the time to generate $(\mathbf{M}^{(r)})^{1/2}\mathbf{W}^{(r)}$ and $(\mathbf{M}^{(w)})^{1/2}\mathbf{W}^{(w)}$ separately, and the total execution time of FBIM. We note that the total execution time of FBIM also includes the time of applying the preconditioners and other overhead time. The left panel of Fig. 3.5 shows the profiling results of the densely-packed configuration for different values of ξ. First, we observe that our implementation of FBIM is dominated by two subroutines: exporting $\mathbf{M}^{(r)}$ and evaluating $\mathbf{M}^{(w)}\boldsymbol{\mu}$ (iteratively). The CPU time of the sparse matrix-vector product $\mathbf{M}^{(r)}\boldsymbol{\mu}$ and generating $(\mathbf{M}^{(w)})^{1/2}\mathbf{W}^{(w)}$ (non-iteratively) is negligible. The CPU time for generating $(\mathbf{M}^{(r)})^{1/2}\mathbf{W}^{(r)}$ is also small because of the rapid matrix-vector multiplication of the sparse matrix $\mathbf{M}^{(r)}$. The CPU time for generating $(\mathbf{M}^{(r)})^{1/2}\mathbf{W}^{(r)}$ also includes the time for applying



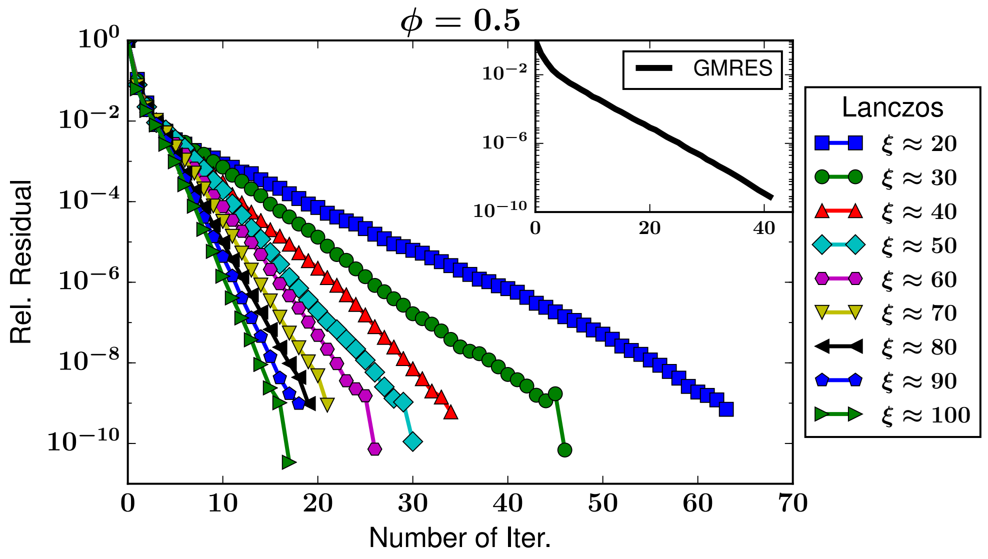


Fig. 3.4: Convergence of the Lanczos iteration for generating $(\mathbf{M}^{(r)})^{1/2}\mathbf{W}^{(r)}$ and the GMRES iteration (inset) for solving the saddle-point linear system Eq. (3.3.8) with block-diagonal preconditioning for both iterative solvers. Generally, it requires more GMRES iterations for the dense packing ($\phi = 0.5$), as shown in the inset. The number of GMRES iteration does not depend on ξ, since the choice of ξ does not change $\mathbf{M}$ in the linear system (3.3.8). The number of Lanczos iteration decreases with ξ, since the real-space kernel $\mathbb{G}_{\xi}^{(r)}$ becomes more short-ranged as ξ increases, and therefore, the block-diagonal preconditioner gets progressively better in approximating the inverse of $(\mathbf{M}^{(r)})^{1/2}$.

the preconditioner, which accounts for about 20% - 30% of the computational work.

Generally, the execution time of the real-space subroutines decreases with ξ, because the amount of work in the real space reduces as $\mathbb{G}_{\xi}^{(r)}$ becomes more short-ranged with ξ. On the other hand, the execution time of Fourier-space subroutines remains almost constant for the range of ξ considered in this test. This is because the cost of grid operations in Fourier-space sums (spreading/interpolation of a Gaussian to grid in NUFFT) dominates the cost of FFT in two dimensions. We expect the FFT cost would eventually become dominant in three dimensions. In our implementation, we observe that even the total CPU time of FBIM decreases with ξ, whereas Fiore *et al.* [57] report an optimal ξ for their GPU implementation of the PSE method in three dimensions. There is a wide range of ξ that approximately minimizes the total CPU time (from $\xi \approx 70$ to $\xi \approx 100$), as shown in the left panel of Fig. 3.5. We recall, however, that the choice of ξ in FBIM is also limited by the accuracy of Alpert quadrature. Using the criterion that $r_{\text{Alpert}}/r_c \lesssim 0.6$ for the densely-packed configuration, we obtain that $\xi \lesssim 150$ for the 4^{th}-order Alpert quadrature, and that $\xi \lesssim 75$ for the 8^{th}-order Alpert quadrature. Similar optimal range of ξ is also obtained in the profiling analysis for the dilute configuration.

Another important computational aspect that needs to be addressed is how the FBIM scales as the number of rigid particles grows while the packing fraction is held fixed. This aspect of FBIM is particularly important for applications involving a large number of particles. In the right panel of Fig. 3.5, we report the total execution time of FBIM with increasing number of rigid disks, while the packing fraction is held fixed at $\phi = 0.25$ and $\phi = 0.5$, and ξ is also fixed ($\xi \approx 70$) for both configurations. We conclude that FBIM scales linearly in the number of rigid particles for both dilute and dense suspensions.

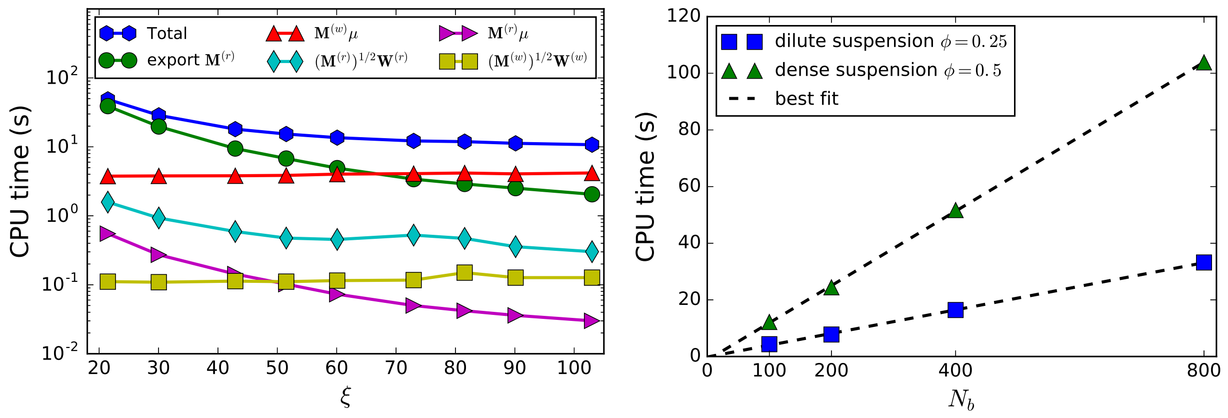


Fig. 3.5: (*Left panel*) Execution time of FBIM and its algorithmic components versus ξ for the dense packing $\phi = 0.5$. In our implementation, the computational work of FBIM is dominated by the subroutines for exporting $\mathbf{M}^{(r)}$ sparsely and for evaluating the matrix-vector product $\mathbf{M}^{(w)}\boldsymbol{\mu}$ (iteratively in GMRES). The total execution time of FBIM decreases with ξ, and the minimum CPU time is achieved near $\xi \approx 70$, where the CPU time for exporting $\mathbf{M}^{(r)}$ and evaluating $\mathbf{M}^{(w)}\boldsymbol{\mu}$ crosses. (*Right panel*) Linear scaling of FBIM with growing number of disks in the suspension, while the packing fraction is held fixed.

3.4.4 Brownian dynamics of rigid particles

In the following tests, we combine the FBIM with stochastic temporal integrators to perform Brownian Dynamics (BD) for rigid particles. Applying the weakly first-order accurate Euler-Maruyama (EM) scheme to Eq. (3.2.6), we obtain the BD algorithm,

$$\begin{aligned}\boldsymbol{Q}^{n+1} = \boldsymbol{Q}^n + \Delta t \boldsymbol{N}^n \boldsymbol{F}^n + \sqrt{2k_BT\Delta t}\,(\boldsymbol{N}^n)^{\frac{1}{2}}\mathbf{W}^n \\ + \Delta t \frac{k_BT}{\delta}\left[\boldsymbol{N}\left(\boldsymbol{Q}^n + \frac{\delta}{2}\widetilde{\mathbf{W}}^n\right)\widetilde{\mathbf{W}}^n - \boldsymbol{N}\left(\boldsymbol{Q}^n - \frac{\delta}{2}\widetilde{\mathbf{W}}^n\right)\widetilde{\mathbf{W}}^n\right],\end{aligned} \tag{3.4.3}$$

where Δt is the time step size, the superscript denotes the time step level at which quantities are evaluated (e.g., $\boldsymbol{Q}^n = \boldsymbol{Q}(t = n\Delta t)$ and $\boldsymbol{N}^n = \boldsymbol{N}(\boldsymbol{Q}^n)$), δ is a small parameter, and $\mathbf{W}^n$ and $\widetilde{\mathbf{W}}^n$ are uncorrelated vectors of i.i.d standard Gaussian random variables.

The last term in Eq. (3.4.3) is a centered *random finite difference* (RFD) approximation to the stochastic drift term that is equal in expectation to $(\Delta t\, k_BT)(\partial_{\boldsymbol{Q}} \cdot \boldsymbol{\mathcal{N}})^n$ for sufficiently small δ [82, 94, 95]. The RFD term guarantees that the EM scheme is a consistent stochastic integrator of Eq. (3.2.6), but is simpler and more efficient in practice than the Fixman midpoint scheme [102], in which the action of $\boldsymbol{N}^{-\frac{1}{2}}$ is required. The choice of δ is determined by a balance between truncation and roundoff error in the centered RFD, which gives $\delta/a \sim \epsilon^{1/3}$, where ϵ is the accuracy of the matrix-vector product $\boldsymbol{N}\boldsymbol{F}$ and a is the characteristic length of the particle. We note that the EM scheme requires solving the saddle-point linear system (3.3.8) three times: once for generating the velocity $\boldsymbol{N}^n\boldsymbol{F}^n + \sqrt{2k_BT/\Delta t}\,(\boldsymbol{N}^n)^{\frac{1}{2}}\mathbf{W}^n$, and twice for the RFD approximation.

The EM scheme is not particularly accurate even for the deterministic motion. Another weakly first-order accurate temporal integrator that has been observed to give a better accuracy is the stochastic Adams-Bashforth (AB) scheme [95],

$$\begin{aligned}\boldsymbol{Q}^{n+1} = \boldsymbol{Q}^n &+ \Delta t\left(\frac{3}{2}\boldsymbol{N}^n\boldsymbol{F}^n - \frac{1}{2}\boldsymbol{N}^{n-1}\boldsymbol{F}^{n-1}\right) + \sqrt{2k_BT\Delta t}\,(\boldsymbol{N}^n)^{\frac{1}{2}}\mathbf{W}^n \\ &+ \Delta t\frac{k_BT}{\delta}\left[\boldsymbol{N}\left(\boldsymbol{Q}^n + \frac{\delta}{2}\widetilde{\mathbf{W}}^n\right)\widetilde{\mathbf{W}}^n - \boldsymbol{N}\left(\boldsymbol{Q}^n - \frac{\delta}{2}\widetilde{\mathbf{W}}^n\right)\widetilde{\mathbf{W}}^n\right],\end{aligned} \tag{3.4.4}$$

in which the deterministic mobility $\boldsymbol{\mathcal{N}}\boldsymbol{F}$ in Eq. (3.2.6) is approximated by the second-order Adams-Bashforth approximation. We observe, however, that the AB scheme is more expensive to use in practice, because it requires four mobility problem solves instead of three in the EM scheme. More efficient schemes can be developed but are not the focus of our work [104].

3.4.4.1 Free diffusion of a single starfish

We consider a starfish-shaped particle freely diffusing (i.e., no applied force and torque) in a periodic unit lattice. A similar benchmark problem was studied by Delong *et al.* [82] using the FIB method, and by Delmotte *et al.* [84] using the FCM method for a single spherical particle in a periodic domain. In their case the body mobility matrix does not depend on the position of the particle, and therefore $\partial_{\boldsymbol{Q}} \cdot \boldsymbol{\mathcal{N}} = 0$. However, for a starfish $\boldsymbol{\mathcal{N}}$ depends on orientation due to interactions with periodic images and $\partial_{\boldsymbol{Q}} \cdot \boldsymbol{\mathcal{N}}$ is nonzero.

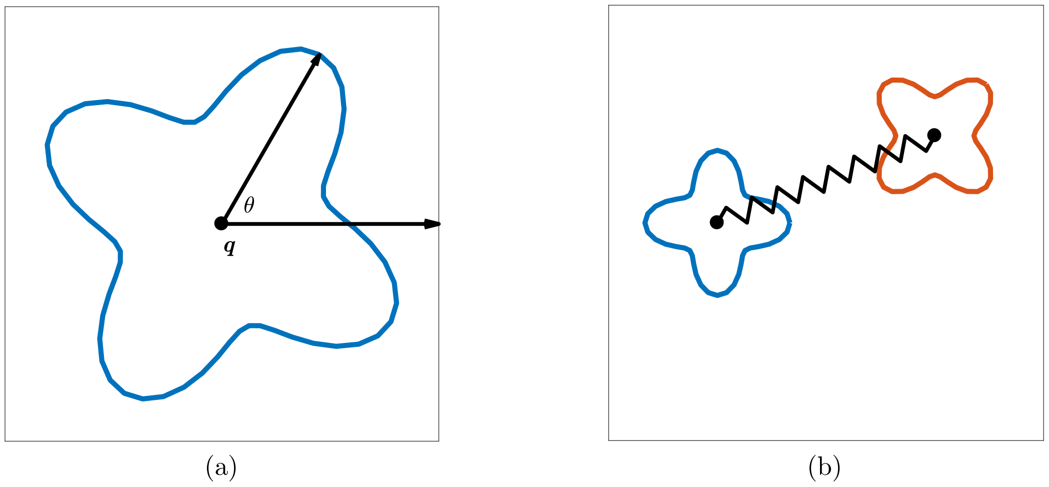


Fig. 3.6: (*Left panel*) A single starfish particle with four-fold rotational symmetry, described by the position of tracking point $\boldsymbol{q}$ and its rotation θ. (*Right panel*) Equilibrium position for a pair of starfish particles connected by a harmonic spring. The enclosing box is the periodic unit cell.

The starfish particle has four-fold rotational symmetry (left panel of Fig. 3.6), and is described by

$$\Gamma : (x(s), y(s)) = r_s(1 + b\cos(4s)) \cdot (\cos s, \sin s), \quad s \in [0, 2\pi], \tag{3.4.5}$$

where the characteristic length of the starfish particle is its maximum radius $a = r_s(1 + b)$. In the continuum setting, the body mobility matrix of the four-fold starfish particle is a diagonal matrix and depends only on the rotation θ, i.e.,

$$\boldsymbol{\mathcal{N}}(\theta) = \begin{bmatrix} \mathcal{N}_{xx}(\theta) & & \\ & \mathcal{N}_{yy}(\theta) & \\ & & \mathcal{N}_{\theta\theta}(\theta) \end{bmatrix}, \tag{3.4.6}$$

where $\mathcal{N}_{xx}$, $\mathcal{N}_{yy}$ and $\mathcal{N}_{\theta\theta}$ are the rotational and translational self-mobilities. Because of the symmetry of the starfish particle, we also have $\mathcal{N}_{xx} = \mathcal{N}_{yy}$. The elements of $\boldsymbol{\mathcal{N}}(\theta)$ can be computed to 12 digits of accuracy using the second-kind solver by solving the deterministic mobility problem with $\boldsymbol{F}$ set to be the columns of the identity matrix for different θ.

Applying the EM scheme without RFD to the freely-diffusing starfish particle, we obtain the biased scheme,

$$\boldsymbol{Q}^{n+1} = \boldsymbol{Q}^n + \sqrt{2k_BT\Delta t}(\boldsymbol{N}^n)^{\frac{1}{2}}\mathbf{W}^n. \tag{3.4.7}$$

In the limit $\Delta t \to 0$, the biased scheme Eq. (3.4.7) is consistent with the Ito SDE

$$\begin{aligned}\frac{d\boldsymbol{Q}}{dt} &= \sqrt{2k_BT\Delta t}\,(\boldsymbol{\mathcal{N}}(\theta))^{\frac{1}{2}}\,\boldsymbol{\mathcal{W}}(t) \\ &= -\boldsymbol{\mathcal{N}}(\theta)(\partial_{\boldsymbol{Q}}\widetilde{U}) + \sqrt{2k_BT\Delta t}\,(\boldsymbol{\mathcal{N}}(\theta))^{\frac{1}{2}}\,\boldsymbol{\mathcal{W}}(t) + (k_BT)\partial_{\boldsymbol{Q}}\cdot(\boldsymbol{\mathcal{N}}(\theta)),\end{aligned} \tag{3.4.8}$$

where the bias potential is $\widetilde{U}\left(\boldsymbol{Q} = \{\boldsymbol{q}, \theta\}\right) = k_BT\log(\mathcal{N}_{\theta\theta})$. By examining the corresponding Fokker-Plank equation, we can show that the biased SDE (3.4.8) preserves the biased equilibrium distribution

$$\widetilde{P}_{eq}(\boldsymbol{Q}) \equiv \widetilde{P}_{eq}(\theta) = Z^{-1}\exp\left(-\widetilde{U}(\theta)/k_BT\right) = \left(Z\mathcal{N}_{\theta\theta}\right)^{-1}. \tag{3.4.9}$$

We note that the correct Gibbs-Boltzmann distribution preserved by the unbiased scheme (EM with RFD) is a uniform distribution $P_{eq}(\boldsymbol{Q}) =$ constant.

In our computation, we set $b = 0.3$ and $a = 1.3r_s = 0.45$, which gives a relatively high packing fraction $\phi \approx 0.393$ for the starfish particle, in order to amplify the difference between the biased distribution $\widetilde{P}_{eq}$ and the correct Gibbs-Boltzmann distribution P_{eq} (see left panel of Fig. 3.7). The starfish particle is discretized by $N_p = 64$ quadrature points using the 4^{th}-order Alpert quadrature. In the first-kind mobility solver, we set the error tolerance level $\epsilon = 10^{-7}$. The RFD parameter δ is set by $\delta/a \sim \epsilon^{1/3}$. The short-time translational χ_{trans} and rotational χ_{rot} diffusion coefficients are determined by the Stokes-Einstein relations,

$$\begin{aligned} \chi_{\text{trans}} &= k_B T \langle \mathcal{N}_{xx} \rangle \approx 2.47 \times 10^{-3}, \\ \chi_{\text{rot}} &= k_B T \langle \mathcal{N}_{\theta\theta} \rangle \approx 2.081 \times 10^{-1}, \end{aligned} \tag{3.4.10}$$

where the average $\langle \cdot \rangle$ is taken with respect to the equilibrium distribution. We use a small time step size $\Delta t = 0.02$ to minimize truncation errors, and each simulation is run for $T = 100$, where τ_{rot} is the rotational Brownian time scale.

Figure 3.7 shows the estimated biased and unbiased equilibrium distributions (with error-bars of 2 standard deviations) obtained from 600 independent trajectories of the starfish particle freely diffusing in a periodic unit lattice. The numerical results using EM without RFD indeed match the biased equilibrium distribution $\widetilde{P}_{eq}(\theta)$, whereas EM with RFD preserves the correct Gibbs-Boltzmann distribution at equilibrium. In the right panel of Fig. 3.7, we compare the translational mean square displacement (MSD) for the biased and unbiased schemes, and also compare to the Einstein formula

$$\left\langle ||\boldsymbol{q}(t+s) - \boldsymbol{q}(t)||^2 \right\rangle = 4 k_B T \langle \mathcal{N}_{xx} \rangle \, t, \tag{3.4.11}$$

where the average $\langle \cdot \rangle$ is taken with respect to the biased and unbiased equilibrium distribution, respectively. We conclude that consistent approximation to the stochastic drift term, such as the RFD approximation, is not only important for the equilibrium dynamics, but is also necessary for producing the short-time dynamics (MSD) correctly.

3.4.4.2 A pair of interacting starfish

In this test, we perform BD simulation of a pair of starfish particles connected by a Hookean spring with rest length l_s in a periodic square lattice of length $l = 2$ (right panel of Fig. 3.6). The two starfish particles interact through the spring connecting the two tracking

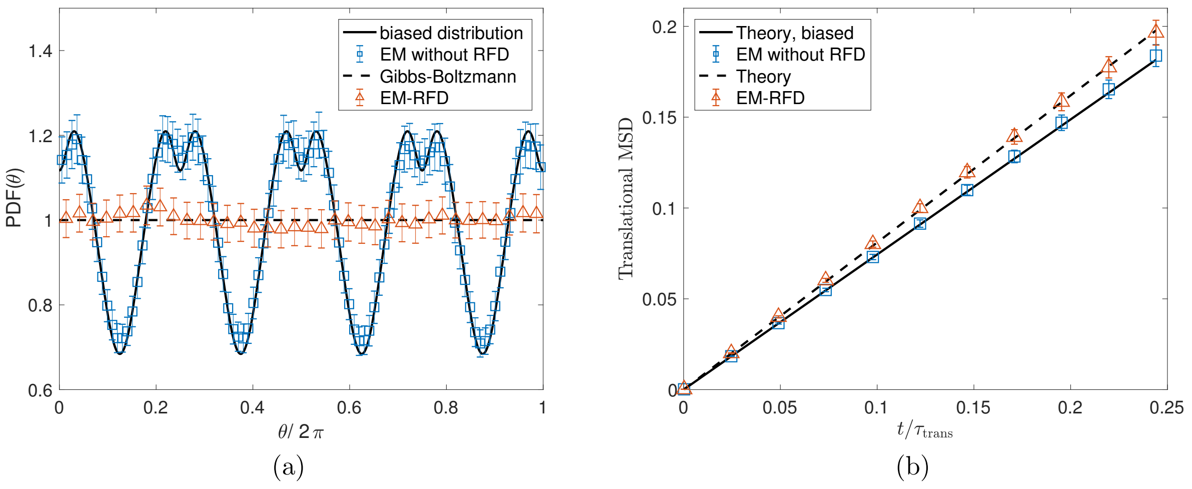


Fig. 3.7: (*Left panel*) Equilibrium probability distribution of a single starfish particle freely diffusing in a periodic square lattice. The EM scheme without RFD produces a biased equilibrium distribution, and matches the theory Eq. (3.4.9) (solid). The EM scheme with RFD produces the correct Gibbs-Boltzmann distribution (dashed). (*Right panel*) Translational mean square displacement (MSD) for the freely-diffusing starfish particle using the biased and unbiased EM schemes.

points $\boldsymbol{q}_1, \boldsymbol{q}_2$ via a potential U_{spring}, and the rotation of each particle is also attached to a preferred angle through a harmonic potential $U_{\text{rot}}(\theta)$. The total potential $U(\boldsymbol{Q})$ is given by

$$\begin{aligned} U(\boldsymbol{Q}) &= U(\boldsymbol{q}_1, \theta_1, \boldsymbol{q}_2, \theta_2) \\ &= U_{\text{spring}}(\boldsymbol{q}_1, \boldsymbol{q}_2) + U_{\text{rot}}(\theta_1) + U_{\text{rot}}(\theta_2) \\ &= \frac{k_s}{2}(|\boldsymbol{q}_1 - \boldsymbol{q}_2| - l_s)^2 + \frac{k_\theta}{2}\left(\theta_1 - \frac{\pi}{4}\right)^2 + \frac{k_\theta}{2}\left(\theta_2 - \frac{\pi}{2}\right)^2, \end{aligned} \tag{3.4.12}$$

where k_s, k_θ are the stiffness coefficients. In the equilibrium, the Gibbs-Boltzmann preserved by Eq. (3.2.6) with $\boldsymbol{F} = -\partial_{\boldsymbol{Q}} U$ is

$$\begin{aligned} P_{eq}(\boldsymbol{Q}) &\propto \exp\left(-U(\boldsymbol{Q})/k_B T\right) \\ &\propto \exp\left(-\frac{U_{\text{spring}}(\boldsymbol{q}_1, \boldsymbol{q}_2)}{k_B T}\right) \cdot \mathcal{N}\left(\frac{\pi}{2}, \frac{k_B T}{k_\theta}\right) \cdot \mathcal{N}\left(\frac{\pi}{4}, \frac{k_B T}{k_\theta}\right), \end{aligned} \tag{3.4.13}$$

where $\mathscr{N}(\mu, \sigma^2)$ denotes the normal distribution with mean μ and variance σ^2.

In the computation, each particle has size $a/l = 1.3\, r_s/l = 1/12$, and is discretized by $N_p = 48$ quadrature points using the 4^{th}-order Alpert quadrature. The spring rest length is $l_s = 5a$ and the stiffness coefficient k_s is set based on the criterion that 3 standard deviations of the distance $d = |\boldsymbol{q}_1 - \boldsymbol{q}_2|$ is approximately $2a$, i.e., $3\sqrt{k_B T/k_s} \approx 2a$, so that the particles very rarely overlap. The presence of harmonic springs defines the spring relaxation time scales,

$$\tau_s = \frac{1}{k_s \langle \mathcal{N}_{xx} \rangle} \quad \text{and} \quad \tau_\theta = \frac{1}{k_\theta \langle \mathcal{N}_{\theta\theta} \rangle}, \tag{3.4.14}$$

where $\mathcal{N}_{xx}$ and $\mathcal{N}_{\theta\theta}$ are the translational and rotational self-mobilities of a single particle, which can be computed with high accuracy using the second-kind solver. The value of the parameters are $k_s = 81$, $k_\theta \approx 2.446$, and $\tau_s = \tau_\theta \approx 0.1189$. Recall that the dominant computational work of FBIM is to compute the mobility $\boldsymbol{N}\boldsymbol{F}$ (see left panel of Fig. 3.5). To reduce the amount of computational work, we found that solving the mobility problems in RFD with a lower tolerance level $\epsilon = 10^{-3}$ while maintaining a higher tolerance $\epsilon = 10^{-6}$ in the deterministic mobility does not introduce any observable statistical errors in the computation.

In Fig. 3.8, we compare errors of the mean and covariance of θ_1, θ_2 and the distance $d = |\boldsymbol{q}_1 - \boldsymbol{q}_2|$ with respect to the equilibrium statistics calculated analytically from Eq. (3.4.13). The numerical results are generated from 16 independent trajectories with length $T \approx 14.86$. We observe in Fig. 3.8 that AB2 is more accurate than EM (see the panel that shows $\text{cov}(d, d)$). We also observe that the cross covariances are statistically indistinguishable from zero for sufficiently small time step sizes, indicating that the distance between particles and their rotations are uncorrelated, as expected from the equilibrium distribution Eq. (3.4.13).

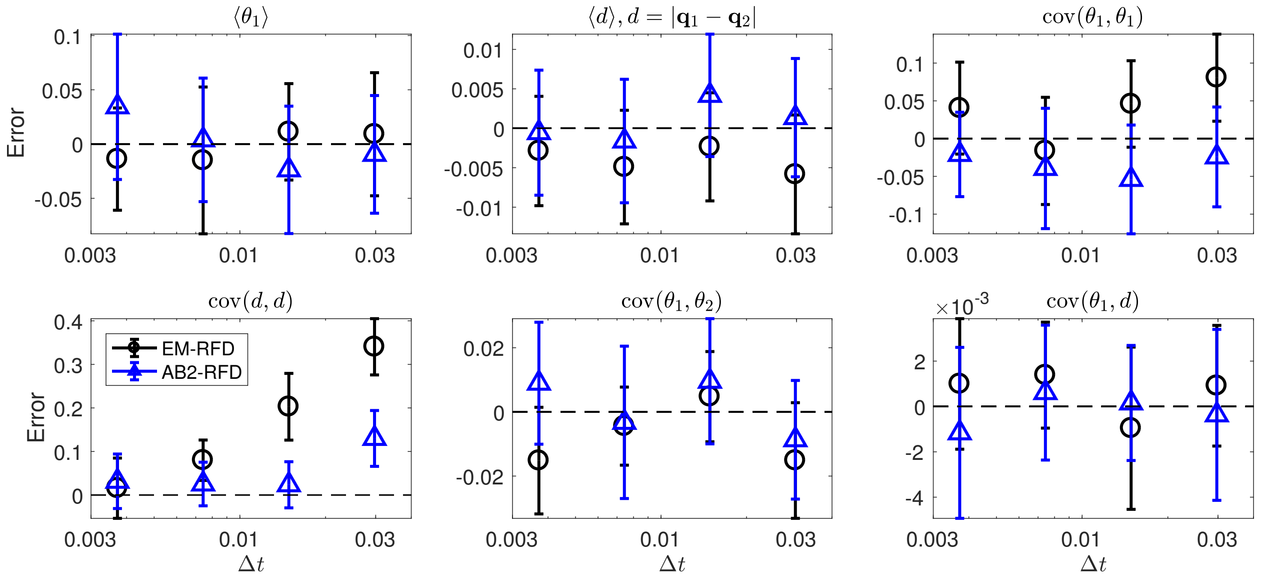


Fig. 3.8: Error bars (with two standard deviations) of the mean and covariance of distance between two tracking points $d = |\boldsymbol{q}_1 - \boldsymbol{q}_2|$ and rotations θ_1, θ_2 for a pair of starfish particles interacting with the potential Eq. (3.4.12). The numerical results are produced using the EM and AB schemes, with both including the RFD term. The AB scheme is more accurate than the EM scheme for $\mathrm{cov}(d, d)$. The cross covariances are statistically indistinguishable from zero, indicating the particle positions and their rotations are uncorrelated, as expected from the equilibrium distribution Eq. (3.4.13).

3.5 Conclusions

In this chapter we presented a fluctuating boundary integral method (FBIM) for simulating the overdamped Brownian Dynamics (BD) of rigid particles of complex shape in periodic domains. To the best of our knowledge, it is the first boundary integral method that accounts for Brownian motion of nonspherical particles immersed in a viscous incompressible fluid. Its main advantages are that particles of complex shape can be directly discretized with a surface mesh, and the deterministic mobility of the particles can be computed with high accuracy by using high-order singular quadrature techniques. Importantly, the Brownian displacements of the particles are computed along the way with only a marginal increase in the overall cost, and strictly satisfy discrete fluctuation-dissipation balance. To accomplish this, instead of adding a stochastic stress tensor to the fluid equations as done in fluctuating hydrodynamics, we eliminated the fluid in the spirit of boundary integral representations. This lead to a Stochastic Stokes Boundary Value Problem (SSBVP) in which we prescribed a random surface velocity distribution that has covariance proportional to the (periodic) Stokeslet. We found that using a first-kind boundary integral formulation is simplest since the first-kind integral operator inherits the SPD property from the Green's function for Stokes flow. The matrix discretizing the single-layer operator directly gives the covariance of the required fluctuating surface velocity, including a suitable handling of the singularity of the Oseen tensor. While formal analysis suggests that second-kind boundary integral methods are to be preferred over first-kind methods, we found that first-kind methods can be more accurate for dense suspensions, and showed that a simple block-diagonal preconditioner can effectively handle the ill-conditioning of the first-kind formulation. We confirmed through different benchmark problems that FBIM can efficiently compute both deterministic and Brownian motion of rigid particles in accordance with the order of accuracy of the singular quadrature scheme. Our preconditioned iterative solvers (GMRES and Lanczos) converged within a small number of iterations and only grew slowly with the packing fraction. We also confirmed that the computational cost of FBIM scales linearly with the number of particles, even for moderately dense packing fractions. Finally, we coupled FBIM with stochastic temporal integrators, and showed that it reproduced the correct equilibrium Gibbs-Boltzmann distribution of Brownian suspensions of particles of complex shape.

The FBIM presented in this work is a only a first step toward the overarching goal of performing accurate, efficient and robust BD simulations of a large collection of rigid particles of complex shape. Our proof-of-concept implementation of FBIM and numerical examples were presented in two spatial dimensions only. The continuum formulation of FBIM, which generates both deterministic and stochastic velocities in agreement with Eq. (3.2.20), applies directly to three dimensions. In principle, our discrete formulation of FBIM can also be extended to three spatial dimensions. This extension requires, however, developing a suitable quadrature rule for the single-layer potential. While in two dimensions we were able to use the trapezoidal rule as the underlying quadrature rule and apply Alpert corrections to account for the singularity of the Oseen tensor, in three dimensions these pieces need to be developed anew. First, a suitable discretization of the particles' surfaces (e.g., using higher-order triangular elements) and a suitable non-singular quadrature rule need to be developed. For special particles shapes, notably spheres or spheroids, one can use specially chosen surface grids with the trapezoidal rule [5, 6]; however, for general particle shapes it is not straightforward to achieve spectral accuracy. One of the desired properties of the quadrature rule is to ensure that the far-field component of the discrete single-layer matrix $\mathbf{M}$ is symmetric and positive definite. In particular, we expect that a good quadrature rule would yield the SPD matrix $\mathbf{M} = \mathbf{\Psi}\mathbf{G}\mathbf{\Psi}^T$, where $\mathbf{G}_{ij} = \mathbb{G}(\mathbf{x}_i, \mathbf{x}_j)$, and $\mathbf{\Psi}$ encodes the quadrature weights and mesh connectivity. Second, a singular quadrature near-field correction for the single-layer kernel needs to be constructed. A recently developed option is quadrature-by-expansion (QBX) [89, 90]. Recently, af Klinteberg and Tornberg [6] applied

QBX to simulate a collection of non-Brownian spheroids immersed in a Stokes flow; however, the generalizations to more complex particle shapes requires an underlying, high-order smooth quadrature rule.

A particular challenge in developing singular quadrature schemes is preserving underlying symmetry properties of the single layer operator. Namely, the single layer integral operator is SPD and is rotationally invariant, meaning that if the body is rotated the result of applying the operator is also rotated in the same way. The Alpert quadrature correction used in this work gives a banded high-order log-singularity correction for the near-field component of $\mathbf{M}$ that is neither symmetric, nor positive definite, nor rotationally invariant. We found all of these artifacts to be numerical errors below the order of accuracy of the quadrature scheme, as expected. However, it would be much better to have a singular quadrature scheme that does not have such unphysical artifacts *by construction*. The traditional focus in boundary integral methods has been on achieving higher-order accuracy. What is more important for Brownian suspensions is preserving physical properties of the continuum operators in their discrete "mimetic" counterparts, so that even coarse resolutions give physically-consistent (even if not very accurate) discretizations.

We note that a Galerkin boundary integral formulation combined with a multipole expansion has been used by Singh and Adhikari and collaborators to model an active suspensions of spherical particles in an unbounded domain [112]. In some sense, this is an extension of the traditional Stokesian Dynamics method to include moments higher than the stresslet, as necessary to account for active flows. Recently the method has been extended to include Brownian contributions and to account for confinement near a no-slip boundary [113]. While also based on a boundary integral formulation, this class of methods differs in significant ways from the one proposed here. Most importantly, our approach uses numerical quadrature instead of analytical integration, and therefore generalizes to arbitrary (smooth) particle shapes. Furthermore, we do not truncate a multipole hierarchy at a small number of moments (say, after the dipole or quadrupole contribution), which can lead to uncontrolled loss of accuracy when the dilute approximation is no longer valid. In the FMM-accelerated boundary integral method, the number of terms used in multipole expansions is chosen to achieve a user-specified tolerance in the far field. High-order quadrature is used to achieve the same tolerance in the near field. That said, generalizations of FBIM to three dimensions require a nontrivial amount of effort, and for spherical particle the Galerkin approach mentioned above may be an effective alternative that yields sufficiently accurate answers in practice.

Another key aspect of Brownian Dynamics simulations is the temporal integration schemes. There are three main challenges in this regard. The first is that in three dimensions orientation cannot be represented by a vector. This can most straightforwardly be handled by using normalized quaternions to represent particle orientation, as shown in [94]. In the end, as long as a method to compute particle velocities in agreement with Eq. (3.2.20) is provided, handling the quaternion constraint simply amounts to using quaternion multiplication, rather than addition, to update orientations [94, 104]. The second challenge is capturing the stochastic drift term proportional to the divergence of the body mobility matrix using linear-scaling iterative methods. In this work we used methods that perform random finite difference on the body mobility. This requires solving two mobility problems per time step just to capture the stochastic drift term, therefore at least doubling the cost of a time step. In future work we will describe novel temporal integrators that can be used with FBIM to give more accurate answers for larger time step sizes, and which use only a single mobility solve to capture the stochastic drift term [104]. A third challenge is to handle the fact that Brownian displacements can lead to overlaps between the particles, even for small time step sizes. Unlike the rigid multiblob method [23], which builds on a regularized first-kind boundary integral formulation, traditional boundary integral methods based on singular quadratures break down when particles overlap. Even if particles do not overlap, unless the surface discretization is refined adaptively (which would be too costly for denser suspensions), traditional boundary integral representations will give unphysical answers that can easily lead to breakdown especially in the presence of noise. A possible solution to this is to use a regularization of the formulation for particle gaps below the resolution of the method. This is done naturally in the rigid multiblob method [104], and also more recently in a Galerkin multipole method [113], by using the Rotne-Prager-Yamakawa regularization of the Oseen tensor.

Another important direction of future work is to extend FBIM to model Brownian suspensions in other geometries, notably in unbounded domains. One option is to adapt our method to use a newly developed spectral Ewald summation for the free-space Stokeslet [114]. We note, however, that the computational cost of the FFTs in Ewald-type methods may become non-trivial for three dimensional problems. A challenge of great interest is incorporating the Fast Multipole Method (FMM) to generate the random surface velocity, thus developing a grid-free (near) linear-scaling method for Brownian suspension in an unbounded domain. For simple confined geometries, such as colloids sedimented in the vicinity of a single no-slip wall, one can use known analytical Green's functions for Stokes flow as done in [113]. It turns out that because of the more rapid decay of the Green's function in the presence of a wall, iterative methods can efficiently generate the random surface velocity without requiring special handling of the far-field interactions [95]. Nevertheless, achieving both linear scaling and controlled accuracy are challenges even in such simple geometries. For finite domains of more complicated geometry, one can discretize the domain boundary explicitly [115], and then employ the free-space Spectral Ewald method [114].

3.6 Appendix: Body mobility matrix and Lorentz reciprocal theorem

In this appendix we derive the following expression for elements of the body mobility matrix $\boldsymbol{\mathcal{N}}$ in terms of the periodic Green's function $\mathbb{G}(\boldsymbol{x}, \boldsymbol{y})$ of the Stokes equation,

$$(\boldsymbol{\mathcal{N}})_{ij} \equiv \mathcal{N}_{ij} = \int_\Gamma \int_\Gamma \boldsymbol{\lambda}^{(j)}(\boldsymbol{x}) \cdot \mathbb{G}(\boldsymbol{x}, \boldsymbol{y}) \cdot \boldsymbol{\lambda}^{(i)}(\boldsymbol{y}) \, \mathrm{d}S_{\boldsymbol{y}} \, \mathrm{d}S_{\boldsymbol{x}}, \tag{3.6.1}$$

where the precise definition of $\boldsymbol{\lambda}^{(i)}, \boldsymbol{\lambda}^{(j)}$ appears later.

For simplicity, we consider only a single rigid body Γ immersed in a Stokes fluid with periodic boundary conditions. The generalization of (3.6.1) to account for many bodies is straightforward. First, we recall that, the mobility problem that solves for the the translational velocity $\boldsymbol{u}$ and the rotational velocity $\boldsymbol{\omega}$ of the body, in response to the force $\boldsymbol{f}$ and torque $\boldsymbol{\tau}$ exerted on the body is described by steady Stokes equation with no-slip boundary condition, and force and torque balance conditions:

$$\begin{aligned}
-\boldsymbol{\nabla} \cdot \boldsymbol{\sigma} &= \nabla \pi - \eta \nabla^2 \boldsymbol{v} = \boldsymbol{0}, \\
\boldsymbol{\nabla} \cdot \boldsymbol{v} &= 0, \\
\boldsymbol{v}(\boldsymbol{x}) &= \boldsymbol{u} + \boldsymbol{\omega} \times (\boldsymbol{x} - \boldsymbol{q}), \quad \forall \boldsymbol{x} \in \Gamma, \\
\int_\Gamma \boldsymbol{\lambda}(\boldsymbol{x}) \, \mathrm{d}S_{\boldsymbol{x}} = \boldsymbol{f} \quad &\text{and} \quad \int_\Gamma (\boldsymbol{x} - \boldsymbol{q}) \times \boldsymbol{\lambda}(\boldsymbol{x}) \, \mathrm{d}S_{\boldsymbol{x}} = \boldsymbol{\tau},
\end{aligned} \tag{3.6.2}$$

where η is the fluid viscosity, $\boldsymbol{\sigma}$ the fluid stress tensor, and $\boldsymbol{v}$ is the fluid velocity, respectively. Here $\boldsymbol{\lambda} = (\boldsymbol{\sigma} \cdot \boldsymbol{n})(\boldsymbol{x})$ is the surface traction of the body with $\boldsymbol{n}$ being the unit normal vector to the surface. The mobility problem (3.6.2) can be viewed as a linear mapping

$$\boldsymbol{U} = \boldsymbol{\mathcal{N}} \boldsymbol{F}, \tag{3.6.3}$$

where $\boldsymbol{\mathcal{N}}$ is the body mobility matrix that relates the rigid body motion $\boldsymbol{U} = \{\boldsymbol{u}, \boldsymbol{\omega}\}$ to the applied force and torque $\boldsymbol{F} = \{\boldsymbol{f}, \boldsymbol{\tau}\}$.

Let $\{\boldsymbol{v}^{(i)}, \boldsymbol{\sigma}^{(i)}, \boldsymbol{u}^{(i)}, \boldsymbol{\omega}^{(i)}\}$ and $\{\boldsymbol{v}^{(j)}, \boldsymbol{\sigma}^{(j)}, \boldsymbol{u}^{(j)}, \boldsymbol{\omega}^{(j)}\}$ denote the solutions to the mobility problem (3.6.2) with applied force and torque $\boldsymbol{F} = \{\boldsymbol{f}^{(i)}, \boldsymbol{\tau}^{(i)}\} = \boldsymbol{e}^{(i)}$ and $\boldsymbol{F} = \{\boldsymbol{f}^{(j)}, \boldsymbol{\tau}^{(j)}\} = \boldsymbol{e}^{(j)}$ respectively. Here $\boldsymbol{e}^{(i)}$ and $\boldsymbol{e}^{(j)}$ are the standard basis vectors (in two dimensions, $\boldsymbol{e}^{(i)} \in \mathbb{R}^3$). It is not difficult to see that $\boldsymbol{U}^{(j)} = \{\boldsymbol{u}^{(j)}, \boldsymbol{\omega}^{(j)}\}$ corresponds to the j^{th} column of $\boldsymbol{\mathcal{N}}$. It is understood that whenever the force or torque is made dimensionless in the canonical problem, the other quantities' dimensions are adjusted accordingly. Thus, if the force is made dimensionless, velocities have units of the force-velocity mobility, while the tractions have units of inverse area. If the torque is made dimensionless, then the velocities have units of the torque-velocity mobility and the tractions have units of inverse length.

Invoking the Lorentz Reciprocal Theorem (LRT) [1] and eliminating boundary terms arise from integration-by-parts using periodic BCs, we obtain

$$\int_\Gamma \boldsymbol{v}^{(i)} \cdot \boldsymbol{\lambda}^{(j)} \, \mathrm{d}S = \int_\Gamma \boldsymbol{v}^{(j)} \cdot \boldsymbol{\lambda}^{(i)} \, \mathrm{d}S. \tag{3.6.4}$$

After substituting the no-slip BC for $\boldsymbol{v}^{(j)}$ on the RHS of (3.6.4), and make use of the force and torque balance condition for $\boldsymbol{\lambda}^{(i)}$, we obtain that

$$\begin{aligned} \int_\Gamma \boldsymbol{v}^{(i)} \cdot \boldsymbol{\lambda}^{(j)} \, dS &= \int_\Gamma \left[\boldsymbol{u}^{(j)} + \boldsymbol{\omega}^{(j)} \times (\boldsymbol{x} - \boldsymbol{q})\right] \cdot \boldsymbol{\lambda}^{(i)} \, \mathrm{d}S_{\boldsymbol{x}} \\ &= \boldsymbol{u}^{(j)} \cdot \boldsymbol{f}^{(i)} + \boldsymbol{\omega}^{(j)} \cdot \boldsymbol{\tau}^{(i)} \\ &= \boldsymbol{U}^{(j)} \cdot \boldsymbol{e}^{(i)} = \mathcal{N}_{ij}. \end{aligned} \tag{3.6.5}$$

We recall that $\boldsymbol{v}^{(i)}(\boldsymbol{x})$ for $\boldsymbol{x} \in \Gamma$ can be written as

$$\boldsymbol{v}^{(i)}(\boldsymbol{x}) = \int_\Gamma \mathbb{G}(\boldsymbol{x}, \boldsymbol{y}) \cdot \boldsymbol{\lambda}^{(i)}(\boldsymbol{y}) \, \mathrm{d}S_{\boldsymbol{y}}, \quad \boldsymbol{x} \in \Gamma, \tag{3.6.6}$$

where $\mathbb{G}$ is the Green's function of the Stokes equation with unit viscosity and periodic BCs (periodic Stokeslet), as dictated by the first-kind boundary integral formulation of the mobility problem [1]. Lastly, we substitute (3.6.6) for $\boldsymbol{v}^{(i)}$ on the LHS of (3.6.5) to conclude that

$$\mathcal{N}_{ij} = \int_\Gamma \int_\Gamma \boldsymbol{\lambda}^{(i)}(\boldsymbol{x}) \cdot \mathbb{G}(\boldsymbol{x}, \boldsymbol{y}) \cdot \boldsymbol{\lambda}^{(j)}(\boldsymbol{y}) \, \mathrm{d}S_{\boldsymbol{y}} \, \mathrm{d}S_{\boldsymbol{x}}. \tag{3.6.7}$$

3.7 Appendix: A completed second-kind formulation in two dimensions

In this appendix we present a completed second-kind boundary integral formulation of the deterministic Stokes boundary value problem (3.2.13), as used to compute highly-accurate reference results in the main body of the chapter (see Sec. 3.4). We reproduce this formulation here for the benefit of the reader, since the full formulation is not contained in published work to our knowledge. Using a *double-layer* integral representation of the Stokes flow[1], we can write the exterior flow velocity $\boldsymbol{v}(\boldsymbol{x} \in E)$ in terms of an unknown double-layer density $\boldsymbol{\varphi}(\boldsymbol{x})$ as

$$\boldsymbol{v}(\boldsymbol{x} \in E) = (\boldsymbol{\mathcal{D}}_\Gamma \boldsymbol{\varphi})(\boldsymbol{x}) + (\boldsymbol{\mathcal{G}}[\boldsymbol{q}]\boldsymbol{f})(\boldsymbol{x}) + (\boldsymbol{\mathcal{R}}[\boldsymbol{q}]\tau)(\boldsymbol{x}), \tag{3.7.1}$$

where $\boldsymbol{\mathcal{D}}_\Gamma$ is the double-layer integral potential, $\boldsymbol{\mathcal{G}}[\boldsymbol{q}]$ and $\boldsymbol{\mathcal{R}}[\boldsymbol{q}]$ are the Stokeslet and rotlet at $\boldsymbol{q}$, respectively. In two dimensions, these operators are defined as

$$(\boldsymbol{\mathcal{D}}_\Gamma \boldsymbol{\varphi})_j(\boldsymbol{x}) = \frac{1}{4\pi\eta} \int_\Gamma T_{jlm}(\boldsymbol{x} - \boldsymbol{y} + \mathrm{p}) n_m(\boldsymbol{y}) \varphi_l(\boldsymbol{y}) \, \mathrm{d}S_{\boldsymbol{y}}, \tag{3.7.2}$$

$$(\boldsymbol{\mathcal{G}}[\boldsymbol{q}]\boldsymbol{f})_j(\boldsymbol{x}) = \frac{1}{4\pi\eta} G_{jl}(\boldsymbol{x} - \boldsymbol{q}) f_l, \tag{3.7.3}$$

$$(\boldsymbol{\mathcal{R}}[\boldsymbol{q}]\tau)_j(\boldsymbol{x}) = \frac{1}{4\pi\eta} R_j(\boldsymbol{x} - \boldsymbol{q})\tau, \tag{3.7.4}$$

where G_{jl}, T_{jlm} and R_j are the free-space Stokeslet, stresslet, and rotlet with unit viscosity, respectively. In two dimensions,

$$G_{jl} = -\delta_{jl} \log r + \frac{x_j x_l}{r^2}, \quad T_{jlm} = -\frac{4 x_j x_l x_m}{r^4}, \quad R_j = \frac{x_j^\perp}{r^2}, \tag{3.7.5}$$

where $\boldsymbol{x}^\perp = (x_2, -x_1)$. It is well-known that the double-layer representation by itself cannot represent a flow that exerts force and torque. Therefore, a completion flow is generally required to complete the representation. The choice of completion flow is not unique [91], and for simplicity, we use the completion flow proposed by Power and Miranda [116], that is generated by a Stokeslet with applied force $\boldsymbol{f}$ at the tracking point $\boldsymbol{q}$ inside the particle, denoted as $\boldsymbol{\mathcal{G}}[\boldsymbol{q}]\boldsymbol{f}$, and a rotlet with torque τ (a scalar quantity in two dimensions), denoted as $\boldsymbol{\mathcal{R}}[\boldsymbol{q}]\tau$. We note that it is possible to avoid this kind of completion and still get a second-kind integral equation for the mobility problem [3, 4].

Taking the limit as $\boldsymbol{x}$ approaches the boundary Γ and using the jump condition for the double-layer potential [1], we obtain a second-kind boundary integral equation of the unknown translational and angular velocities $\{\boldsymbol{u}, \omega\}$ and the double-layer density $\boldsymbol{\varphi}$,

$$\boldsymbol{u} + \omega(\boldsymbol{x} - \boldsymbol{q})^\perp = \frac{1}{2}\boldsymbol{\varphi}(\boldsymbol{x}) + (\boldsymbol{\mathcal{D}}_\Gamma \boldsymbol{\varphi})(\boldsymbol{x}) + \boldsymbol{\mathcal{G}}[\boldsymbol{q}]\boldsymbol{f} + \boldsymbol{\mathcal{R}}[\boldsymbol{q}]\tau, \quad \forall \boldsymbol{x} \in \Gamma, \tag{3.7.6}$$

and Eq. (3.7.6) is closed by relating $\{\boldsymbol{u}, \omega\}$ to $\boldsymbol{\varphi}$ via[7]

$$\boldsymbol{u} = \frac{1}{|\Gamma|} \int_\Gamma \boldsymbol{\varphi}(\boldsymbol{x}) \, \mathrm{d}S_{\boldsymbol{x}}, \quad \omega = \frac{1}{W} \int_\Gamma \boldsymbol{\varphi}(\boldsymbol{x}) \cdot (\boldsymbol{x} - \boldsymbol{q})^\perp \, \mathrm{d}S_{\boldsymbol{x}}, \tag{3.7.7}$$

where $|\Gamma|$ is the length of Γ and $W = \int_\Gamma |\boldsymbol{x} - \boldsymbol{q}|^2 \, \mathrm{d}S_{\boldsymbol{x}}$. We remark that the random surface velocity $-\breve{\boldsymbol{v}}(\boldsymbol{x})$ can also be included on the left-hand-side of Eq. (3.7.6). This provides us a mixed first- and second-kind formulation for solving the SSBVP (3.2.10). That is, we

[7]These are the two dimensional analogs of [1, Eqs. (4.9.22)-(4.9.23)].

can generate the random surface velocity using the first-kind formulation described in the main body of the chapter, and then, solve Eqs. (3.7.6) and (3.7.7) for $\{\boldsymbol{u}, \omega\}$ to generate the action of $\boldsymbol{\mathcal{N}}$ and $\boldsymbol{\mathcal{N}}^{\frac{1}{2}}$ using a well-conditioned spectrally-accurate second-kind formulation. In this work we choose to use the first-kind formulation for generating both the deterministic motions and the Brownian displacements in the main body of the chapter, because the discrete fluctuation-dissipation balance is exactly preserved in the first-kind formulation, but not in the mixed formulation.

From a numerical standpoint, a desirable feature of the second-kind formulation in two dimensions is that the double-layer kernel (stresslet) has no singularities for points on the boundary, and discretizing the second-kind boundary integral equations using the regular trapezoidal rule leads to spectral accuracy. However, this holds neither in three dimensions nor for bodies that are near contact.

After discretizing Eqs. (3.7.6) and (3.7.7), we are required to evaluate the following periodic sums involving the Stokeslet, stresslet and rotlet in the resulting linear system,

$$u_j^G(\mathbf{x}_t) = \frac{1}{4\pi\eta} \sum_{\mathbf{p}\in\mathbb{Z}^2} G_{jl}(\mathbf{x}_t - \boldsymbol{q} + \mathbf{p}L) f_l, \tag{3.7.8}$$

$$u_j^R(\mathbf{x}_t) = \frac{1}{4\pi\eta} \sum_{\mathbf{p}\in\mathbb{Z}^2} R_j(\mathbf{x}_t - \boldsymbol{q} + \mathbf{p}L) \tau, \tag{3.7.9}$$

$$u_j^T(\mathbf{x}_t) = \frac{1}{4\pi\eta} \sum_{\mathbf{p}\in\mathbb{Z}^2} \sum_{s=1}^{N} T_{jlm}(\mathbf{x}_t - \mathbf{x}_s + \mathbf{p}L) S_{lm}(\mathbf{x}_s), \tag{3.7.10}$$

where $\mathbf{x}_t, \mathbf{x}_s \in \Gamma$ denote the target and source points, respectively, and $S_{lm}(\mathbf{x_s}) = \varphi_l(\mathbf{x}_s) n_m(\mathbf{x}_s) \Delta s$. We employ the Hasimoto function Eq. (3.3.13) to decompose the Stokeslet, rotlet and stresslet into near-field and far-field contributions, and thereafter, these sums can be efficiently computed using the Spectral Ewald method [77, 5]. The decomposition of the periodic Stokeslet has been presented in the main body of the chapter (see Eqs. (3.3.14) and (3.3.15)). The periodic sum of the rotlet can be decomposed as

$$u_j^R(\mathbf{x}_t) = \frac{1}{4\pi\eta} \sum_{\mathbf{p}\in\mathbb{Z}^2} R_j^{(r)}(\mathbf{x}_t - \boldsymbol{q} + \mathbf{p}L; \xi)\tau + \frac{1}{\eta V} \sum_{\mathbf{k}\neq\mathbf{0}} R_j^{(w)}(\mathbf{k}; \xi) \sin(\mathbf{k}\cdot(\mathbf{x}_t - \boldsymbol{q}))\tau,$$

where

$$R_j^{(r)}(\boldsymbol{x}; \xi) = \frac{x_j^\perp}{r^2}(1 - \xi^2 r^2) e^{-\xi^2 r^2}, \tag{3.7.11a}$$

$$R_j^{(w)}(\mathbf{k}; \xi) = \frac{2\pi k_j^\perp}{k^2}\left(1 + \frac{k^2}{4\xi^2}\right) e^{-k^2/4\xi^2}. \tag{3.7.11b}$$

The complete decomposition of the periodic double-layer potential consists of four parts: the mean stresslet term $T_{jlm}^{(0)}$, which defines a zero mean flow in periodic domains [5], the near-field $T_{jlm}^{(r)}$ and far-field $T_{jlm}^{(w)}$ contributions, and the term that arises from the diagonal elements of double-layer integral when $\mathbf{x}_t = \mathbf{x}_s$ (see [117]). In summary, we have

$$\begin{aligned} u_j^T(\mathbf{x}_t) = \frac{1}{4\pi\eta} \Bigg(& \sum_{s=1}^{N} T_{jlm}^{(0)}(\mathbf{x}_s) S_{lm}(\mathbf{x}_s) \\ & + \sum_{\mathbf{p}\in\mathbb{Z}^2} \sum_{n=1}^{N} T_{jlm}^{(r)}(\mathbf{x}_t - \mathbf{x}_s + \mathbf{p}L; \xi) S_{lm}(\mathbf{x}_s) \\ & + \frac{1}{V} \sum_{\mathbf{k}\neq\mathbf{0}} T_{jlm}^{(w)}(\mathbf{k}; \xi) \sum_{n=1}^{N} S_{lm}(\mathbf{x}_s) \sin(\mathbf{k}\cdot(\mathbf{x}_t - \mathbf{x}_s)), \\ & + 2\pi \sum_{s=1}^{N} \kappa(\mathbf{x}_s) \left[(\hat{\boldsymbol{t}} \otimes \hat{\boldsymbol{t}})(\mathbf{x}_s)\right]_{jl} \varphi_l(\mathbf{x}_s) \Bigg), \end{aligned} \tag{3.7.12}$$

where κ and $\hat{\boldsymbol{t}}$ are the curvature and unit tangent vector on Γ, and [118]

$$T_{jlm}^{(0)}(\boldsymbol{x}) = \frac{4\pi}{V}\delta_{lm} x_j, \tag{3.7.13a}$$

$$T_{jlm}^{(r)}(\boldsymbol{x}; \xi) = \left[-\frac{4x_j x_l x_m}{r^4}(1 + \xi^2 r^2) + 2\xi^2(\delta_{lm} x_j + \delta_{mj} x_l)\right] e^{-\xi^2 r^2}, \tag{3.7.13b}$$

$$T^{(w)}_{jlm}(\mathbf{k};\xi) = -\frac{4\pi}{k^4}\left[\left(1+\frac{k^2}{4\xi^2}\right)\left[k^2(k_j\delta_{lm}+k_l\delta_{mj})-2k_jk_lk_m\right]+k^2k_m\delta_{jl}\right]e^{-k^2/4\xi^2}. \quad (3.7.13c)$$

After we have efficient routines to compute the periodic sums described above, the discrete linear system obtained from Eqs. (3.7.6) and (3.7.7) can be solved iteratively using GMRES without preconditioning, since the second-kind formulation is well-conditioned.

Bibliography

[1] C. Pozrikidis, Boundary integral and singularity methods for linearized viscous flow, Cambridge University Press, 1992.

[2] A.-K. Tornberg, L. Greengard, A fast multipole method for the three-dimensional stokes equations, Journal of Computational Physics 227 (3) (2008) 1613–1619.

[3] M. Rachh, L. Greengard, Integral equation methods for elastance and mobility problems in two dimensions, SIAM Journal on Numerical Analysis 54 (5) (2016) 2889–2909.

[4] E. Corona, L. Greengard, M. Rachh, S. Veerapaneni, An integral equation formulation for rigid bodies in stokes flow in three dimensions, Journal of Computational Physics 332 (2017) 504–519.

[5] L. Af Klinteberg, A.-K. Tornberg, Fast ewald summation for stokesian particle suspensions, International Journal for Numerical Methods in Fluids 76 (10) (2014) 669–698.

[6] L. af Klinteberg, A.-K. Tornberg, A fast integral equation method for solid particles in viscous flow using quadrature by expansion, Journal of Computational Physics 326 (2016) 420–445.

[7] Y. Bao, J. Kaye, C. S. Peskin, A gaussian-like immersed-boundary kernel with three continuous derivatives and improved translational invariance, Journal of Computational Physics 316 (2016) 139 – 144, software and updated documentation available at https://

github.com/stochasticHydroTools/IBMethod, including also a new 5-pt kernel with three continuous derivatives. doi:http://dx.doi.org/10.1016/j.jcp.2016.04.024. URL http://www.sciencedirect.com/science/article/pii/S0021999116300663

[8] Y. Bao, A. Donev, B. E. Griffith, D. M. McQueen, C. S. Peskin, An Immersed Boundary Method with Divergence-Free Velocity Interpolation and Force Spreading, Journal of Computational Physics 347 (2017) 183–206. doi:http://dx.doi.org/10.1016/j.jcp.2017.06.041.

[9] Y. Bao, M. Rachh, E. Keaveny, L. Greengard, A. Donev, A fluctuating boundary integral method for Brownian suspensions, submitted to J. Comp. Phys., preprint ArXiv:1709.01480 (2017).

[10] C. S. Peskin, The immersed boundary method, Acta Numerica 11 (January 2002) (2003) 479–517.

[11] C. S. Peskin, Flow patterns around heart valves: A numerical method, Journal of Computational Physics 10 (2) (1972) 252–271.

[12] C. S. Peskin, Numerical analysis of blood flow in the heart, Journal of Computational Physics 25 (3) (1977) 220–252.

[13] D. McQueen, C. Peskin, Shared-Memory Parallel Vector Implementation of the Immersed Boundary Method for the Computation of Blood Flow in the Beating Mammalian Heart, The Journal of Supercomputing 11 (3) (1997) 213–236.

[14] B. Griffith, R. Hornung, D. McQueen, C. Peskin, An adaptive, formally second order accurate version of the immersed boundary method, J. Comput. Phys. 223 (1) (2007) 10–49, software available at https://github.com/ibamr/ibamr.

[15] B. Griffith, X. Luo, D. McQueen, C. Peskin, Simulating the fluid dynamics of natural and prosthetic heart valves using the immersed boundary method, International Journal of Applied Mechanics 1 (01) (2009) 137–177.

[16] B. Griffith, Immersed boundary model of aortic heart valve dynamics with physiological driving and loading conditions, Int J Numer Meth Biomed Eng 28 (2012) 317–345.

[17] A. P. S. Bhalla, R. Bale, B. E. Griffith, N. A. Patankar, A unified mathematical framework and an adaptive numerical method for fluid-structure interaction with rigid, deforming, and elastic bodies, Journal of Computational Physics 250 (2013) 446–476.

[18] E. Lushi, C. S. Peskin, Modeling and simulation of active suspensions containing large numbers of interacting micro-swimmers, Computers & Structures 122 (0) (2013) 239 – 248.

[19] T. G. Fai, B. E. Griffith, Y. Mori, C. S. Peskin, Immersed Boundary Method for Variable Viscosity and Variable Density Problems Using Fast Constant-Coefficient Linear Solvers I: Numerical Method and Results, SIAM Journal on Scientific Computing 35 (5) (2013) B1132–B1161.

[20] Y. Kim, M.-C. Lai, C. S. Peskin, Numerical simulations of two-dimensional foam by the immersed boundary method, Journal of Computational Physics 229 (13) (2010) 5194–5207. doi:10.1016/j.jcp.2010.03.035.

[21] Y. Kim, M.-C. Lai, C. S. Peskin, Y. Seol, Numerical simulations of three-dimensional foam by the immersed boundary method, Journal of Computational Physics 269 (2014) 1–21.

[22] B. Kallemov, A. P. S. Bhalla, B. E. Griffith, A. Donev, An immersed boundary method for rigid bodies, Communications in Applied Mathematics and Computa-

tional Science 11 (1) (2016) 79–141, software available at https://github.com/stochasticHydroTools/RigidBodyIB.

[23] F. Balboa Usabiaga, B. Kallemov, B. Delmotte, A. P. S. Bhalla, B. E. Griffith, A. Donev, Hydrodynamics of suspensions of passive and active rigid particles: a rigid multiblob approach, Communications in Applied Mathematics and Computational Science 11 (2) (2016) 217–296, software available at https://github.com/stochasticHydroTools/RigidMultiblobsWall. doi:10.2140/camcos.2016.11.217.

[24] M.-C. Lai, C. S. Peskin, An Immersed Boundary Method with Formal Second-Order Accuracy and Reduced Numerical Viscosity, Journal of Computational Physics 160 (2) (2000) 705–719.

[25] B. E. Griffith, C. S. Peskin, On the order of accuracy of the immersed boundary method: Higher order convergence rates for sufficiently smooth problems, Journal of Computational Physics 208 (1) (2005) 75–105. doi:10.1016/j.jcp.2005.02.011. URL http://linkinghub.elsevier.com/retrieve/pii/S0021999105000835

[26] C. S. Peskin, B. F. Printz, Improved volume conservation in the computation of flows with immersed elastic boundaries, Journal of computational physics 105 (1) (1993) 33–46.

[27] Z. Li, M.-C. Lai, The Immersed Interface Method for the NavierStokes Equations with Singular Forces, Journal of Computational Physics 171 (2) (2001) 822–842.

[28] L. Lee, R. J. LeVeque, An Immersed Interface Method for Incompressible Navier–Stokes Equations, SIAM Journal on Scientific Computing 25 (3) (2003) 832–856.

[29] D. B. Stein, R. D. Guy, B. Thomases, Immersed boundary smooth extension: A high-order method for solving pde on arbitrary smooth domains using fourier spectral methods, Journal of Computational Physics 304 (2016) 252–274.

[30] D. B. Stein, R. D. Guy, B. Thomases, Immersed Boundary Smooth Extension (IBSE): A high-order method for solving incompressible flows in arbitrary smooth domains, Journal of Computational Physics 335 (2016) 155–178. arXiv:1609.03851.

[31] B. Griffith, On the volume conservation of the immersed boundary method, Commun. Comput. Phys. 12 (2012) 401–432.

[32] W. Strychalski, R. D. Guy, Intracellular Pressure Dynamics in Blebbing Cells, Biophysical Journal 110 (5) (2016) 1168–1179.

[33] A. Dutt, V. Rokhlin, Fast Fourier Transforms for Nonequispaced Data, SIAM Journal on Scientific Computing 14 (6) (1993) 1368–1393.

[34] L. Greengard, J. Lee, Accelerating the nonuniform fast fourier transform, SIAM Review 46 (3) (2004) 443–454. doi:10.1137/S003614450343200X.

[35] D. Devendran, C. S. Peskin, An immersed boundary energy-based method for incompressible viscoelasticity, J. Comp. Phys. 231 (14) (2012) 4613–4642.

[36] A. M. Roma, C. S. Peskin, M. J. Berger, An adaptive version of the immersed boundary method, J. Comput. Phys. 153 (2) (1999) 509–534.

[37] B. Griffith, An accurate and efficient method for the incompressible Navier-Stokes equations using the projection method as a preconditioner, J. Comp. Phys. 228 (20) (2009) 7565–7595.

[38] R. Cortez, M. Minion, The Blob Projection Method for Immersed Boundary Problems, Journal of Computational Physics 161 (2) (2000) 428–453.

[39] F. B. Usabiaga, J. B. Bell, R. Delgado-Buscalioni, A. Donev, T. G. Fai, B. E. Griffith, C. S. Peskin, Staggered Schemes for Fluctuating Hydrodynamics, SIAM J. Multiscale Modeling and Simulation 10 (4) (2012) 1369–1408.

[40] C. Peskin, The immersed boundary method, Acta Numerica 11 (2002) 479–517.

[41] Y. Liu, Y. Mori, Properties of discrete delta functions and local convergence of the immersed boundary method, SIAM J. Numer. Anal. 50 (6) (2012) 2986–3015.

[42] J. M. Stockie, Analysis and computation of immersed boundaries, with application to pulp fibres, Ph.D. thesis (1997).

[43] J. F. Brady, R. J. Phillips, J. C. Lester, G. Bossis, Dynamic simulation of hydrodynamically interacting suspensions, Journal of Fluid Mechanics 195 (1988) 257–280.

[44] X. Yang, X. Zhang, Z. Li, G.-W. He, A smoothing technique for discrete delta functions with application to immersed boundary method in moving boundary simulations, Journal of Computational Physics 228 (20) (2009) 7821–7836.

[45] M. Unser, Splines: a perfect fit for signal and image processing, IEEE Signal Processing Magazine 16 (6) (1999) 22–38.

[46] M. Unser, A. Aldroubi, M. Eden, On the asymptotic convergence of B-spline wavelets to Gabor functions, IEEE Transactions on Information Theory 38 (2) (1992) 864–872.

[47] M.-c. Lai, Z. Li, A remark on jump conditions for the three-dimensional Navier-Stokes equations involving an immersed moving membrane, Applied Mathematics Letters 14 (2) (2001) 149–154.

[48] A. Goza, S. Liska, B. Morley, T. Colonius, Accurate computation of surface stresses and forces with immersed boundary methods, Journal of Computational Physics 321 (2016) 860–873.

[49] H. A. Williams, L. J. Fauci, D. P. Gaver III, Evaluation of interfacial fluid dynamical stresses using the immersed boundary method, Discrete and continuous dynamical systems. Series B 11 (2) (2009) 519.

[50] S. Liska, T. Colonius, A fast immersed boundary method for external incompressible viscous flows using lattice Green's functions, Journal of Computational Physics 331 (2017) 257–279.

[51] R. Cortez, C. S. Peskin, J. M. Stockie, D. Varela, Parametric Resonance in Immersed Elastic Boundaries, SIAM Journal on Applied Mathematics 65 (2) (2004) 494–520. doi:10.1137/S003613990342534X.
URL http://epubs.siam.org/doi/abs/10.1137/S003613990342534X

[52] W. Ko, J. M. Stockie, Correction to Parametric Resonance in Immersed Elastic Boundaries 65 (2) (2012) 6–11. arXiv:1207.4744v1.

[53] W. Ko, J. M. Stockie, Parametric resonance in spherical immersed elastic shells, SIAM Journal on Applied Mathematics 76 (1) (2016) 58–86. arXiv:1411.1345, doi:10.1137/15M101631X.
URL http://epubs.siam.org/doi/10.1137/15M101631X

[54] H. Gao, H. Wang, C. Berry, X. Luo, B. E. Griffith, Quasi-static image-based immersed boundary-finite element model of left ventricle under diastolic loading, International Journal for Numerical Methods in Biomedical Engineering 30 (11) (2014) 1199–1222. doi:10.1002/cnm.2652.

[55] B. E. Griffith, X. Luo, Hybrid finite difference/finite element version of the immersed boundary method, To appear in Int J Numer Meth Biomed Eng.

[56] D. Boffi, L. Gastaldi, L. Heltai, C. S. Peskin, On the hyper-elastic formulation of the immersed boundary method, Computer Methods in Applied Mechanics and Engineering 197 (25-28) (2008) 2210–2231. doi:10.1016/j.cma.2007.09.015.

[57] A. M. Fiore, F. B. Usabiaga, A. Donev, J. W. Swan, Rapid sampling of stochastic displacements in brownian dynamics simulations, J. Chem. Phys. 146 (12) (2017) 124116, software available at https://github.com/stochasticHydroTools/PSE. doi:10.1063/1.4978242.
URL http://dx.doi.org/10.1063/1.4978242

[58] A. Ghosh, P. Fischer, Controlled Propulsion of Artificial Magnetic Nanostructured Propellers, Nano Letters 9 (6) (2009) 2243–2245. doi:10.1021/nl900186w.

[59] C. Peskin, G. Odell, G. Oster, Cellular motions and thermal fluctuations: the Brownian ratchet, Biophysical Journal 65 (1) (1993) 316–324.

[60] H. Rafii-Tabar, R. Tavakoli-Darestani, Modelling the stochastic dynamics of biological nano-motors: An overview of recent results, Journal of Computational and Theoretical Nanoscience 6 (4) (2009) 806–819.

[61] H.-R. Jiang, N. Yoshinaga, M. Sano, Active Motion of a Janus Particle by Self-Thermophoresis in a Defocused Laser Beam, Phys. Rev. Lett. 105 (26) (2010) 268302. doi:10.1103/PhysRevLett.105.268302.
URL https://link.aps.org/doi/10.1103/PhysRevLett.105.268302

[62] D. L. Koch, G. Subramanian, Collective hydrodynamics of swimming microorganisms: Living fluids, Annual Review of Fluid Mechanics 43 (2011) 637–659.

[63] J. Palacci, S. Sacanna, A. P. Steinberg, D. J. Pine, P. M. Chaikin, Living crystals of light-activated colloidal surfers, Science 339 (6122) (2013) 936–940.

[64] W. F. Paxton, K. C. Kistler, C. C. Olmeda, A. Sen, S. K. St. Angelo, Y. Cao, T. E. Mallouk, P. E. Lammert, V. H. Crespi, Catalytic Nanomotors: Autonomous Movement of Striped Nanorods, Journal of the American Chemical Society 126 (41) (2004) 13424–13431. doi:10.1021/ja047697z.

[65] D. Takagi, A. B. Braunschweig, J. Zhang, M. J. Shelley, Dispersion of self-propelled rods undergoing fluctuation-driven flips, Phys. Rev. Lett. 110 (3) (2013) 038301.

[66] J. R. Howse, R. A. L. Jones, A. J. Ryan, T. Gough, R. Vafabakhsh, R. Golestanian, Self-Motile Colloidal Particles: From Directed Propulsion to Random Walk, Phys. Rev. Lett. 99 (4) (2007) 48102. doi:10.1103/PhysRevLett.99.048102.

[67] S. J. Ebbens, J. R. Howse, In pursuit of propulsion at the nanoscale, Soft Matter 6 (4) (2010) 726–738. doi:10.1039/B918598D.

[68] S. Kim, S. J. Karrila, Microhydrodynamics: principles and selected applications, Courier Corporation, 2013.

[69] R. M. Jendrejack, J. J. de Pablo, M. D. Graham, Stochastic simulations of DNA in flow: Dynamics and the effects of hydrodynamic interactions, J. Chem. Phys. 116 (17) (2002) 7752–7759.

[70] R. M. Jendrejack, D. C. Schwartz, M. D. Graham, J. J. de Pablo, Effect of confinement on DNA dynamics in microfluidic devices, J. Chem. Phys. 119 (2003) 1165.

[71] J. P. Hernandez-Ortiz, J. J. de Pablo, M. D. Graham, Fast Computation of Many-Particle Hydrodynamic and Electrostatic Interactions in a Confined Geometry, Phys. Rev. Lett. 98 (14) (2007) 140602.

[72] Y. Zhang, J. J. de Pablo, M. D. Graham, An immersed boundary method for brownian dynamics simulation of polymers in complex geometries: Application to dna flowing

through a nanoslit with embedded nanopits, The Journal of Chemical Physics 136 (2012) 014901.

[73] Z. Liang, Z. Gimbutas, L. Greengard, J. Huang, S. Jiang, A fast multipole method for the rotne–prager–yamakawa tensor and its applications, Journal of Computational Physics 234 (2013) 133–139.

[74] A. Sierou, J. F. Brady, Accelerated Stokesian Dynamics simulations, J. Fluid Mech. 448 (2001) 115–146.

[75] A. J. Banchio, J. F. Brady, Accelerated stokesian dynamics: Brownian motion, The Journal of chemical physics 118 (2003) 10323.

[76] M. Wang, J. F. Brady, Spectral ewald acceleration of stokesian dynamics for polydisperse suspensions, Journal of Computational Physics 306 (2016) 443–477.

[77] D. Lindbo, A.-K. Tornberg, Spectrally accurate fast summation for periodic stokes potentials, Journal of Computational Physics 229 (23) (2010) 8994–9010.

[78] M. Fixman, Construction of langevin forces in the simulation of hydrodynamic interaction, Macromolecules 19 (4) (1986) 1204–1207.

[79] T. Ando, E. Chow, Y. Saad, J. Skolnick, Krylov subspace methods for computing hydrodynamic interactions in brownian dynamics simulations, The Journal of Chemical Physics 137 (6) (2012) –. doi:http://dx.doi.org/10.1063/1.4742347.
URL http://scitation.aip.org/content/aip/journal/jcp/137/6/10.1063/1.4742347

[80] R. Adhikari, K. Stratford, M. E. Cates, A. J. Wagner, Fluctuating Lattice Boltzmann, Europhysics Letters 71 (2005) 473–479. arXiv:arXiv:cond-mat/0402598.

[81] B. Dünweg, A. Ladd, Lattice Boltzmann simulations of soft matter systems, Adv. Comp. Sim. for Soft Matter Sciences III (2009) 89–166.

[82] S. Delong, F. B. Usabiaga, R. Delgado-Buscalioni, B. E. Griffith, A. Donev, Brownian Dynamics without Green's Functions, J. Chem. Phys. 140 (13) (2014) 134110, software available at https://github.com/stochasticHydroTools/FIB. doi:10.1063/1.4869866.

[83] E. E. Keaveny, Fluctuating force-coupling method for simulations of colloidal suspensions, J. Comp. Phys. 269 (0) (2014) 61 – 79. doi:http://dx.doi.org/10.1016/j.jcp.2014.03.013.

[84] B. Delmotte, E. E. Keaveny, Simulating brownian suspensions with fluctuating hydrodynamics, The Journal of chemical physics 143 (24) (2015) 244109.

[85] J. Rotne, S. Prager, Variational treatment of hydrodynamic interaction in polymers, The Journal of Chemical Physics 50 (1969) 4831.

[86] H. Hasimoto, On the periodic fundamental solutions of the stokes equations and their application to viscous flow past a cubic array of spheres, J. Fluid Mech 5 (02) (1959) 317–328.

[87] D. Liu, E. Keaveny, M. R. Maxey, G. E. Karniadakis, Force-coupling method for flows with ellipsoidal particles, Journal of Computational Physics 228 (10) (2009) 3559–3581.

[88] D. J. Smith, A boundary element regularized stokeslet method applied to cilia-and flagella-driven flow, Proceedings of the Royal Society of London A 465 (2112) (2009) 3605–3626.

[89] C. L. Epstein, L. Greengard, A. Klöckner, On the Convergence of Local Expansions of Layer Potentials, SIAM Journal on Numerical Analysis 51 (5) (2013) 2660–2679.

doi:10.1137/120902859.
URL https://doi.org/10.1137/120902859

[90] A. Klöckner, A. Barnett, L. Greengard, M. ONeil, Quadrature by expansion: A new method for the evaluation of layer potentials, Journal of Computational Physics 252 (2013) 332–349. doi:http://dx.doi.org/10.1016/j.jcp.2013.06.027.
URL http://www.sciencedirect.com/science/article/pii/S0021999113004579

[91] E. E. Keaveny, M. J. Shelley, Applying a second-kind boundary integral equation for surface tractions in stokes flow, Journal of Computational Physics 230 (5) (2011) 2141–2159.

[92] B. K. Alpert, Hybrid Gauss-Trapezoidal Quadrature Rules, SIAM Journal on Scientific Computing 20 (5) (1999) 1551–1584. doi:10.1137/S1064827597325141.
URL https://doi.org/10.1137/S1064827597325141

[93] E. Chow, Y. Saad, Preconditioned krylov subspace methods for sampling multivariate gaussian distributions, SIAM Journal on Scientific Computing 36 (2) (2014) A588–A608.

[94] S. Delong, F. B. Usabiaga, A. Donev, Brownian dynamics of confined rigid bodies, J. Chem. Phys. 143 (14) (2015) 144107, software available at https://github.com/stochasticHydroTools/RigidMultiblobsWall. doi:http://dx.doi.org/10.1063/1.4932062.
URL http://scitation.aip.org/content/aip/journal/jcp/143/14/10.1063/1.4932062

[95] F. B. Usabiaga, B. Delmotte, A. Donev, Brownian dynamics of confined suspensions of active microrollers, J. Chem. Phys. 146 (13) (2017) 134104, software available at https://github.com/stochasticHydroTools/RigidMultiblobsWall.

[96] R. Fox, G. Uhlenbeck, Contributions to Non-Equilibrium Thermodynamics. I. Theory of Hydrodynamical Fluctuations, Physics of Fluids 13 (1970) 1893.

[97] E. J. Hinch, Application of the Langevin equation to fluid suspensions, J. Fluid Mech. 72 (03) (1975) 499–511.

[98] J. N. Roux, Brownian particles at different times scales: a new derivation of the Smoluchowski equation, Phys. A 188 (1992) 526–552.

[99] E. H. Hauge, A. Martin-Lof, Fluctuating hydrodynamics and Brownian motion, J. Stat. Phys. 7 (3) (1973) 259–281.

[100] R. Zwanzig, M. Bixon, Compressibility effects in the hydrodynamic theory of Brownian motion, J. Fluid Mech. 69 (1975) 21–25.

[101] G. A. Pavliotis, A. M. Stuart, Multiscale methods: averaging and homogenization, Vol. 53, Springer, 2008.

[102] M. Fixman, Simulation of polymer dynamics. I. General theory, J. Chem. Phys. 69 (1978) 1527. doi:10.1063/1.436725.
URL http://link.aip.org/link/JCPSA6/v69/i4/p1527/s1&Agg=doi

[103] S. Delong, Y. Sun, B. E. Griffith, E. Vanden-Eijnden, A. Donev, Multiscale temporal integrators for fluctuating hydrodynamics, Phys. Rev. E 90 (2014) 063312, software available at https://github.com/stochasticHydroTools/MixingIBAMR. doi:10.1103/PhysRevE.90.063312.

[104] B. Sprinkle, F. B. Usabiaga, N. A. Patankar, A. Donev, Large scale brownian dynamics of confined suspensions of rigid particles, The Journal of Chemical Physics 147 (24) (2017) 244103, software available at https://github.com/stochasticHydroTools/RigidMultiblobsWall. doi:10.1063/1.5003833.

[105] J. T. Beale, W. Ying, J. R. Wilson, A Simple Method for Computing Singular or Nearly Singular Integrals on Closed Surfaces, Communications in Computational Physics 20 (3) (2016) 733–753. arXiv:1508.00265, doi:10.4208/cicp.030815.240216a.
URL http://arxiv.org/abs/1508.00265http://dx.doi.org/10.4208/cicp.030815.240216a

[106] J. Bremer, Z. Gimbutas, A Nyström method for weakly singular integral operators on surfaces, Journal of Computational Physics 231 (14) (2012) 4885–4903. doi:http://dx.doi.org/10.1016/j.jcp.2012.04.003.
URL http://www.sciencedirect.com/science/article/pii/S002199911200174X

[107] P. Kolm, V. Rokhlin, Numerical quadratures for singular and hypersingular integrals, Computers & Mathematics with Applications 41 (3) (2001) 327–352. doi:http://dx.doi.org/10.1016/S0898-1221(00)00277-7.
URL http://www.sciencedirect.com/science/article/pii/S0898122100002777

[108] M. P. Allen, D. J. Tildesley, Computer Simulations of Liquids, Oxford Science Publications, 1987.

[109] D. Lindbo, A.-K. Tornberg, Spectral accuracy in fast ewald-based methods for particle simulations, Journal of Computational Physics 230 (24) (2011) 8744–8761.

[110] A. S. Sangani, A. Acrivos, Slow flow through a periodic array of spheres, International Journal of Multiphase Flow 8 (4) (1982) 343–360. doi:http://dx.doi.org/10.1016/0301-9322(82)90047-7.
URL http://www.sciencedirect.com/science/article/pii/0301932282900477

[111] A. S. Sangani, A. Acrivos, Slow flow past periodic arrays of cylinders with application to heat transfer, International Journal of Multiphase Flow 8 (3) (1982) 193–206.

doi:http://dx.doi.org/10.1016/0301-9322(82)90029-5.
URL http://www.sciencedirect.com/science/article/pii/0301932282900295

[112] R. Singh, S. Ghose, R. Adhikari, Many-body microhydrodynamics of colloidal particles with active boundary layers, Journal of Statistical Mechanics: Theory and Experiment 2015 (6) (2015) P06017.
URL http://stacks.iop.org/1742-5468/2015/i=6/a=P06017

[113] R. Singh, R. Adhikari, Fluctuating hydrodynamics and the brownian motion of an active colloid near a wall, European Journal of Computational Mechanics 26 (1-2) (2017) 78–97.

[114] L. af Klinteberg, D. S. Shamshirgar, A.-K. Tornberg, Fast Ewald summation for free-space Stokes potentials, Research in the Mathematical Sciences 4 (1) (2017) 1. doi:10.1186/s40687-016-0092-7.

[115] O. Marin, K. Gustavsson, A.-K. Tornberg, A highly accurate boundary treatment for confined stokes flow, Computers & Fluids 66 (2012) 215–230.

[116] H. Power, G. Miranda, Second Kind Integral Equation Formulation of Stokes' Flows Past a Particle of Arbitrary Shape, SIAM Journal on Applied Mathematics 47 (4) (1987) 689–698. doi:10.1137/0147047.
URL https://doi.org/10.1137/0147047

[117] G. Biros, L. Ying, D. Zorin, A fast solver for the Stokes equations with distributed forces in complex geometries, Journal of Computational Physics 193 (1) (2004) 317–348. doi:http://dx.doi.org/10.1016/j.jcp.2003.08.011.
URL http://www.sciencedirect.com/science/article/pii/S0021999103004352

[118] G. A. L. Van De Vorst, G. a. L. Vorst, Integral formulation to simulate the viscous sintering of a two-dimensional lattice of periodic unit cells, Journal of Engineering Mathematics 30 (1-2) (1996) 97–118. doi:10.1007/BF00118825.
URL https://doi.org/10.1007/BF00118825