Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

# An Immersed Boundary method with divergence-free velocity interpolation and force spreading

Yuanxun Bao [a,*], Aleksandar Donev [a], Boyce E. Griffith [b], David M. McQueen [a], Charles S. Peskin [a]

[a] *Courant Institute of Mathematical Sciences, New York University, 251 Mercer Street, New York, NY, USA*
[b] *Departments of Mathematics and Biomedical Engineering, Carolina Center for Interdisciplinary Applied Mathematics, and McAllister Heart Institute, University of North Carolina, Chapel Hill, NC, USA*

A R T I C L E   I N F O

*Article history:*
Received 29 January 2017
Received in revised form 14 June 2017
Accepted 25 June 2017
Available online 28 June 2017

*Keywords:*
Immersed Boundary method
Fluid–structure interaction
Incompressible flow
Volume conservation
Velocity interpolation
Force spreading

A B S T R A C T

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 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 can suffer from poor volume conservation since the interpolated Lagrangian velocity field is not generally divergence-free, and so this can cause spurious volume changes. In practice, the lack of volume conservation is especially pronounced for cases where there are large pressure differences across thin structural boundaries. The aim of this paper is to greatly reduce the volume error of the IB method by introducing velocity-interpolation and force-spreading schemes with the properties that the interpolated velocity field in which the structure moves is at least $\mathscr{C}^1$ and satisfies a continuous divergence-free condition, and that the force-spreading operator is the adjoint of the velocity-interpolation operator. 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. Further, the new method provides smoother Lagrangian forces (tractions) than traditional IB methods. The method presented here is restricted to periodic computational domains. Its generalization to non-periodic domains is important future work.

## 1. Introduction

The *Immersed Boundary* (IB) method [34] 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

* Corresponding author.
*E-mail addresses:* billbao@cims.nyu.edu (Y. Bao), donev@courant.nyu.edu (A. Donev), boyceg@unc.edu (B.E. Griffith), mcqueen@cims.nyu.edu (D.M. McQueen), peskin@cims.nyu.edu (C.S. Peskin).

http://dx.doi.org/10.1016/j.jcp.2017.06.041
0021-9991/© 2017 Elsevier Inc. All rights reserved.

flow patterns around the heart valves [32,33], and since its success in modeling cardiac fluid dynamics [31,15,16,14], it has been extended and applied to various other applications, including but not limited to motion of biological swimmers [3,30], dynamics of red-blood cells [9] and dry foam [21,22], and rigid body motion [20,43].

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 $\delta_h$. One effective way to construct $\delta_h$ is to require the regularized delta function to satisfy a set of moment conditions to achieve approximate grid translation-invariance and desired interpolation accuracy [1,2], 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 [26,19], and it can be relatively poor of volume conservation [35]. 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) [28,27], and more recently, a new method known as *Immersed Boundary Smooth Extension* [37,38]. 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 [17], 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 $\boldsymbol{U}(\boldsymbol{X}) = (\mathcal{J}\mathbf{u})(\boldsymbol{X})$, where $\mathcal{J}$ 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.*, $(\nabla \cdot \boldsymbol{U})(\boldsymbol{X}) = 0$, 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 $\mathcal{J}_{\mathrm{IB}}$) is not *continuously* divergence-free [35], 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 [35]. 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 [21,22] for applications). To address the issue of spurious currents across immersed structure supporting extremely large pressure differences, Guy and Strychalski [39] developed a different extension of the IB method that uses non-uniform Fast Fourier Transform [8,12] (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 [15,14,3,20,43,7]. 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 [17] 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 [15,36] and physical boundary conditions [13]. 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 [28,27] and the Blob-Projection method [5], 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 paper is concerned with further improving volume conservation of IBMAC by constructing a *continuous* velocity-interpolation operator $\mathcal{J}$ that is divergence-free in the *continuous* sense. The discrete IB interpolation operator $\boldsymbol{S}^{\star}$ is simply the restriction of $\mathcal{J}$ to the Lagrangian markers. The key idea introduced in this paper 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 $\delta_h$. 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 [35]. 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 paper is organized as follows. In Sec. 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. 3 introduces the two main contributions of this paper: divergence-free velocity interpolation and force spreading. In Sec. 4, 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. 5, where the volume-conserving characteristics of the new scheme are assessed.

## 2. Equations of motion and spatial discretization

### 2.1. Equations of motion

This section provides a brief description of the continuum equations of motion in the IB framework [34]. We assume a neutrally-buoyant elastic structure $\Gamma$ 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. (2.1) and (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 $\rho$ and the viscosity $\mu$ of the fluid are constant. The fluid–structure coupled equations are:

$$\rho\left(\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u}\cdot\nabla\boldsymbol{u}\right) + \nabla p = \mu\nabla^2\boldsymbol{u} + \boldsymbol{f}, \tag{2.1}$$

$$\nabla\cdot\boldsymbol{u} = 0, \tag{2.2}$$

$$\boldsymbol{f}(\boldsymbol{x}, t) = \int_{\Gamma} \boldsymbol{F}(\boldsymbol{s}, t)\,\delta(\boldsymbol{x} - \boldsymbol{\mathcal{X}}(\boldsymbol{s}, t))\,\mathrm{d}\boldsymbol{s}, \tag{2.3}$$

$$\frac{\partial \boldsymbol{\mathcal{X}}}{\partial t}(\boldsymbol{s}, t) = \boldsymbol{u}(\boldsymbol{\mathcal{X}}(\boldsymbol{s}, t), t) = \int_{\Omega} \boldsymbol{u}(\boldsymbol{x}, t)\,\delta(\boldsymbol{x} - \boldsymbol{\mathcal{X}}(\boldsymbol{s}, t))\,\mathrm{d}\boldsymbol{x}, \tag{2.4}$$

$$\boldsymbol{F}(\boldsymbol{s}, t) = \boldsymbol{\mathcal{F}}[\boldsymbol{\mathcal{X}}(\cdot, t)\,;\boldsymbol{s}] = -\frac{\delta E}{\delta \boldsymbol{\mathcal{X}}}(\boldsymbol{s}, t). \tag{2.5}$$

Eqs. (2.3) and (2.4) are the fluid–structure interaction equations that couple the Eulerian and the Lagrangian variables. Eq. (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,

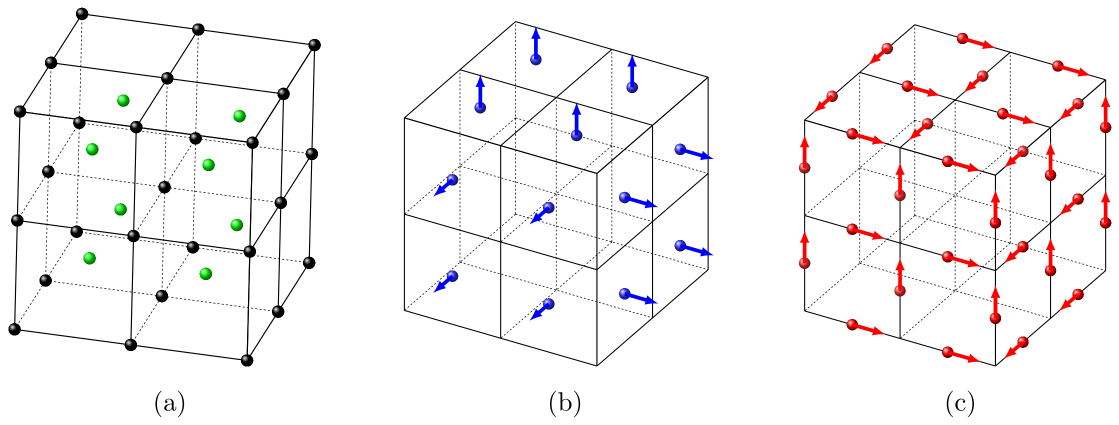


**Fig. 1.** Staggered grids on which discrete grid functions are defined. (a) Cell-centered (green) and node-centered (black) grids for scalar functions. (b) Face-centered grid for vector grid functions. (c) Edge-centered grid for vector grid functions. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

where $\boldsymbol{\mathcal{X}}(\boldsymbol{s}, t)$ is the physical position of the Lagrangian point $\boldsymbol{s}$. Eq. (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. (2.5), 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. Spatial discretization

Throughout the paper, 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. 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. 1b), denoted by $\mathbb{F}$, with each 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. 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. 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. 3, we will use these half-shifted staggered grids to construct divergence-free velocity interpolation and force spreading.

To discretize the differential operators in Eqs. (2.1) and (2.2), we introduce the central difference operators corresponding to the partial derivatives $\partial/\partial x_\alpha$,

$$D^h_\alpha \varphi := \frac{\varphi(\mathbf{x} + \frac{h}{2}\mathbf{e}_\alpha) - \varphi(\mathbf{x} - \frac{h}{2}\mathbf{e}_\alpha)}{h}, \quad \alpha = 1, 2, 3, \tag{2.6}$$

where $\varphi$ 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 $D^h_\alpha$ to define the discrete gradient, divergence and curl operators:

$$\mathbf{G}^h \varphi := (D^h_1\varphi,\ D^h_2\varphi,\ D^h_3\varphi), \tag{2.7}$$

$$\mathbf{D}^h \cdot \mathbf{v} := D^h_\alpha v_\alpha, \tag{2.8}$$

$$\mathbf{D}^h \times \mathbf{v} := \epsilon_{ijk} D^h_j v_k, \tag{2.9}$$

where $\mathbf{v}$ is a vector grid function, $\epsilon_{ijk}$ 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,

$$\mathbf{G}^h : \varphi(\mathbb{C}) \longrightarrow \mathbf{v}(\mathbb{F}) \text{ or } \varphi(\mathbb{N}) \longrightarrow \mathbf{v}(\mathbb{E}), \tag{2.10}$$

$$\mathbf{D}^h \cdot : \mathbf{v}(\mathbb{E}) \longrightarrow \varphi(\mathbb{N}) \text{ or } \mathbf{v}(\mathbb{F}) \longrightarrow \varphi(\mathbb{C}), \tag{2.11}$$

$$\mathbf{D}^h \times : \mathbf{v}(\mathbb{E}) \longrightarrow \mathbf{v}(\mathbb{F}) \text{ or } \mathbf{v}(\mathbb{F}) \longrightarrow \mathbf{v}(\mathbb{E}). \tag{2.12}$$

Although the curl operator does not appear in the equations of motion explicitly, we define it here for use in Sec. 3. 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 $\nabla^2$:

$$L^h\varphi := \sum_{\alpha=1}^{3} \frac{\varphi(\mathbf{x}+h\mathbf{e}_\alpha) - 2\varphi(\mathbf{x}) + \varphi(\mathbf{x}-h\mathbf{e}_\alpha)}{h^2}. \tag{2.13}$$

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.

### 2.2.1. Advection

We follow the same treatment of discretization of the advection term as in earlier presentations of the IB method [7]. From the incompressibility of the fluid flow $\nabla \cdot \boldsymbol{u} = 0$, we can write the advection term in the skew-symmetric form

$$[(\boldsymbol{u}\cdot\nabla)\boldsymbol{u}]_\alpha = \frac{1}{2}\boldsymbol{u}\cdot(\nabla u_\alpha) + \frac{1}{2}\nabla\cdot(\boldsymbol{u}u_\alpha), \quad \alpha = 1,2,3. \tag{2.14}$$

Let $\boldsymbol{N}(\mathbf{u})$ denote the discretization of Eq. (2.14), and we define

$$[\boldsymbol{N}(\mathbf{u})]_\alpha = \frac{1}{2}\tilde{\mathbf{u}}\cdot\mathbf{G}^{2h}u_\alpha + \frac{1}{2}\mathbf{D}^{2h}\cdot(\tilde{\mathbf{u}}u_\alpha), \quad \alpha = 1,2,3, \tag{2.15}$$

where $\tilde{\mathbf{u}}$ denotes an averaged collocated advective velocity whose components all live on the same grid as $u_\alpha$. The advective velocity $\tilde{\mathbf{u}}$ in [7] 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

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

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. (2.15) so that the resulting grid functions are all defined on the same grid as $u_\alpha$. A more compact discretization of the advection term has been previously described in [17,13,42].

### 2.2.2. Fluid–structure interaction

The immersed structure $\Gamma$ 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 $\boldsymbol{U}(\boldsymbol{X}) = (\mathcal{J}\mathbf{u})(\boldsymbol{X})$. In the conventional IB method, the *continuous* velocity-interpolation operator $\mathcal{J}_{\text{IB}}$ can be defined as

$$\boldsymbol{U}(\boldsymbol{X}) = (\mathcal{J}_{\text{IB}}\mathbf{u})(\boldsymbol{X}) := \sum_{\mathbf{x}\in\mathbb{F}} \mathbf{u}(\mathbf{x})\delta_h(\mathbf{x}-\boldsymbol{X})h^3. \tag{2.16}$$

We note that the interpolated velocity field given by Eq. (2.16) is *not* generally divergence-free with respect to the continuum divergence operator,[1] *i.e.*, generally

$$(\nabla\cdot\boldsymbol{U})(\boldsymbol{X}) = -\sum_{\mathbf{x}\in\mathbb{F}} \mathbf{u}(\mathbf{x})\cdot(\nabla\delta_h)(\mathbf{x}-\boldsymbol{X})h^3 \neq 0, \tag{2.17}$$

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

$$(\boldsymbol{S}^\star[\mathbf{X}]\mathbf{u})(\mathbf{X}) = (\mathcal{J}\mathbf{u})(\mathbf{X}). \tag{2.18}$$

[1] The interpolated velocity given by Eq. (2.16) has the same regularity as the regularized delta function $\delta_h$ which are generally at least $\mathscr{C}^1$ in the IB method, and therefore, the divergence of $\boldsymbol{U}$ is well-defined.

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.

The discretization of the interaction equations (Eqs. (2.3) and (2.4)) can be concisely written in the form

$$\mathbf{f}(\mathbf{x}) = (\boldsymbol{S}[\mathbf{X}]\mathbf{F})\,(\mathbf{x}), \tag{2.19}$$

$$\mathbf{U}(\mathbf{X}) = \left(\boldsymbol{S}^{\star}[\mathbf{X}]\mathbf{u}\right)(\mathbf{X}), \tag{2.20}$$

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 $\boldsymbol{S}^{\star}_{\text{IB}}$ are simply discrete approximations of the surface and volume integrals in Eqs. (2.3) and (2.4), *i.e.*,

$$(\boldsymbol{S}_{\text{IB}}[\mathbf{X}]\mathbf{F})(\mathbf{x}) := \sum_{m=1}^{M} \mathbf{F}_m\,\delta_h(\mathbf{x}-\mathbf{X}_m)\,\Delta\mathbf{s}, \tag{2.21}$$

$$(\boldsymbol{S}^{\star}_{\text{IB}}[\mathbf{X}]\mathbf{u})(\mathbf{X}) := \sum_{\mathbf{x}\in\mathbb{F}} \mathbf{u}(\mathbf{x})\delta_h(\mathbf{x}-\mathbf{X})h^3, \tag{2.22}$$

and they are adjoint operators with respect to the power identity (inner product) defined later in Eq. (3.10). Note that Eq. (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. (2.21) and (2.22), the Dirac delta function is replaced by a regularized delta function $\delta_h$ to facilitate the coupling between the Eulerian and Lagrangian grids, which is taken to be of the tensor-product form

$$\delta_h(\mathbf{x}) = \frac{1}{h^3}\phi\left(\frac{x_1}{h}\right)\phi\left(\frac{x_2}{h}\right)\phi\left(\frac{x_3}{h}\right), \tag{2.23}$$

where $\phi(r)$ denotes the one-dimensional immersed-boundary kernel that is constructed from a set of moment conditions to achieve approximate grid translation invariance [34,2].

In the following section, we will develop a new velocity-interpolation operator $\mathcal{J}$ that produces a *continuously* divergence-free interpolated velocity field constructed from a discretely divergence-free discrete fluid velocity.

In summary, the spatially-discretized equations of motion are

$$\rho\left(\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} + \boldsymbol{N}(\mathbf{u})\right) + \mathbf{G}^h p = \mu\mathbf{L}^h\mathbf{u} + \boldsymbol{S}[\mathbf{X}]\mathbf{F}, \tag{2.24}$$

$$\mathbf{D}^h\cdot\mathbf{u} = 0, \tag{2.25}$$

$$\frac{\mathrm{d}\mathbf{X}}{\mathrm{d}t} = \mathbf{U}(\mathbf{X},t) = \boldsymbol{S}^{\star}[\mathbf{X}]\mathbf{u}. \tag{2.26}$$

## 3. Divergence-free velocity interpolation and force spreading

This section presents the two main contributions of this paper: 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 Appendix A, will facilitate the reading of this section.

### 3.1. Divergence-free velocity interpolation

Here we introduce a new recipe for constructing an interpolated velocity field $\boldsymbol{U}(\boldsymbol{X}) = (\mathcal{J}\mathbf{u})(\boldsymbol{X})$ 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 $\Gamma$. 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 $\mathbf{u}(\mathbf{x})$,

$$\mathbf{u}_0 = \frac{1}{V}\sum_{\mathbf{x}\in\mathbb{F}} \mathbf{u}(\mathbf{x})h^3, \tag{3.1}$$

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

$$\begin{cases} \mathbf{D}^h \times \mathbf{a} = \mathbf{u} - \mathbf{u}_0, \\ \mathbf{D}^h \cdot \mathbf{a} = 0, \end{cases} \tag{3.2}$$

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. (3.2), then the discrete velocity potential $\mathbf{a}(\mathbf{x})$ is only uniquely defined up to $\mathbf{G}^h\psi$, where $\psi$ 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 Appendix B, we prove that the discrete vector potential $\mathbf{a}(\mathbf{x})$ defined by Eq. (3.2) exists (see Theorem 3). To determine $\mathbf{a}(\mathbf{x})$ explicitly, we take the discrete curl of the first equation in Eq. (3.2) and use the identity Eq. (A.3) with the gauge condition of $\mathbf{a}(\mathbf{x})$, which leads to a vector Poisson equation for $\mathbf{a}(\mathbf{x})$,

$$-\mathbf{L}^h \mathbf{a} = \mathbf{D}^h \times \mathbf{u}, \tag{3.3}$$

that can be efficiently solved. Note that the solution of the Poisson problem Eq. (3.3) 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

$$\boldsymbol{A}(\boldsymbol{X}) = \sum_{\mathbf{x}\in\mathbb{E}} \mathbf{a}(\mathbf{x})\, \delta_h(\mathbf{x} - \boldsymbol{X})h^3. \tag{3.4}$$

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

$$(\nabla \times \boldsymbol{A})(\boldsymbol{X}) = \sum_{\mathbf{x}\in\mathbb{E}} \mathbf{a}(\mathbf{x}) \times (\nabla\delta_h)(\mathbf{x} - \boldsymbol{X})h^3, \tag{3.5}$$

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

$$\boldsymbol{U}(\boldsymbol{X}) = (\mathcal{J}\mathbf{u})(\boldsymbol{X}) = \mathbf{u}_0 + \sum_{\mathbf{x}\in\mathbb{E}} \mathbf{a}(\mathbf{x}) \times (\nabla\delta_h)(\mathbf{x} - \boldsymbol{X})h^3. \tag{3.6}$$

We note that the interpolation Eq. (3.4) 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. (3.6). Notice that, by construction, the interpolated velocity in Eq. (3.6) 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. (3.3). Second, if the regularized delta function $\delta_h$ is $\mathscr{C}^k$, we note that the interpolated velocity field given by Eq. (3.6) 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 [2]). Moreover, the continuity of derivatives of $\delta_h$ also applies globally, including at the edges for the cube. Since the continuum vector potential defined by Eq. (3.4) 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 $\nabla \cdot \boldsymbol{U}$ 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 [34], unless the Lagrangian mesh is discretized with impractically high resolution (8 markers per fluid meshwidth, see Fig. 5). 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 $\phi_{4h}$), the IB kernels considered in this paper include the $\mathscr{C}^3$ 5-point and 6-point kernels [1,2] (denoted by $\phi_{5h}^{\text{new}}$ and $\phi_{6h}^{\text{new}}$, respectively), and the $\mathscr{C}^2$ 4-point B-spline kernel [40],

$$\phi_{4h}^B(r) = \begin{cases} \frac{2}{3} - r^2 + \frac{1}{2}r^3 & 0 \le |r| < 1, \\ \frac{4}{3} - 2r + r^2 - \frac{1}{6}r^3 & 1 \le |r| < 2, \\ 0 & |r| \ge 2, \end{cases} \tag{3.7}$$

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

$$\phi_{6h}^B(r) = \begin{cases} \frac{11}{20} - \frac{1}{2}r^2 + \frac{1}{4}r^4 - \frac{1}{12}r^5 & 0 \le |r| < 1, \\ \frac{17}{40} + \frac{5}{8}r - \frac{7}{4}r^2 + \frac{5}{4}r^3 - \frac{3}{8}r^4 + \frac{1}{24}r^5 & 1 \le |r| < 2, \\ \frac{81}{40} - \frac{27}{8}r + \frac{9}{4}r^2 - \frac{3}{4}r^3 + \frac{1}{8}r^4 - \frac{1}{120}r^5 & 2 \le |r| < 3, \\ 0 & |r| \ge 3. \end{cases} \tag{3.8}$$

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 [40]. The limiting function in this sequence is a Gaussian [41], which is exactly translation-invariant and isotropic. The family of IB kernels with nonzero even moment conditions, such as $\phi_{4h}$ and $\phi_{6h}^{\text{new}}$, also have a Gaussian-like shape, but it is not currently known whether this sequence of functions also converges to a Gaussian.

## 3.2. The force-spreading operator

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

$$(\mathbf{u}, \boldsymbol{S}\mathbf{F})_{\mathbf{x}} = (\boldsymbol{S}^\star\mathbf{u}, \mathbf{F})_{\mathbf{X}}, \tag{3.9}$$

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,[2]

$$\sum_{\mathbf{x}\in\mathbb{F}} \mathbf{u}(\mathbf{x})\cdot\mathbf{f}(\mathbf{x})\,h^3 = \sum_{m=1}^{M} \mathbf{U}_m\cdot\mathbf{F}_m\Delta\mathbf{s}, \tag{3.10}$$

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. (3.10). To see what Eq. (3.10) implies about $\mathbf{f}(\mathbf{x})$, we rewrite both sides in terms of $\mathbf{a}(\mathbf{x})$. On the left-hand side of Eq. (3.10), we use Eq. (3.2) to obtain

$$\begin{aligned}\sum_{\mathbf{x}\in\mathbb{F}} \mathbf{u}(\mathbf{x})\cdot\mathbf{f}(\mathbf{x})\,h^3 &= \mathbf{u}_0\cdot\sum_{\mathbf{x}\in\mathbb{F}}\mathbf{f}(\mathbf{x})h^3 + \sum_{\mathbf{x}\in\mathbb{F}}(\mathbf{D}^h\times\mathbf{a})(\mathbf{x})\cdot\mathbf{f}(\mathbf{x})\,h^3\\ &= \mathbf{u}_0\cdot\mathbf{f}_0 V + \sum_{\mathbf{x}\in\mathbb{E}}\mathbf{a}(\mathbf{x})\cdot(\mathbf{D}^h\times\mathbf{f})(\mathbf{x})\,h^3,\end{aligned} \tag{3.11}$$

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

$$\mathbf{f}_0 = \frac{1}{V}\sum_{\mathbf{x}\in\mathbb{F}}\mathbf{f}(\mathbf{x})\,h^3. \tag{3.12}$$

Note that we have used the summation-by-parts identity Eq. (A.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. (3.11) is $\mathbb{E}$ not $\mathbb{F}$. On the right-hand side of Eq. (3.10), we substitute for $\mathbf{U}_m$ by using the divergence-free velocity interpolation Eq. (3.6),

$$\begin{aligned}\sum_{m=1}^{M}\mathbf{U}_m\cdot\mathbf{F}_m\Delta\mathbf{s} &= \mathbf{u}_0\cdot\sum_{m=1}^{M}\mathbf{F}_m\Delta\mathbf{s} + \sum_{m=1}^{M}\sum_{\mathbf{x}\in\mathbb{E}}\mathbf{a}(\mathbf{x})\times(\nabla\delta_h)(\mathbf{x}-\mathbf{X}_m)\cdot(\mathbf{F}_m\Delta\mathbf{s})\,h^3\\ &= \mathbf{u}_0\cdot\sum_{m=1}^{M}\mathbf{F}_m\Delta\mathbf{s} + \sum_{\mathbf{x}\in\mathbb{E}}\mathbf{a}(\mathbf{x})\cdot\sum_{m=1}^{M}(\nabla\delta_h)(\mathbf{x}-\mathbf{X}_m)\times(\mathbf{F}_m\Delta\mathbf{s})\,h^3.\end{aligned} \tag{3.13}$$

Since $\mathbf{u}_0$ and $\mathbf{a}(\mathbf{x})$ are arbitrary (except for $\mathbf{D}^h\cdot\mathbf{a}=0$), the power identity Eq. (3.10) is satisfied if and only if

$$\mathbf{f}_0 = \frac{1}{V}\sum_{m=1}^{M}\mathbf{F}_m\Delta\mathbf{s} \tag{3.14}$$

and

$$\mathbf{D}^h\times\mathbf{f} = \sum_{k=1}^{M}(\nabla\delta_h)(\mathbf{x}-\mathbf{X}_m)\times(\mathbf{F}_m\Delta\mathbf{s}) + \mathbf{G}^h\varphi,\quad \text{for all } \mathbf{x}\in\mathbb{E}, \tag{3.15}$$

where $\varphi$ 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. (3.15), since from the identity Eq. (A.4) and $\mathbf{D}^h\cdot\mathbf{a}=0$, we have

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

[2] 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 $\sum_{i=1}^{3}\sum_{\mathbf{x}\in\mathbb{F}}u_i(\mathbf{x})f_i(\mathbf{x})h^3$.

Indeed, we are required to include this term since the left-hand side of Eq. (3.15) is discretely divergence-free but there is no reason to expect the first term on the right-hand side of Eq. (3.15) is also divergence-free. Note that it is not required to find $\varphi$ in order to determine $\mathbf{f}(\mathbf{x})$, because we can eliminate $\varphi$ by taking the discrete curl on both sides of Eq. (3.15),

$$\mathbf{D}^h \times (\mathbf{D}^h \times \mathbf{f}) = \mathbf{D}^h \times \left( \sum_{m=1}^{M} (\nabla \delta_h)(\mathbf{x} - \mathbf{X}_m) \times (\mathbf{F}_m \Delta \mathbf{s}) \right), \quad \text{for all } \mathbf{x} \in \mathbb{E}. \tag{3.16}$$

By imposing the gauge condition

$$\mathbf{D}^h \cdot \mathbf{f} = 0, \tag{3.17}$$

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

$$-(\mathbf{L}^h \mathbf{f})(\mathbf{x}) = \mathbf{D}^h \times \left( \sum_{m=1}^{M} (\nabla \delta_h)(\mathbf{x} - \mathbf{X}_m) \times (\mathbf{F}_m \Delta \mathbf{s}) \right), \quad \text{for all } \mathbf{x} \in \mathbb{E}. \tag{3.18}$$

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. (3.18). Note that the solution of Eq. (3.18) 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.

## 4. Time-stepping scheme

In this section, we present a second-order time-stepping scheme, similar to the ones developed previously [14], that evolves the spatially-discretized system Eqs. (2.24) to (2.26). 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:

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

$$\widetilde{\mathbf{X}}^{n+\frac{1}{2}} = \mathbf{X}^n + \frac{\Delta t}{2} \boldsymbol{S}^\star \left[\mathbf{X}^n\right] \mathbf{u}^n. \tag{4.1}$$

Step 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

$$\mathbf{f}^{n+\frac{1}{2}} = \boldsymbol{S} \left[\widetilde{\mathbf{X}}^{n+\frac{1}{2}}\right] \mathbf{F}^{n+\frac{1}{2}}. \tag{4.2}$$

Step 3. Solve the fluid equations on the periodic grid [7],

$$\begin{cases} \rho \left( \dfrac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t} + \widetilde{\mathbf{N}}^{n+\frac{1}{2}} \right) + \mathbf{G}^h p^{n+\frac{1}{2}} = \mu \mathbf{L}^h \left( \dfrac{\mathbf{u}^{n+1} + \mathbf{u}^n}{2} \right) + \mathbf{f}^{n+\frac{1}{2}}, \\ \mathbf{D}^h \cdot \mathbf{u}^{n+1} = 0, \end{cases} \tag{4.3}$$

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

$$\widetilde{\mathbf{N}}^{n+\frac{1}{2}} = \frac{3}{2}\mathbf{N}^n - \frac{1}{2}\mathbf{N}^{n-1}, \tag{4.4}$$

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

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

$$\mathbf{X}^{n+1} = \mathbf{X}^n + \Delta t\, \boldsymbol{S}^\star \left[\widetilde{\mathbf{X}}^{n+\frac{1}{2}}\right] \left( \frac{\mathbf{u}^{n+1} + \mathbf{u}^n}{2} \right). \tag{4.5}$$

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 $t = \Delta t$, we can use the second-order Runge–Kutta (RK2) scheme described in [34,7].

In Table 1 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

**Table 1**
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.

| | # 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. (4.1) | 1 | – | 3 | – | 2 | 2 | 6 | 3 |
| $\boldsymbol{S}$ in Eq. (4.2) | 2 | – | 3 | – | 2 | 2 | 6 | 3 |
| Fluid solver | 3 | 3 | 4 | 4 | – | – | – | – |
| $\boldsymbol{S}^\star$ in Eq. (4.5) | 1 | – | 3 | – | 2 | 2 | 6 | 3 |
| Total | 7 | 3 | 13 | 4 | 6 | 6 | 18 | 9 |

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 [34,7] 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.

## 5. 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 [19]. 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.

### *5.1. 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 [27,25]. 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 $\Gamma$ 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 [21],

$$E[\mathbf{X}_1, \ldots, \mathbf{X}_M] = \gamma \sum_{m=1}^{M} |\mathbf{X}_m - \mathbf{X}_{m-1}|, \tag{5.1}$$

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

$$\mathbf{F}_m \Delta s = -\frac{\partial E}{\partial \mathbf{X}_m} = \gamma \left( \frac{\mathbf{X}_{m+1} - \mathbf{X}_m}{|\mathbf{X}_{m+1} - \mathbf{X}_m|} - \frac{\mathbf{X}_m - \mathbf{X}_{m-1}}{|\mathbf{X}_m - \mathbf{X}_{m-1}|} \right). \tag{5.2}$$

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

$$\boldsymbol{\mathcal{X}}(s, 0) = L \cdot \left( \frac{1}{2} + \frac{5}{28}\cos(s),\ \frac{1}{2} + \frac{7}{20}\sin(s) \right), \quad s \in [0, 2\pi]. \tag{5.3}$$

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 $\Gamma$ 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

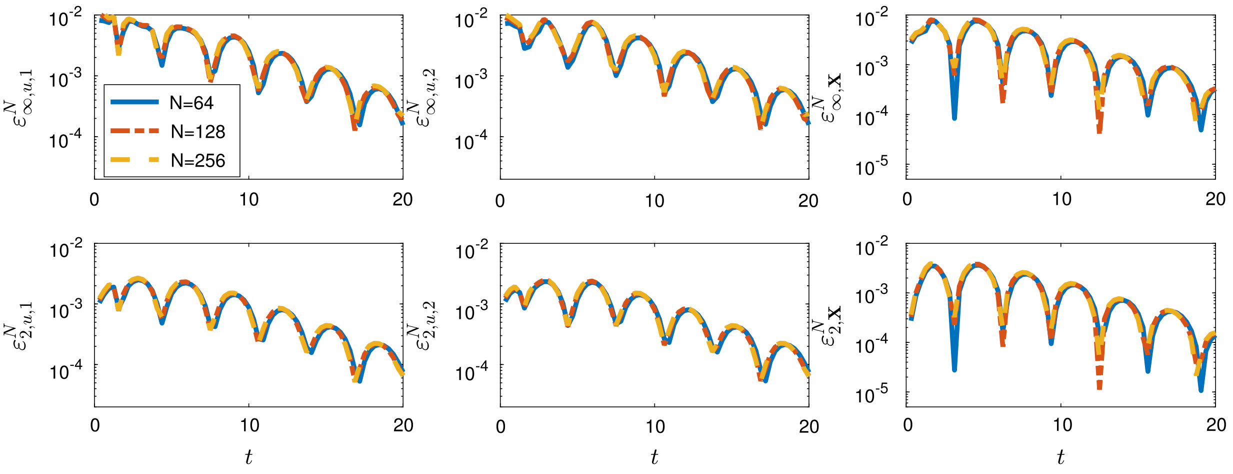


**Fig. 2.** Rescaled $l_\infty$-norm (top panel) and $l_2$-norm (bottom panel) errors of the $x, y$-component of the fluid velocity defined by Eq. (5.4), and errors of the Lagrangian deformation map defined by Eq. (5.5) for the 2D surface tension problem are plotted as a function of time from $t = 0$ 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 $\Delta t = \frac{h}{2}$.

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

$$\varepsilon^N_{p,u,i}(t) = \left\| u^N_i(t) - \mathcal{I}^{2N\to N} u^{2N}_i(t) \right\|_p . \tag{5.4}$$

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,

$$\varepsilon^N_{p,\mathbf{X}}(t) = \left\| \widetilde{\mathbf{X}}^N(t) - \widetilde{\mathbf{X}}^{2N}(t) \right\|_p , \tag{5.5}$$

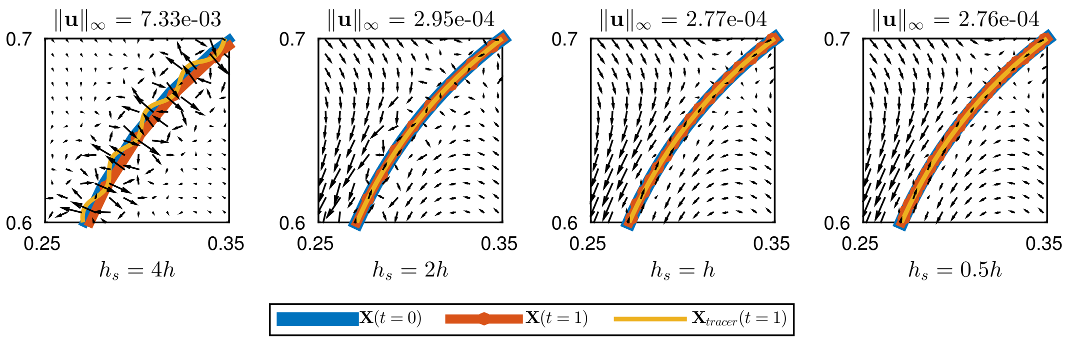
where $M'$ does not change with $N$. We emphasize that the resampled markers are only used to compute the error norm and are discarded after each time step. In Fig. 2 the successive $l_\infty$-norm and $l_2$-norm errors of the $x, y$-component of the fluid velocity and of the deformation map are plotted 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 $\varepsilon^N_{p,\mathbf{X}}(t)$ 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. The observation that all three error curves almost align with each other (as shown in Fig. 2) confirms that second-order convergence in $\mathbf{u}$ and $\mathbf{X}$ is achieved.

## 5.2. Area conservation and IB marker spacing

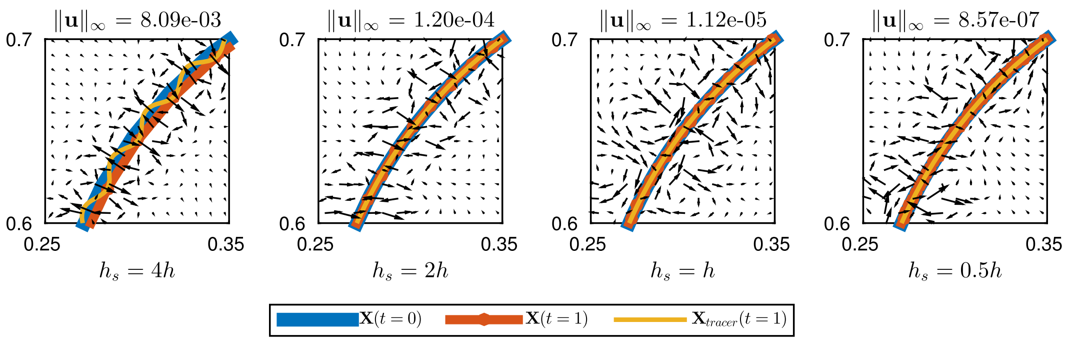
As an immediate consequence of fluid incompressibility, the volume enclosed by a closed immersed boundary should be exactly conserved as it deforms and moves with the fluid. However, it is observed that even in the simplest scenario of a pressurized membrane in its circular equilibrium configuration [17], the volume error of an IB method with conventional interpolation and spreading systematically grows at a rate proportional to the pressure jump across the elastic interface [35]. 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 [17]. A thin elastic membrane $\boldsymbol{\mathcal{X}}(s,t)$, initially in a circular equilibrium configuration,

$$\boldsymbol{\mathcal{X}}(s,0) = \left( \frac{1}{2} + \frac{1}{4}\cos(s), \frac{1}{2} + \frac{1}{4}\sin(s) \right), \quad s \in [0, 2\pi], \tag{5.6}$$



(a) IBMAC



(b) DFIB

**Fig. 3.** 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 $\phi^{\text{new}}_{6h}$ is used in IBMAC and DFIB. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

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

$$\boldsymbol{F}(s,t) = \kappa \frac{\partial^2 \boldsymbol{\mathcal{X}}}{\partial s^2}, \tag{5.7}$$

in which $\kappa$ 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

$$\mathbf{F}_m = \frac{\kappa}{(\Delta s)^2}\left(\mathbf{X}_{m+1} - 2\mathbf{X}_m + \mathbf{X}_{m-1}\right), \tag{5.8}$$

which corresponds to a collection of Lagrangian markers connected by linear springs of zero rest length with stiffness $\kappa$. 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 $128 \times 128$ 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. (5.6), and they move passively with the interpolated velocity according to Eqs. (4.1) and (4.5). 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 $\phi^{\text{new}}_{6h}$ to form the regularized delta function $\delta_h$.

In Fig. 3 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. 3). Each subplot of Fig. 3 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

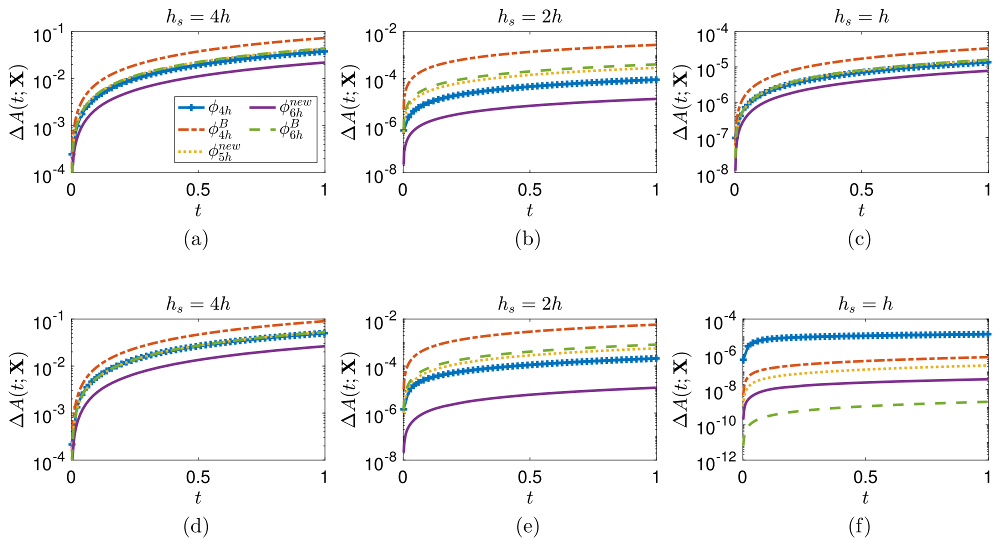


**Fig. 4.** Normalized area errors of the pressurized circular membrane (relative to the initial area, see (5.9)) simulated by IBMAC (top panel) and DFIB (bottom panel) with the IB kernels: $\phi_{4h} \in \mathscr{C}^1$, $\phi_{4h}^{\text{B}} \in \mathscr{C}^2$, $\phi_{5h}^{\text{new}} \in \mathscr{C}^3$, $\phi_{6h}^{\text{new}} \in \mathscr{C}^3$ and $\phi_{6h}^{\text{B}} \in \mathscr{C}^4$, and with Lagrangian marker spacings $h_s \in \{4h, 2h, h\}$ indicated above each figure panel.

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. 3 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 Lagrangian mesh is refined gradually from $h_s = 4h$ to $\frac{h}{2}$ (from left to right in Fig. 3), 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

$$\Delta A(t;\mathbf{X}) := \frac{|A(t;\mathbf{X}) - A(0;\mathbf{X})|}{A(0;\mathbf{X})}, \tag{5.9}$$

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 $t$. Figs. 4 and 5 show the normalized area errors defined by Eq. (5.9) for DFIB and IBMAC with different choices of the IB kernels: $\phi_{4h} \in \mathscr{C}^1$, $\phi_{4h}^{\text{B}} \in \mathscr{C}^2$, $\phi_{5h}^{\text{new}} \in \mathscr{C}^3$, $\phi_{6h}^{\text{new}} \in \mathscr{C}^3$ and $\phi_{6h}^{\text{B}} \in \mathscr{C}^4$. 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. 4a, 4b to Fig. 4d, 4e). 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. 4b, 4c) 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. 4e, 4f), and in the best scenario, $\Delta A(t;\mathbf{X})$ for $\phi_{6h}^{\text{B}}$ 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 $\phi_{6h}^{\text{B}}$ at $h_s = \frac{h}{4}, \frac{h}{8}$ and for $\phi_{6h}^{\text{new}}$ at $h_s = \frac{h}{8}$ (see Fig. 5e, 5f). 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 $h_s \le h$, no matter how densely the Lagrangian mesh is refined (see Figs. 4c, 5c). 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 $h_s \le h$ with $\phi_{6h}^{\text{B}}$, and this kernel also has the highest regularity of the kernel functions considered in this work.

The area errors of DFIB shown in Fig. 4 and Fig. 5 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 $A_{\text{true}}$ by using the tracers,

$$\Delta A(t;\mathbf{X}_{\text{tracer}}) := \frac{|A(t;\mathbf{X}_{\text{tracer}}) - A_{\text{true}}|}{A_{\text{true}}}, \tag{5.10}$$

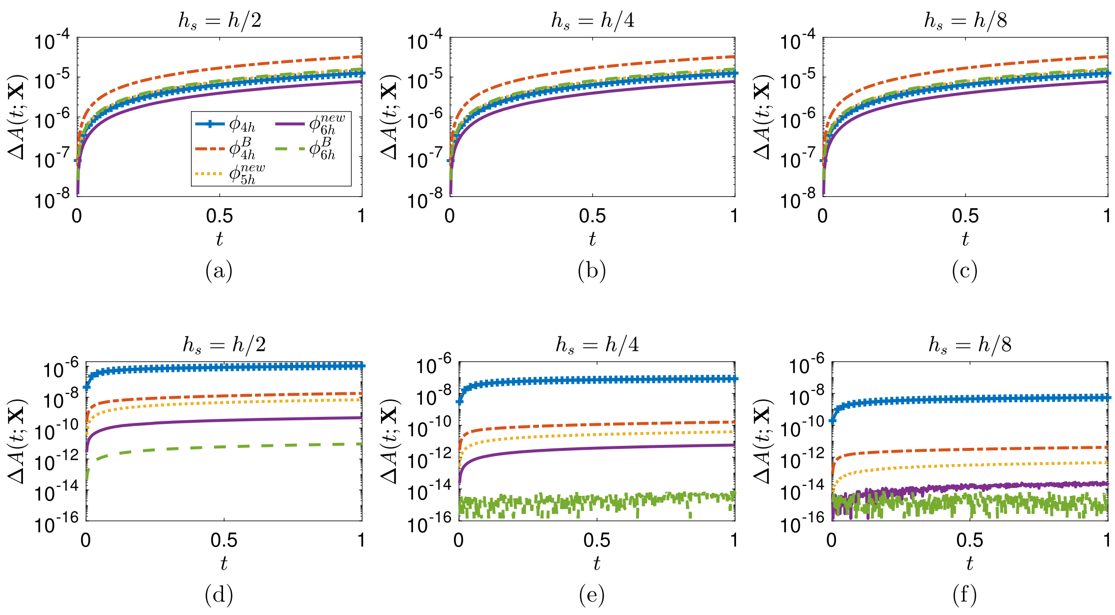


**Fig. 5.** Normalized area errors of the pressurized circular membrane (relative to the initial area, see (5.9)) simulated by IBMAC (top panel) and DFIB (bottom panel) with the IB kernels: $\phi_{4h} \in \mathscr{C}^1$, $\phi_{4h}^{\mathrm{B}} \in \mathscr{C}^2$, $\phi_{5h}^{\mathrm{new}} \in \mathscr{C}^3$, $\phi_{6h}^{\mathrm{new}} \in \mathscr{C}^3$ 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 $\phi_{6h}^{\mathrm{B}}$ at $h_s = \frac{h}{4}, \frac{h}{8}$ and for $\phi_{6h}^{\mathrm{new}}$ at $h_s = \frac{h}{8}$.

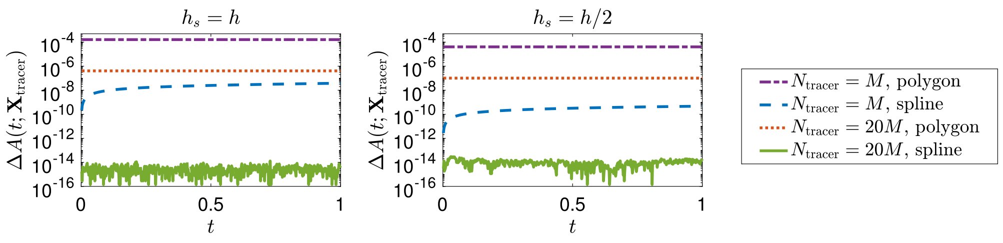


**Fig. 6.** Normalized area errors of the interface enclosed by the tracers that move passively with the interpolated velocity of the DFIB method (relative to the true area of the circle, see Eq. (5.10)). The initial configuration of the interface is given by Eq. (5.6), and $\phi_{6h}^{\mathrm{new}}$ 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.

where we compute $A(t; \mathbf{X}_{\mathrm{tracer}})$ via exact integration of the cubic spline interpolant. Fig. 6 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. 6. Indeed, the area of the spline interpolant through the tracers of resolution $N_{\mathrm{tracer}} = 20M$ 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 [11,44,45,39,29]. 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 [11]. This is inherently linked to the reduced spurious flows compared to traditional methods [39]. In Fig. 7, 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.

## 5.3. 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 IB method may suffer from significant loss in the volume enclosed by the structure [35].

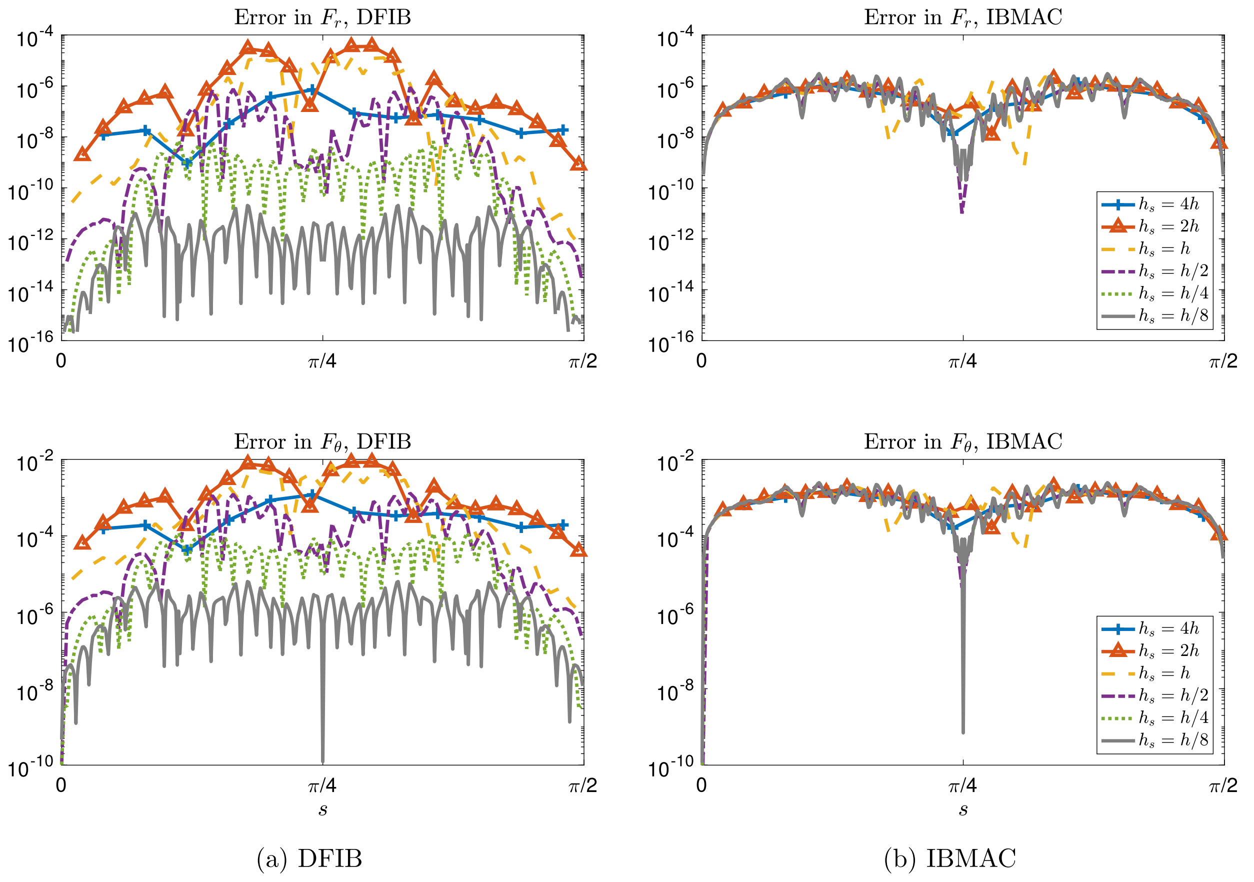


**Fig. 7.** Normalized errors of the normal $F_r$ (top panels) and tangential $F_\theta$ (bottom panels) components of the Lagrangian force $\mathbf{F}(s, t = 1)$ of the circular membrane for $s \in [0, \frac{\pi}{2}]$. The computations are performed using DFIB (left panels) and IBMAC (right panels) with $\phi_{6h}^{\text{new}}$, and with Lagrangian marker spacings $h_s \in \left\{4h, 2h, h, \frac{h}{2}, \frac{h}{4}, \frac{h}{8}\right\}$.

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 [6,23], that is,

$$\boldsymbol{F}(s,t) = \kappa(t)\frac{\partial^2 \boldsymbol{\mathcal{X}}}{\partial s^2}, \tag{5.11}$$

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

$$\kappa(t) = K_c(1 + 2\tau \sin(\omega_0 t)). \tag{5.12}$$

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 $R$,

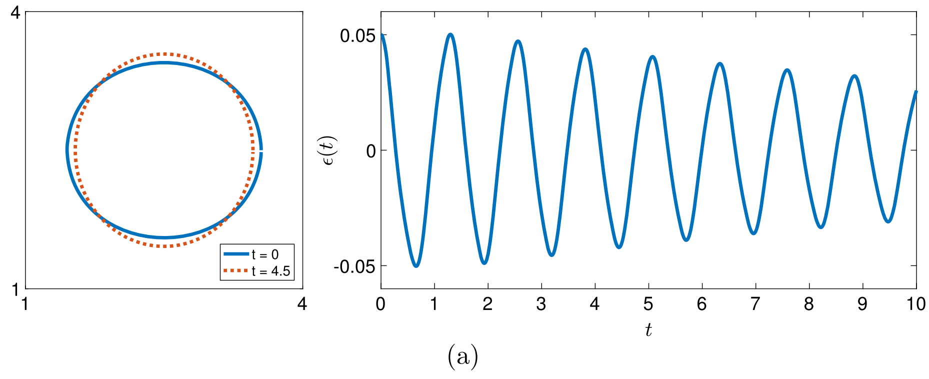
$$\boldsymbol{\mathcal{X}}(s,0) = R(1 + \epsilon_0 \cos(ps))\,\hat{\boldsymbol{r}}(s), \tag{5.13}$$

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 $\omega_0$ 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 [6,23], and recently for an elastic shell in 3D [24]. Motivated by the linear stability analysis of [6,23], we consider two sets of parameters listed in Table 2 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. 8a), and the second set with $\tau = 0.5$ leads to an unstable configuration in which the membrane oscillates with growing amplitude (Fig. 8b).

The computational domain $\Omega = [0, L]^2$ is discretized by a $128 \times 128$ 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

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

| $\rho$ | $\mu$ | $L$ | $R$ | $K_c$ | $\omega_0$ | $p$ | $\epsilon_0$ | $\tau$ |
|---|---|---|---|---|---|---|---|---|
| 1 | 0.15 | 5 | 1 | 10 | 10 | 2 | 0.05 | 0.4 (damped oscillation)<br>0.5 (growing oscillation) |



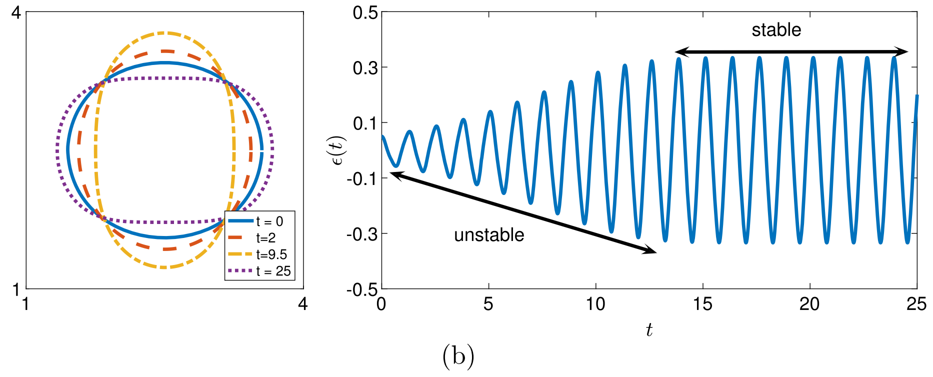


**Fig. 8.** Left panel: snapshots of the 2D membrane with parametric resonance. Right panel: the time-dependent amplitude $\epsilon(t)$ of the perturbed mode in Eq. (5.14). (a) Damped oscillation. (b) Growing oscillation.

initial configuration. The discretization of the Lagrangian force density Eq. (5.11) is constructed in the same way as Eq. (5.8). The time step size is $\Delta t = \frac{h}{10}$ to ensure the stability of computation. On the left panel of Fig. 8 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

$$\boldsymbol{\mathcal{X}}(s,t) = R(1+\epsilon(t)\cos(ps))\,\hat{\mathbf{r}}(s) \tag{5.14}$$

by applying the FFT to the Lagrangian marker positions $\mathbf{X}$. In the case of growing oscillation (Fig. 8b), 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 the cosine kernel $\phi^{\cos}_{4h}$ [35]), 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 $\phi^{\mathrm{B}}_{6h} \in \mathscr{C}^4$. In this test, the area enclosed the Lagrangian markers is computed by the cubic spline approximation discussed in Sec. 5.2. In Fig. 9 and Fig. 10 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. 9), 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 [17], 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 $\phi^{\mathrm{new}}_{6h}$ and $\phi^{\mathrm{B}}_{6h}$ are smaller than those of $\phi^{\mathrm{B}}_{4h}$ and $\phi^{\mathrm{new}}_{5h}$ by approximately one order of magnitude. Additionally, the error curves of DFIB with $\phi^{\mathrm{new}}_{6h}$ and $\phi^{\mathrm{B}}_{6h}$ remain oscillating below $10^{-7}$ while apparent growth of error in time is observed with $\phi^{\mathrm{B}}_{4h}$ and $\phi^{\mathrm{new}}_{5h}$, 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 $N_{\mathrm{tracer}} = 4M$, we

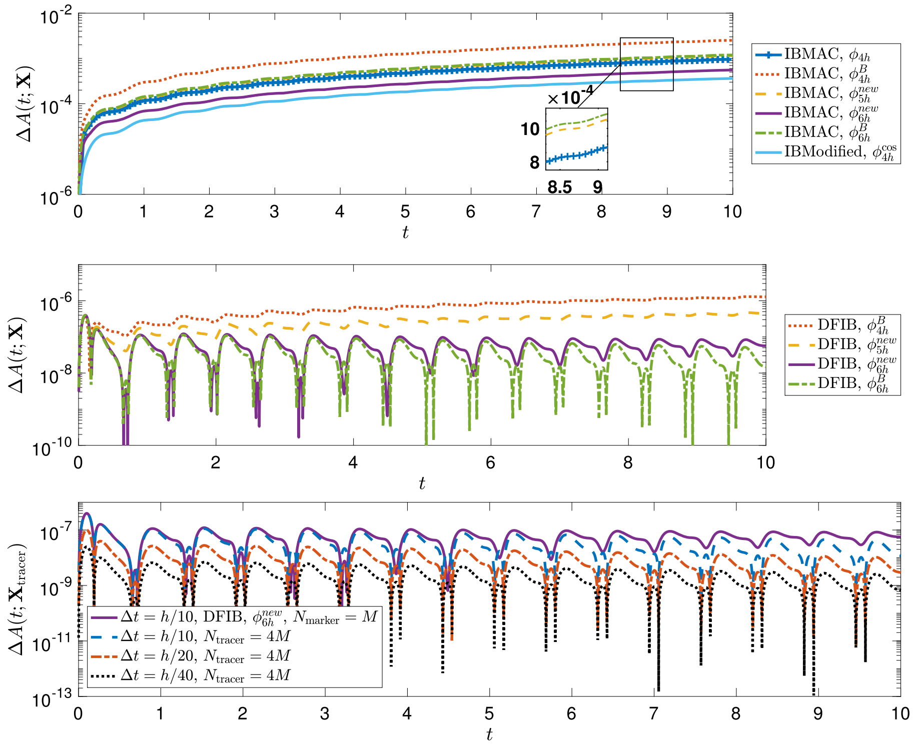


**Fig. 9.** Top and middle panels: normalized area errors $\Delta A(t;\mathbf{X})$ (relative to the initial area, see (5.9)) of the 2D parametric membrane undergoing damped oscillatory motion (corresponding to the motion shown in Fig. 8a) are plotted on the semi-log scale. The area enclosed by the markers 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 (5.10)) enclosed by the interface of $N_{\text{tracer}} = 4M$ tracers by reducing the time step size: $\Delta t \in \left\{\frac{h}{10}, \frac{h}{20}, \frac{h}{40}\right\}$.

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. 9. 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. 10) 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.

## 5.4. 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. 11 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 functional of surface tension in 3D is the product of surface tension constant $\gamma$ (energy per unit area) and the total surface area of the triangular mesh [22], that is,

$$E[\mathbf{X}_1, \dots \mathbf{X}_M] = \gamma \sum_{p=1}^{P} |\mathbf{T}_p|\,, \tag{5.15}$$

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 $E[\mathbf{X}_1, \dots, \mathbf{X}_M]$ with respect to $\mathbf{X}_k$,

$$\mathbf{F}_k \Delta \mathbf{s} = -\frac{\partial E}{\partial \mathbf{X}_k} = -\gamma \sum_{l \in \mathrm{nbor}(k)} \frac{\partial |\mathbf{T}_l|}{\partial \mathbf{X}_k}\,, \tag{5.16}$$

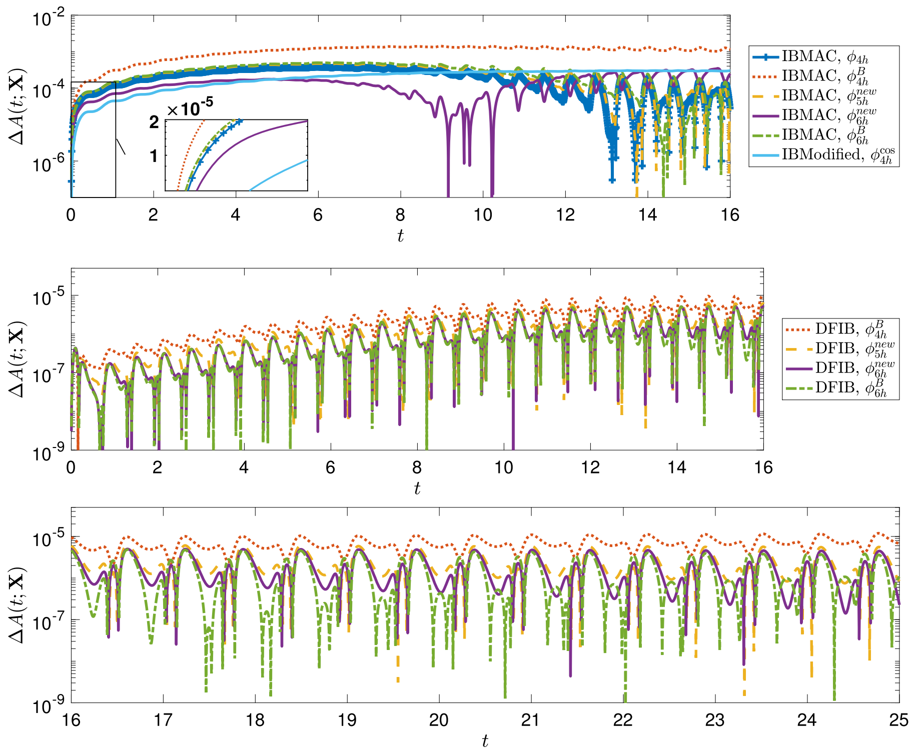


**Fig. 10.** Normalized area errors $\Delta A(t;\mathbf{X})$ of the 2D parametric membrane undergoing growing-amplitude oscillatory motion (corresponding to the motion shown in Fig. 8b) are plotted on the semi-log scale, as done in Fig. 9 for damped motion. The computations are performed using IBMAC 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, 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.

where nbor($k$) denotes the set of indices of triangles that share $\mathbf{X}_k$ as a vertex.[3] Each component of the term $\partial|\mathbf{T}_l|/\partial\mathbf{X}_k$ in Eq. (5.16) can be computed analytically [22],

$$\left(\frac{\partial|\mathbf{T}_l|}{\partial\mathbf{X}_k}\right)_\alpha = \frac{\partial}{\partial\mathbf{X}_{k,\alpha}}\left(\frac{1}{2}\left|(\mathbf{X}_k - \mathbf{X}'_k) \times (\mathbf{X}'_k - \mathbf{X}''_k)\right|\right)$$
$$= \frac{1}{2}\left((\mathbf{X}'_k - \mathbf{X}''_k) \times \hat{\mathbf{n}}_l\right)_\alpha, \quad \alpha = 1, 2, 3, \tag{5.17}$$

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 $\phi_{6h}^{\mathrm{new}}$. 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. 12, the spherical membrane is discretized by triangulation (as shown in Fig. 11) with 5 successive levels of refinement from the regular icosahedron (Fig. 11a), which results in a triangular mesh with $M = 10242$ vertices and $P = 20480$ facets. The radius 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. 12 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

$$\Delta V(t;\mathbf{X}) := \frac{|\mathrm{Vol}(t;\mathbf{X}) - \mathrm{Vol}(0;\mathbf{X})|}{\mathrm{Vol}(0;\mathbf{X})} \tag{5.18}$$

[3] 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.

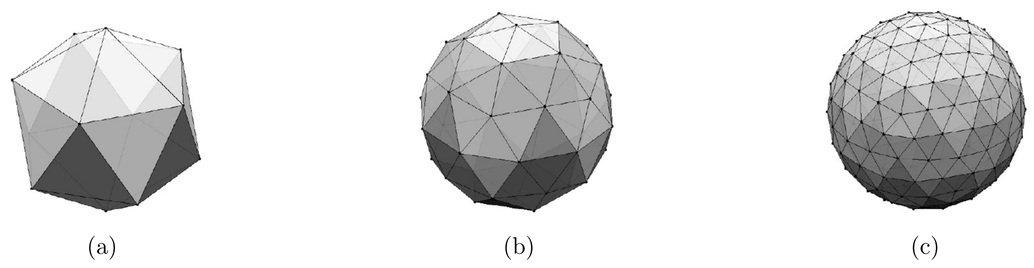


**Fig. 11.** Triangulation of a spherical surface mesh via refinement of a regular icosahedron. (a) Regular icosahedron. (b) Refined mesh after one level of refinement. (c) Refined mesh after two levels of refinement.

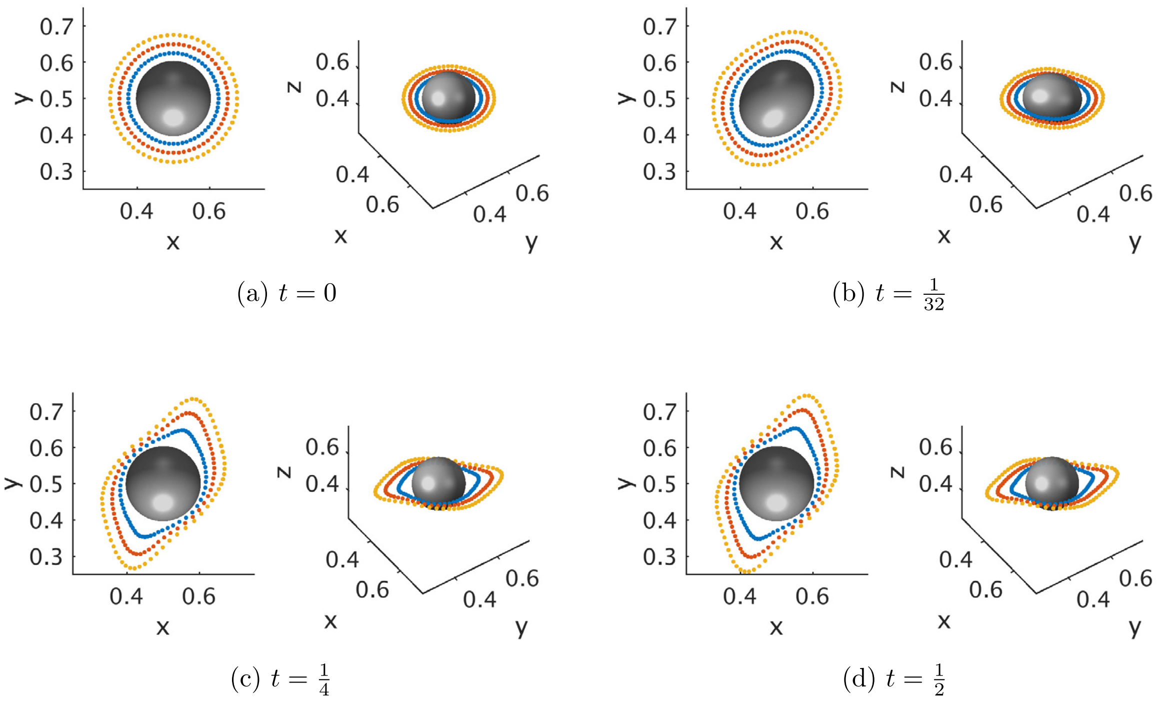


**Fig. 12.** Deformation of a 3D elastic membrane immersed in a viscous fluid with initial velocity $\boldsymbol{u}(\boldsymbol{x},t)=(0,\ \sin(4\pi x),\ 0)$ at $t=0, \frac{1}{32}, \frac{1}{4}$ and $\frac{1}{2}$. The computation is performed using DFIB with $\phi_{6h}^{\text{new}}$ in the periodic box $\Omega=[0,1]^3$ 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.

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. 13a), 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. 13b), 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. 13 can also be explained by contribution from 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. 13). 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,

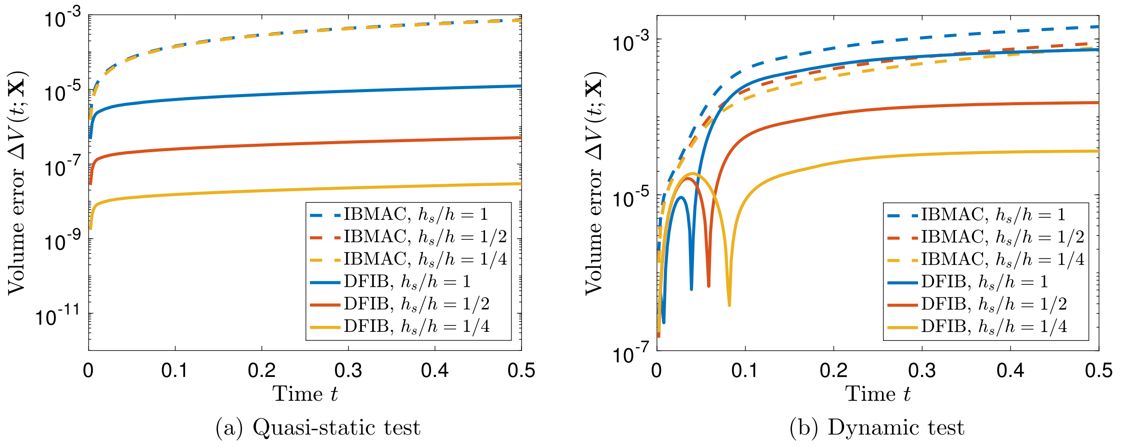


**Fig. 13.** Normalized volume error $\Delta V(t;\mathbf{X})$ of a 3D elastic membrane using IBMAC and DFIB with $h_s = h$, $\frac{h}{2}$, $\frac{h}{4}$, where $h = \frac{1}{128}$. For (a) the quasi-static test, $\Delta V(t,\mathbf{X})$ of DFIB decreases with mesh refinement, while there is no improvement 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 $\phi_{6h}^{\text{new}}$ is used.

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 ($h_s \leq \frac{h}{2}$).

## 6. Conclusions

In this paper, 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 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 $\delta_h$, 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 paper 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 [19,10,18,4]. 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 [9].

Unlike other improved IB methods that either use non-standard finite-difference operators (IBModified [35]) that complicate the implementation of the fluid solver, or rely on analytically-computed correction terms (IIM [28,27] or Blob-Projection method [5]) 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 [16,20]. 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 [29]. 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 [13]. 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 [13].

## Acknowledgements

Y. Bao and A. Donev were supported in part by the National Science Foundation under award DMS-1418706, and by the U.S. Department of Energy Office of Science, Office of Advanced Scientific Computing Research, Applied Mathematics program under Award Number DE-SC0008271. B.E. Griffith acknowledges research support from the National Science Foundation (NSF awards ACI 1450327, DMS 1410873, and CBET 1511427) and the National Institutes of Health (NIH award HL117063).

## Appendix A. 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,

$$\mathbf{D}^h \times \mathbf{G}^h \varphi = 0, \tag{A.1}$$

$$\mathbf{D}^h \cdot (\mathbf{D}^h \times \mathbf{u}) = 0, \tag{A.2}$$

$$\mathbf{D}^h \times (\mathbf{D}^h \times \mathbf{u}) = \mathbf{G}^h (\mathbf{D}^h \cdot \mathbf{u}) - \mathbf{L}^h \mathbf{u}, \tag{A.3}$$

$$\sum_{\mathbf{x}\in\mathbb{F}} \mathbf{u}(\mathbf{x}) \cdot (\mathbf{G}^h \varphi)(\mathbf{x}) h^3 = -\sum_{\mathbf{x}\in\mathbb{C}} (\mathbf{D}^h \cdot \mathbf{u})(\mathbf{x})\, \varphi(\mathbf{x}) h^3, \tag{A.4}$$

$$\sum_{\mathbf{x}\in\mathbb{E}} \mathbf{a}(\mathbf{x}) \cdot (\mathbf{D}^h \times \mathbf{u})(\mathbf{x}) h^3 = \sum_{\mathbf{x}\in\mathbb{F}} (\mathbf{D}^h \times \mathbf{a})(\mathbf{x}) \cdot \mathbf{u}(\mathbf{x}) h^3. \tag{A.5}$$

Eqs. (A.1) and (A.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. (A.4) and (A.5) can be verified via "summation by parts". Note that Eqs. (A.2) and (A.3) also hold if we replace $\mathbf{u}$ (which lives on $\mathbb{F}$) by $\mathbf{a}$ (which lives on $\mathbb{E}$).

## Appendix B. 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. (A.3) to

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

Thus,

$$\sum_{\substack{\mathbf{x}\in\mathbb{E},i\neq j\\ \mathbf{x}\in\mathbb{C},i=j}} \left(D_j^h u_i\right)^2 h^3 = \sum_{\mathbf{x}\in\mathbb{E}} \left|(\mathbf{D}^h \times \mathbf{u})\right|^2 h^3 + \sum_{\mathbf{x}\in\mathbb{E}} (\mathbf{D}^h \cdot \mathbf{u})^2 h^3. \tag{B.1}$$

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

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

$$\sum_{\mathbf{x}} \psi(\mathbf{x}) h^3 = 0, \tag{B.2}$$

*then there exists a grid function $\varphi$ such that*

$$L^h \varphi = \psi. \tag{B.3}$$

**Proof.** This lemma states the solvability of the discrete Poisson problem Eq. (B.3). Since $L^h = \mathbf{D}^h \cdot \mathbf{G}^h$ is symmetric with respect to the inner product on the periodic grid

$$(\varphi, \psi) = \sum_{\mathbf{x}} \varphi(\mathbf{x})\psi(\mathbf{x}) h^3, \tag{B.4}$$

what we have to show is that any $\psi$ satisfying Eq. (B.2) is orthogonal to any $\varphi_0$ 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. (B.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*

$$\sum_{\mathbf{x}\in\mathbb{F}} \mathbf{u}(\mathbf{x}) h^3 = 0 \quad \textit{and} \quad \mathbf{D}^h \cdot \mathbf{u} = 0, \tag{B.5}$$

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

$$\mathbf{u} = \mathbf{D}^h \times \mathbf{a}. \tag{B.6}$$

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

$$-\mathbf{L}^h \mathbf{a} = \mathbf{D}^h \times \mathbf{u}. \tag{B.7}$$

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. (B.7) and using the property that $\mathbf{L}^h$ and $\mathbf{D}^h \cdot$ commute, we also have

$$-\mathbf{L}^h (\mathbf{D}^h \cdot \mathbf{a}) = \mathbf{D}^h \cdot (\mathbf{D}^h \times \mathbf{u}) = 0. \tag{B.8}$$

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

$$\mathbf{G}^h (\mathbf{D}^h \cdot \mathbf{a}) = 0. \tag{B.9}$$

If we use Eq. (B.9) and Eq. (A.3) for $\mathbf{a}$, we can rewrite Eq. (B.7) as

$$\mathbf{D}^h \times (\mathbf{D}^h \times \mathbf{a}) = \mathbf{D}^h \times \mathbf{u}, \tag{B.10}$$

or

$$\mathbf{D}^h \times (\mathbf{D}^h \times \mathbf{a} - \mathbf{u}) = 0. \tag{B.11}$$

But we also know from Eq. (A.2) and the requirement that $\mathbf{D}^h \cdot \mathbf{u} = 0$ that

$$\mathbf{D}^h \cdot (\mathbf{D}^h \times \mathbf{a} - \mathbf{u}) = 0. \tag{B.12}$$

From Eqs. (B.11) and (B.12) and Lemma 1, it follows that

$$\mathbf{D}^h \times \mathbf{a} - \mathbf{u} = \text{constant}. \tag{B.13}$$

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

## References

[1] Y. Bao, A.D. Kaiser, J. Kaye, C.S. Peskin, Gaussian-like immersed boundary kernels with three continuous derivatives and improved translational invariance, arXiv:1505.07529, 2015.

[2] Y. Bao, J. Kaye, C.S. Peskin, A Gaussian-like immersed-boundary kernel with three continuous derivatives and improved translational invariance, J. Comput. Phys. 316 (2016) 139–144.

[3] 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, J. Comput. Phys. 250 (2013) 446–476.

[4] D. Boffi, L. Gastaldi, L. Heltai, C. Peskin, On the hyper-elastic formulation of the immersed boundary method, Comput. Methods Appl. Mech. Eng. 197 (25–28) (2008) 2210–2231.

[5] R. Cortez, M. Minion, The blob projection method for immersed boundary problems, J. Comput. Phys. 161 (2) (2000) 428–453.

[6] R. Cortez, C.S. Peskin, J.M. Stockie, D. Varela, Parametric resonance in immersed elastic boundaries, SIAM J. Appl. Math. 65 (2) (2004) 494–520.

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

[8] A. Dutt, V. Rokhlin, Fast Fourier transforms for nonequispaced data, SIAM J. Sci. Comput. 14 (6) (1993) 1368–1393.

[9] 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 J. Sci. Comput. 35 (5) (2013) B1132–B1161.

[10] 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, Int. J. Numer. Methods Biomed. Eng. 30 (11) (2014) 1199–1222.

[11] A. Goza, S. Liska, B. Morley, T. Colonius, Accurate computation of surface stresses and forces with immersed boundary methods, J. Comput. Phys. 321 (2016) 860–873.

[12] L. Greengard, J. Lee, Accelerating the nonuniform fast Fourier transform, SIAM Rev. 46 (3) (2004) 443–454.

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

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

[15] 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.

[16] B. Griffith, X. Luo, D. McQueen, C. Peskin, Simulating the fluid dynamics of natural and prosthetic heart valves using the immersed boundary method, Int. J. Appl. Mech. 1 (01) (2009) 137–177.

[17] B.E. Griffith, On the volume conservation of the immersed boundary method, Commun. Comput. Phys. 12 (2) (2012) 401–432.

[18] B.E. Griffith, X. Luo, Hybrid finite difference/finite element immersed boundary method, Int. J. Numer. Methods Biomed. Eng. (2017), http://dx.doi.org/10.1002/Cnm.2888.

[19] B.E. Griffith, C.S. Peskin, On the order of accuracy of the immersed boundary method: higher order convergence rates for sufficiently smooth problems, J. Comput. Phys. 208 (1) (2005) 75–105.

[20] B. Kallemov, A.P.S. Bhalla, B.E. Griffith, A. Donev, An immersed boundary method for rigid bodies, Commun. Appl. Math. Comput. Sci. 11 (1) (2016) 79–141, software available at https://github.com/stochasticHydroTools/RigidBodyIB.

[21] Y. Kim, M.-C. Lai, C.S. Peskin, Numerical simulations of two-dimensional foam by the immersed boundary method, J. Comput. Phys. 229 (13) (2010) 5194–5207.

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

[23] W. Ko, J.M. Stockie, Correction to "Parametric resonance in immersed elastic boundaries", arXiv:1207.4744, 2012.

[24] W. Ko, J.M. Stockie, Parametric resonance in spherical immersed elastic shells, SIAM J. Appl. Math. 76 (1) (2016) 58–86.

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

[26] M.-C. Lai, C.S. Peskin, An immersed boundary method with formal second-order accuracy and reduced numerical viscosity, J. Comput. Phys. 160 (2) (2000) 705–719.

[27] L. Lee, R.J. LeVeque, An immersed interface method for incompressible Navier–Stokes equations, SIAM J. Sci. Comput. 25 (3) (2003) 832–856.

[28] Z. Li, M.-C. Lai, The immersed interface method for the Navier–Stokes equations with singular forces, J. Comput. Phys. 171 (2) (2001) 822–842.

[29] S. Liska, T. Colonius, A fast immersed boundary method for external incompressible viscous flows using lattice Green's functions, J. Comput. Phys. 331 (2017) 257–279.

[30] E. Lushi, C.S. Peskin, Modeling and simulation of active suspensions containing large numbers of interacting micro-swimmers, Comput. Struct. 122 (2013) 239–248.

[31] 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, J. Supercomput. 11 (3) (1997) 213–236.

[32] C.S. Peskin, Flow patterns around heart valves: a numerical method, J. Comput. Phys. 10 (2) (1972) 252–271.

[33] C.S. Peskin, Numerical analysis of blood flow in the heart, J. Comput. Phys. 25 (3) (1977) 220–252.

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

[35] C.S. Peskin, B.F. Printz, Improved volume conservation in the computation of flows with immersed elastic boundaries, J. Comput. Phys. 105 (1) (1993) 33–46.

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

[37] 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, J. Comput. Phys. 304 (2016) 252–274.

[38] 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, J. Comput. Phys. 335 (2017) 155–178.
[39] W. Strychalski, R.D. Guy, Intracellular pressure dynamics in blebbing cells, Biophys. J. 110 (5) (2016) 1168–1179.
[40] M. Unser, Splines: a perfect fit for signal and image processing, IEEE Signal Process. Mag. 16 (6) (1999) 22–38.
[41] M. Unser, A. Aldroubi, M. Eden, On the asymptotic convergence of B-spline wavelets to Gabor functions, IEEE Trans. Inf. Theory 38 (2) (1992) 864–872.
[42] 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, Multiscale Model. Simul. 10 (4) (2012) 1369–1408.
[43] 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, Commun. Appl. Math. Comput. Sci. 11 (2) (2016) 217–296, software available at https://github.com/stochasticHydroTools/RotationalDiffusion.
[44] H.A. Williams, L.J. Fauci, D.P. Gaver III, Evaluation of interfacial fluid dynamical stresses using the immersed boundary method, Discrete Contin. Dyn. Syst., Ser. B 11 (2) (2009) 519.
[45] 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, J. Comput. Phys. 228 (20) (2009) 7821–7836.