Available online at www.sciencedirect.com
ScienceDirect
Comput. Methods Appl. Mech. Engrg. 197 (2008) 2049–2067

Computer methods in applied mechanics and engineering
www.elsevier.com/locate/cma

Implicit second-order immersed boundary methods with boundary mass

Yoichiro Mori [a,*], Charles S. Peskin [b]

[a] Department of Mathematics, University of British Columbia, 1984 Mathematics Road, Vancouver BC, Canada
[b] Courant Institute of Mathematical Sciences, New York University, 251 Mercer Street, New York, NY, USA

Received 22 February 2007; received in revised form 26 May 2007; accepted 29 May 2007
Available online 10 August 2007

Abstract

The immersed boundary method is a computational framework for problems involving the interaction of a fluid and immersed elastic structures. Immersed boundary computations typically evaluate the elastic forces explicitly in the configuration of the immersed elastic structure. In many applications this results in a severe restriction on the time step. We present a semi-implicit and a fully implicit second-order accurate immersed boundary method. The methods provide a natural way to handle mass on the immersed elastic structures. We demonstrate their performance for a prototypical fluid–structure interaction problem. The methods are shown to possess superior stability properties that significantly alleviate the typically severe time step restriction of explicit computations.

Keywords: Immersed boundary method; Fluid–structure interaction; Implicit method; Boundary with mass

1. Introduction

Many problems of biofluid dynamics involve the interaction of an immersed elastic structure with an underlying fluid. The immersed boundary method is a generally useful method to solve such problems. The method has been successfully applied to a variety of problems including blood flow in the heart [1], vibrations of the cochlear basilar membrane [2,3], blood clotting [4–6], aquatic locomotion [7–10], insect flight [11,12], flow with suspended particles [13,14] and other physical problems [15–18]. We refer to [19] for a more extensive list of applications.

The immersed boundary method uses an Eulerian description for the velocity field of the fluid and a Lagrangian description for the configuration of the immersed elastic structure. The force generated by the immersed elastic structure drives the fluid flow and the fluid flow in turn moves the immersed elastic structure to a new configuration. This interaction between the fluid and the immersed elastic structure is expressed in terms of spreading and interpolation operations by use of smoothed Dirac delta functions [19].

The immersed elastic structure generates a force **F** which is a function of its configuration **X**. The time stepping scheme of most immersed boundary computations uses an explicit evaluation of **F** in terms of **X**. In many applications, this results in a severe restriction on the time step [19,20]. There have been attempts to develop implicit schemes to overcome this difficulty [21,22,4]. We also note a recent effort in [23]. Implicit immersed boundary methods have also been introduced in the context of finite element discretizations [24–26]. In this paper, we present two second-order accurate implicit immersed boundary methods, one of which is semi-implicit and the other fully implicit, which are built upon an implicit first-order method proposed in [22]. The bulk of the computational work is spent on the solution of linear equations, which are solved using a Krylov subspace method. As we shall see, each Krylov subspace iteration is approximately equal in computational cost to an explicit time step.

We show that both methods presented here attain approximate second-order accuracy for smooth problems.

[*] Corresponding author.
E-mail addresses: mori@math.ubc.ca (Y. Mori), peskin@cims.nyu.edu (C.S. Peskin).

0045-7825/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.cma.2007.05.028

Most convergence studies of the immersed boundary method have used an infinitely thin elastic membrane in test problems. The analytical solution to such problems have discontinuities in the derivative of the velocity field. This lack of smoothness makes it difficult for immersed boundary computations to attain its full formal accuracy. Following [27] we demonstrate second-order accuracy of the present methods with a smooth problem where the elastic membrane is replaced by an elastic shell with tapered elastic stiffness and tapered mass density.

We demonstrate by way of numerical experiment that the two implicit methods have superior stability properties. For any fixed spatial grid, the time step restriction is far less severe in both implicit methods in comparison to an explicit calculation. Moreover, the time step restrictions for the implicit methods do not depend on the spatial grid resolution. For the model problem treated here, the time step needs to be roughly halved each time the grid spacing is halved when an explicit method is used. In the implicit methods, the iteration count for the Krylov subspace method stays roughly the same with the refinement of the spatial grid. This makes the implicit methods increasingly efficient in comparison to an explicit method as the spatial grid is refined.

The implicitness of the methods give us a natural way to incorporate mass of the immersed elastic structure at minimal additional computational cost. In the original immersed boundary method, the immersed elastic structure is treated as neutrally buoyant in the fluid. Several different methods have been proposed and used successfully to handle mass of the immersed elastic structure [28–30]. The way mass is handled here has some attractive features which we believe adds further utility to the present methods.

In Section 2, we present the model problem and its immersed boundary formulation. In Section 3, we review spatial discretization and discuss how we deal with mass on the immersed elastic structure. In Section 4, we introduce the semi-implicit and fully implicit methods. We then discuss the solution of the linear equations in Section 5. This is followed by computational results in Section 6. In the final section, we summarize the characteristics of the new methods.

2. Model problem

We consider a two-dimensional viscoelastic fluid. A schematic of the model problem we consider is given in Fig. 1. Let the fluid domain be denoted by $U \subset \mathbb{R}^2$. The velocity and pressure fields $\mathbf{u}(\mathbf{x},t)$ and $p(\mathbf{x},t)$ are functions of the fixed cartesian coordinate $\mathbf{x} = (x_1,x_2) = (x,y) \in U$. We shall also introduce Lagrangian markers $\mathbf{X}(\boldsymbol{\xi},t)$, parametrized by a curvilinear coordinate system $\boldsymbol{\xi} = (\xi_1,\xi_2) = (\eta,\theta) \in \Omega \subset \mathbb{R}^2$. We may view $\mathbf{X} = (X_1,X_2)$ as a time-dependent map from the Lagrangian material coordinates in Ω to their cartesian positions in U.

The image of Ω, $\mathbf{X}(\Omega,t)$ may be the whole domain U or a proper subset of U. The region $\mathbf{X}(\Omega,t)$ will be where the material is not only viscous but also elastic. In the present paper, the region $\mathbf{X}(\Omega,t)$ may also carry an excess mass density in addition to the background uniform mass density.

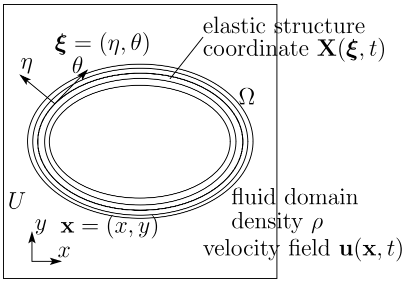
$\boldsymbol{\xi} = (\eta,\theta)$
elastic structure
coordinate $\mathbf{X}(\boldsymbol{\xi},t)$
η
θ
Ω
U
fluid domain
density ρ
velocity field $\mathbf{u}(\mathbf{x},t)$
y
$\mathbf{x} = (x,y)$
x

Fig. 1. A schematic diagram of the model problem.

We start with the following familiar equations of continuum mechanics:

$$\tilde{\rho}\frac{\mathrm{D}\mathbf{u}}{\mathrm{D}t} = \nabla \cdot \sigma, \tag{1}$$

$$\nabla \cdot \mathbf{u} = 0, \tag{2}$$

$$\frac{\mathrm{D}\tilde{\rho}}{\mathrm{D}t} = -\tilde{\rho}\nabla \cdot \mathbf{u}. \tag{3}$$

Eq. (1) expresses momentum balance, where $\tilde{\rho}$ is the mass density of the viscoelastic continuum, $\mathrm{D}/\mathrm{D}t$ the material derivative and σ the stress tensor. Eq. (2) is the incompressibility condition and Eq. (3) expresses mass balance.

We must supply the above with a constitutive relation for the stress tensor σ. We consider the following:

$$\sigma = -pI + \mu(\nabla\mathbf{u} + (\nabla\mathbf{u})^{\mathrm{T}}) + \sigma_{\mathrm{el}}, \tag{4}$$

where p is the pressure, I is the identity tensor, and μ is the viscosity and σ_{el} is the elastic stress to be specified later. The elastic stress tensor σ_{el} is 0 in $U - \mathbf{X}(\Omega,t)$. The stress is thus a sum of an isotropic pressure, a viscous stress and an elastic stress. The pressure p is determined by the incompressibility condition (2). Given the constitutive relation (4), we may consider the material in $\mathbf{X}(\Omega,t)$ to be composed of a viscous Newtonian fluid and a volumeless *immersed elastic structure* which generates elastic stress. Substituting (4) into (1), we have

$$\tilde{\rho}\frac{\mathrm{D}\mathbf{u}}{\mathrm{D}t} + \nabla p = \mu\Delta\mathbf{u} + \nabla \cdot \sigma_{\mathrm{el}}. \tag{5}$$

We shall make use of a Lagrangian description for the mass density $\tilde{\rho}$ and the elastic stress σ_{el}. We first make note of the following relation that links the Lagrangian and Eulerian descriptions:

$$\frac{\partial\mathbf{X}}{\partial t} = \mathbf{u}. \tag{6}$$

Write $\tilde{\rho}$ as a sum of a spatially uniform constant ρ and an additional spatially varying component carried with the immersed elastic structure. In $U-\mathbf{X}(\Omega,t)$, we let $\tilde{\rho}=\rho$, a uniform constant. In $\mathbf{X}(\Omega,t)$ we let,

$$\tilde{\rho}=\rho+M|\mathbb{F}|^{-1}, \tag{7}$$

where $\mathbb{F}$ is the deformation matrix $\frac{\partial\mathbf{X}}{\partial\boldsymbol{\xi}}$ and $M(\boldsymbol{\xi},t)$ is the mass density measured in the material coordinate $\boldsymbol{\xi}$. The term $M|\mathbb{F}|^{-1}$ is a Lagrangian variable whereas ρ and $\tilde{\rho}$ are Eulerian variables. In Eq. (7) as well as in the sequel, whenever both Eulerian and Lagrangian quantities appear in a single equation, the Lagrangian terms will be understood as being evaluated at the coordinate $\boldsymbol{\xi}$ corresponding to the coordinate $\mathbf{x}$ at which the Eulerian terms are evaluated. Note in (7) that we need to multiply M with the inverse of the determinant of the deformation tensor $|\mathbb{F}|^{-1}$ to obtain the mass density in the Eulerian coordinate. Now, substitute (7) into (3). Making use of the incompressibility condition (2) in Eulerian ($\nabla\cdot\mathbf{u}=0$) and Lagrangian $\left(\frac{\partial|\mathbb{F}|}{\partial t}=0\right)$ forms, we see that (3) reduces to $\frac{\partial M}{\partial t}=0$. Thus, $M(\boldsymbol{\xi})$ is constant in time. We note that $|\mathbb{F}|$ is not necessarily equal to 1 uniformly, since we are to use curvilinear linear coordinates for the Lagrangian mesh.

Next, consider the elastic stress tensor. In $U-\mathbf{X}(\Omega,t)$, σ_{el} is taken to be 0. In $\mathbf{X}(\Omega,t)$ the stress tensor may be nonzero, and there we have a material coordinate system $\boldsymbol{\xi}$. Let the stress tensor have the following form:

$$(\sigma_{\mathrm{el}})_{ij}=\widetilde{T}\tau_i\tau_j,\quad \boldsymbol{\tau}=\frac{\partial\mathbf{X}}{\partial\theta}\Big/\left|\frac{\partial\mathbf{X}}{\partial\theta}\right|, \tag{8}$$

where $(\sigma_{\mathrm{el}})_{ij}$ denotes the ij component of the stress tensor, $\widetilde{T}$ is some scalar function of $\mathbf{x}$ and τ_i denotes the ith component of the unit coordinate vector $\boldsymbol{\tau}$. This stress tensor expresses an immersed elastic material which can be regarded as a *fiber continuum*. The material coordinate θ runs along the fibers, and the local fiber direction is given by the unit coordinate vector $\boldsymbol{\tau}$. The local strength of the fiber is specified by $\widetilde{T}$.

We would now like to rewrite the elastic forcing term $\nabla\cdot\sigma_{\mathrm{el}}$ in Eq. (5) using material coordinates

$$\begin{aligned}\frac{\partial(\sigma_{\mathrm{el}})_{ij}}{\partial x_j}&=\frac{\partial}{\partial x_j}(\widetilde{T}\tau_i\tau_j)=\frac{\partial}{\partial x_j}\left(T\tau_i|\mathbb{F}|^{-1}\frac{\partial X_j}{\partial\theta}\right)\\&=\frac{\partial}{\partial\theta}(T\tau_i)|\mathbb{F}|^{-1}+T\tau_i\frac{\partial}{\partial x_j}\left(|\mathbb{F}|^{-1}\frac{\partial X_j}{\partial\theta}\right),\end{aligned} \tag{9}$$

where repeated indices are to be summed, and $T\equiv\widetilde{T}|\mathbb{F}|/\left|\frac{\partial\mathbf{X}}{\partial\theta}\right|$. We would like to show that the second term in the last line of the above is equal to 0. Note

$$|\mathbb{F}|^{-1}\frac{\partial X_1}{\partial\theta}=\frac{\frac{\partial X_1}{\partial\theta}}{\frac{\partial X_1}{\partial\eta}\frac{\partial X_2}{\partial\theta}-\frac{\partial X_2}{\partial\eta}\frac{\partial X_1}{\partial\theta}}=-(\mathbb{F}^{-1})_{\eta X_2}=-\frac{\partial\eta}{\partial X_2}, \tag{10}$$

where $(\mathbb{F}^{-1})_{\eta X_2}$ refers to the ηX_2 component of the matrix $\mathbb{F}^{-1}$. Likewise, $|\mathbb{F}|^{-1}\frac{\partial X_2}{\partial\theta}=\frac{\partial\eta}{\partial X_1}$. We immediately see that

$$\frac{\partial}{\partial x_j}\left(|\mathbb{F}|^{-1}\frac{\partial X_j}{\partial\theta}\right)=0 \tag{11}$$

by the equality of mixed derivatives. We therefore conclude that

$$\nabla\cdot\sigma_{\mathrm{el}}=\frac{\partial}{\partial\theta}(T\boldsymbol{\tau})|\mathbb{F}|^{-1},\quad T=\widetilde{T}|\mathbb{F}|\Big/\left|\frac{\partial\mathbf{X}}{\partial\theta}\right|. \tag{12}$$

The above equality can also be derived by a force balance argument, by noting that $\widetilde{T}$ is the product of T, the magnitude of the tension per fiber, and $\left|\frac{\partial\mathbf{X}}{\partial\theta}\right|/|\mathbb{F}|$, the fiber density in the cross-sectional plane perpendicular to $\boldsymbol{\tau}$. The above transformation of the elastic force from an Eulerian to Lagrangian description can be performed for general stress tensors. For the purposes of the present paper, this is not needed.

Substituting (7) and (12) in (5)

$$\rho\frac{\mathrm{D}\mathbf{u}}{\mathrm{D}t}+\nabla p=\mu\Delta\mathbf{u}+\mathbf{f}, \tag{13}$$

$$\mathbf{f}=\mathbf{F}|\mathbb{F}|^{-1}, \tag{14}$$

$$\mathbf{F}=\mathbf{F}_{\mathrm{E}}-M\frac{\partial^2\mathbf{X}}{\partial t^2}, \tag{15}$$

$$\mathbf{F}_{\mathrm{E}}=\frac{\partial}{\partial\theta}(T\boldsymbol{\tau}). \tag{16}$$

We have made use of (6) in deriving (15). In the above, it is to be understood that $\mathbf{f}$ is 0 in $U-\mathbf{X}(\Omega,t)$. Eqs. (13) and (2) are the momentum balance equations we would like to solve. Note that the inertial force due to the extra mass carried by the elastic material is incorporated in terms of a *d'Alembert force*, $-M\frac{\partial^2\mathbf{X}}{\partial t^2}$. This system is nothing other than the incompressible Navier–Stokes equations with an external force field $\mathbf{f}$.

We are now ready to write the equations of motion of a viscoelastic continuum in immersed boundary form. A salient feature of the immersed boundary method is the use of the Dirac delta function to link the Eulerian and Lagrangian descriptions. We write (6) and (14) respectively in the following way:

$$\frac{\partial\mathbf{X}}{\partial t}(\boldsymbol{\xi},t)=\int_U\mathbf{u}(\mathbf{x},t)\delta(\mathbf{x}-\mathbf{X}(\boldsymbol{\xi},t))\mathrm{d}\mathbf{x}, \tag{17}$$

$$\mathbf{f}(\mathbf{x},t)=\int_\Omega\mathbf{F}(\mathbf{X}(\cdot,\cdot),\boldsymbol{\xi},t)\delta(\mathbf{x}-\mathbf{X}(\boldsymbol{\xi},t))\mathrm{d}\boldsymbol{\xi}. \tag{18}$$

In (17) and (18), δ denotes the Dirac delta function. We shall call (17) and (18) the *interpolation* and *spreading* operations respectively. The equivalence of (17) and (6) follows directly from the definition of the Dirac delta function. The equivalence of (18) and (14) can be seen as follows. The spreading operation (18) should be understood in the following sense. Given any smooth vector valued function $\mathbf{w}(\mathbf{x})$,

$$\begin{aligned}\int_U\mathbf{f}(\mathbf{x},t)\cdot\mathbf{w}(\mathbf{x})\,\mathrm{d}\mathbf{x}&=\int_U\int_\Omega\mathbf{F}(\mathbf{X}(\cdot,t),\boldsymbol{\xi},t)\cdot\mathbf{w}(\mathbf{x})\delta(\mathbf{x}\\&\qquad-\mathbf{X}(\boldsymbol{\xi},t))\mathrm{d}\boldsymbol{\xi}\,\mathrm{d}\mathbf{x}\\&=\int_\Omega\mathbf{F}(\mathbf{X}(\cdot,t),\boldsymbol{\xi},t)\cdot\mathbf{w}(\mathbf{X}(\boldsymbol{\xi},t))\mathrm{d}\boldsymbol{\xi}.\end{aligned} \tag{19}$$

Note that $\mathbf{f}\cdot\mathbf{w}$ is integrated over U, and $\mathbf{F}\cdot\mathbf{w}$ is integrated over Ω. This, together with the arbitrariness of $\mathbf{w}$, shows that $\mathbf{f}$ and $\mathbf{F}$ are linked by multiplication by $|\mathbb{F}|$, the Jacobian of the coordinate transformation from Lagrangian to Eulerian coordinates. This is nothing other than Eq. (14).

In summary, the equations of motion in immersed boundary form are

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t}+\mathbf{u}\cdot\nabla\mathbf{u}\right)+\nabla p=\mu\Delta\mathbf{u}+\mathbf{f}, \tag{20}$$

$$\nabla\cdot\mathbf{u}=0, \tag{21}$$

$$\frac{\partial \mathbf{X}}{\partial t}(\xi,t)=\int_U \mathbf{u}(\mathbf{x},t)\delta(\mathbf{x}-\mathbf{X}(\xi,t))\mathrm{d}\mathbf{x}, \tag{22}$$

$$\mathbf{f}(\mathbf{x},t)=\int_\Omega \mathbf{F}(\mathbf{X}(\cdot,\cdot),\xi,t)\delta(\mathbf{x}-\mathbf{X}(\xi,t))\mathrm{d}\xi, \tag{23}$$

$$\mathbf{F}=\mathbf{F}_{\mathrm{E}}-M\frac{\partial^2\mathbf{X}}{\partial t^2}. \tag{24}$$

The above derivation assumed that the stress tensor does not experience any jumps across internal interfaces, and the model problem we consider in this paper satisfies this condition. The immersed boundary formulation is, however, more general than the above derivation suggests. Interface conditions can be accommodated in the immersed boundary method by suitable choice of the Lagrangian force density $\mathbf{F}$. In fact, the ease with which such interface conditions can be handled was one of the original motivations for the development of the method. We refer the reader to [26] for a systematic exposition of the incorporation of interface conditions within the immersed boundary framework. From here on, we shall deal exclusively with the immersed boundary formulation.

We have yet to specify the elastic stress, and for this purpose, we must specialize the above framework to a model problem. We deal with an immersed fiber continuum whose elastic force density is given by (16). What we have to specify are the fiber strength T and the fiber directions $\boldsymbol{\tau}$ with the associated material coordinate system.

We consider a problem in which a circular elastic structure is immersed in a periodic fluid domain. Formally, we let U be a periodic square domain of side l, and $\Omega=\{\xi=(\eta,\theta)|\eta\in[0,1],\ \theta\in\mathbb{R}/2\pi\mathbb{Z}\}$. We note that Ω is 2π periodic in the angular coordinate θ.

We took the fiber direction $\boldsymbol{\tau}$ to be the θ coordinate direction. For T we take

$$T=T\left(\xi,\left|\frac{\partial\mathbf{X}}{\partial\theta}\right|\right). \tag{25}$$

Thus, T is a position dependent function of $\left|\frac{\partial\mathbf{X}}{\partial\theta}\right|$.

We can also be obtain an elastic force density of the form (16) by taking the following elastic potential energy for the immersed elastic structure

$$\mathscr{E}[\mathbf{X}]=\int_\Omega E\left(\xi,\left|\frac{\partial\mathbf{X}}{\partial\theta}\right|\right)\mathrm{d}\xi, \tag{26}$$

where $E(\xi,s)$ is the elastic energy density function. That E is only a function of $\left|\frac{\partial\mathbf{X}}{\partial\theta}\right|$ expresses the fact that the immersed structure is a fiber continuum. By taking the functional derivative of the above with respect to $\mathbf{X}$, we obtain the following expression for T in terms of the potential energy density E:

$$T=\frac{\partial E}{\partial s}\left(\xi,\left|\frac{\partial\mathbf{X}}{\partial\theta}\right|\right). \tag{27}$$

Much of the ensuing discussion does not depend on the particular form of the elastic force. The fact that the elastic force depends only on $\left|\frac{\partial\mathbf{X}}{\partial\theta}\right|$ will play a role only in the formation of the preconditioner for Krylov subspace solvers, to be discussed later in Section 5.

3. Spatial discretization

Spatial discretization is mostly identical to the standard immersed boundary method as detailed in [19].

We first discuss discretization of the Navier–Stokes equation in the fluid domain U. For the velocity field $\mathbf{u}$, we employ a regular cartesian grid with grid spacing h. We introduce the following central, forward and backward difference operators. For a function $\phi(\mathbf{x})$ defined in U

$$(D^0_{h,\alpha}\phi)(\mathbf{x})=\frac{\phi(\mathbf{x}+h\mathbf{e}_\alpha)-\phi(\mathbf{x}-h\mathbf{e}_\alpha)}{2h}, \tag{28}$$

$$(D^+_{h,\alpha}\phi)(\mathbf{x})=\frac{\phi(\mathbf{x}+h\mathbf{e}_\alpha)-\phi(\mathbf{x})}{h}, \tag{29}$$

$$(D^-_{h,\alpha}\phi)(\mathbf{x})=\frac{\phi(\mathbf{x})-\phi(\mathbf{x}-h\mathbf{e}_\alpha)}{h}. \tag{30}$$

Here, $\mathbf{e}_\alpha$ are the unit coordinate vectors and $\alpha=x,y$. We shall use the notation $\mathbf{D}^0_h,\ \mathbf{D}^+_h,\ \mathbf{D}^-_h$ which are vectors of difference operators whose components are $D_{h,\alpha}$. We now discretize the differential operators using the above as follows:

$$\nabla p\rightarrow\mathbf{D}^0_h p, \tag{31}$$

$$\nabla\cdot\mathbf{u}\rightarrow\mathbf{D}^0_h\cdot\mathbf{u}, \tag{32}$$

$$\Delta u_i\rightarrow\mathbf{D}^+_h\cdot(\mathbf{D}^-_h u_i)\equiv L_h u_i, \tag{33}$$

where u_i denotes the two components of the velocity field $\mathbf{u}$. In discretizing the convection term $\mathbf{u}\cdot\nabla\mathbf{u}$, we first note that the following equality holds due to the incompressibility condition:

$$\mathbf{u}\cdot\nabla\mathbf{u}=\nabla\cdot(\mathbf{u}\otimes\mathbf{u}). \tag{34}$$

We take the average of the above expressions to obtain the following discretization of the convection term [31,19]:

$$\mathbf{u}\cdot\nabla u_i\rightarrow\frac{1}{2}\mathbf{u}\cdot\mathbf{D}^0_h u_i+\frac{1}{2}\mathbf{D}^0_h\cdot(\mathbf{u}u_i)\equiv Q_h(\mathbf{u})u_i. \tag{35}$$

The incompressible Navier–Stokes equations (20) and (21) are thus spatially discretized as follows:

$$\rho\left(\frac{\partial\mathbf{u}}{\partial t}+Q_h(\mathbf{u})\mathbf{u}\right)+\mathbf{D}^0_h p=\mu L_h\mathbf{u}+\mathbf{f}, \tag{36}$$

$$\mathbf{D}^0_h\cdot\mathbf{u}=0. \tag{37}$$

Here L_h and $Q_h(\mathbf{u})$ are applied component-wise to $\mathbf{u}$.

We note the following consequence of discretizing the convection term using (35). Introduce the following inner product on U:

$$(w, v)_U = \sum_{\mathbf{x}\in\mathscr{G}_U} w(\mathbf{x})v(\mathbf{x})h^2, \tag{38}$$

$$(\mathbf{w}, \mathbf{v})_U = (w_1, v_1)_U + (w_2, v_2)_U. \tag{39}$$

The above sum is over grid points of U. We have

$$\begin{aligned}(w, Q_h(\mathbf{u})v)_U &= \frac{1}{2}\left(w, u_i D^0_{h,i}v\right)_U + \frac{1}{2}\left(w, D^0_{h,i}(u_i v)\right)_U \\ &= \frac{1}{2}\left(u_i w, D^0_{h,i}v\right)_U - \frac{1}{2}\left(D^0_{h,i}w, u_i v\right)_U \\ &= -\frac{1}{2}\left(D^0_{h,i}(u_i w), v\right)_U - \frac{1}{2}\left(u_i D^0_{h,i}w, v\right)_U \\ &= -(Q_h(\mathbf{u})w, v)_U.\end{aligned} \tag{40}$$

In the above, we used the summation convention for repeated indices. In particular, this skew symmetry property of $Q_h(\mathbf{u})$ implies that

$$(\mathbf{u}, Q_h(\mathbf{u})\mathbf{u})_U = 0. \tag{41}$$

The skew symmetry property of $Q_h(\mathbf{u})$ ensures that the identity for the continuous velocity field $(\mathbf{u}, \mathbf{u}\nabla\mathbf{u}) = 0$ holds for the discretized velocity field. We shall use this property when we later discuss discrete energy conservation.

We deal with the incompressibility constraint based on the projection method [32,33]. It is well known that any smooth vector field may be decomposed into its divergence free and curl free components. Let P be the operator which maps a 2D vector field $\mathbf{w}$ to its divergence-free component

$$\mathbf{w} = P\mathbf{w} + \nabla\phi_w, \quad \nabla\cdot P\mathbf{w} = 0, \quad P\nabla\phi_w = 0. \tag{42}$$

Here ϕ_w is some scalar field. We introduce the discretization of P by a simple discretization of the above

$$\mathbf{w} = P_h\mathbf{w} + \mathbf{D}^0_h\phi_w, \quad \mathbf{D}^0_h\cdot P_h\mathbf{w} = 0, \quad P_h\mathbf{D}^0_h\phi_w = 0. \tag{43}$$

We may use the above P_h and (37) to eliminate p in (36). We obtain

$$\rho\left(\frac{\partial\mathbf{u}}{\partial t} + P_h(Q_h(\mathbf{u})\mathbf{u})\right) = \mu L_h\mathbf{u} + P_h\mathbf{f}. \tag{44}$$

We used $P_h\mathbf{u} = \mathbf{u}$ which follows from (37). We also used $P_hL_h = L_hP_h$. This can be seen by taking the discrete Fourier transform in $\mathbf{x}$, which is possible thanks to the periodic boundary conditions. Both P_h and L_h are reduced to multiplicative operations on a single frequency in discrete Fourier space, and from this, we can easily see that the two operators commute.

We now discuss the discretization of the spreading and interpolation operators. Spreading (23) and interpolation (22) operations involve integration against a delta function kernel. We approximate these integrals using discrete delta functions. The discrete delta functions we use are of the form

$$\delta_h(\mathbf{x}) = \frac{1}{h^2}d\left(\frac{x}{h}\right)d\left(\frac{y}{h}\right). \tag{45}$$

Following [19], we use the following function for d:

$$d(r) = \begin{cases}\frac{1}{8}(3 - 2|r| + \sqrt{1 + 4|r| - 4r^2}) & |r| \leqslant 1, \\ \frac{1}{8}(5 - 2|r| - \sqrt{-7 + 12|r| - 4r^2}) & 1 < |r| \leqslant 2, \\ 0 & 2 < |r|.\end{cases} \tag{46}$$

Using the above discrete delta function δ_h, we discretize spreading and interpolation as follows. We discretize Ω with a rectangular cartesian grid of grid spacing $\Delta\eta$ and $\Delta\theta$. For $\phi(\mathbf{x})$ and $\Phi(\xi)$ defined respectively on U and Ω

$$(S_h(\mathbf{X})\Phi)(\mathbf{x}) \equiv \sum_{\xi\in\mathscr{G}_\Omega}\Phi(\xi)\delta_h(\mathbf{x} - \mathbf{X}(\xi))\Delta\eta\Delta\theta, \tag{47}$$

$$(S^*_h(\mathbf{X})\phi)(\xi) \equiv \sum_{\mathbf{x}\in\mathscr{G}_U}\phi(\mathbf{x})\delta_h(\mathbf{x} - \mathbf{X}(\xi))h^2. \tag{48}$$

The summation above is over grid points in Ω in (47) and over grid points in U in (48). S_h and S^*_h are discretizations of the spreading operation (23) and the interpolation operation (22) respectively. Spreading S_h and interpolation S^*_h are transposes of each other in the following sense. Introduce the following inner product on the Lagrangian coordinate Ω:

$$(W, V)_\Omega = \sum_{\xi\in\mathscr{G}_\Omega} W(\mathbf{X})V(\mathbf{X})\Delta\eta\Delta\theta, \tag{49}$$

$$(\mathbf{W}, \mathbf{V})_\Omega = (W_1, V_1)_\Omega + (W_2, V_2)_\Omega. \tag{50}$$

The above sum is over grid points of Ω. We have

$$\begin{aligned}(w, S_hV)_U &= \sum_{\mathbf{x}\in\mathscr{G}_U} w(\mathbf{x})(S_hV)(\mathbf{x})h^2 \\ &= \sum_{\mathbf{x}\in\mathscr{G}_U} w(\mathbf{x})h^2\sum_{\xi\in\mathscr{G}_\Omega}V(\xi)\delta_h(\mathbf{x} - \mathbf{X}(\xi))\Delta\eta\Delta\theta \\ &= \sum_{\xi\in\mathscr{G}_\Omega}V(\xi)\Delta\eta\Delta\theta\sum_{\mathbf{x}\in\mathscr{G}_U}w(\mathbf{x})\delta_h(\mathbf{x} - \mathbf{X}(\xi))h^2 \\ &= (S^*_hw, V)_\Omega.\end{aligned} \tag{51}$$

This equality holds because we use of the same discrete delta function for both spreading and interpolation. We note that S_h and S^*_h are functions of $\mathbf{X}$.

The elastic force (16) is discretized in the following fashion:

$$\begin{aligned}\mathbf{F}_\mathrm{E}(\mathbf{X}) &= D^+_\theta\left(T(\xi, |D^-_\theta\mathbf{X}|)\frac{D^-_\theta\mathbf{X}}{|D^-_\theta\mathbf{X}|}\right) \\ &= D^-_\theta\left(T(\xi, |D^+_\theta\mathbf{X}|)\frac{D^+_\theta\mathbf{X}}{|D^+_\theta\mathbf{X}|}\right),\end{aligned} \tag{52}$$

where

$$(D^+_\theta\Phi)(\theta) = \frac{\Phi(\theta + \Delta\theta) - \Phi(\theta)}{\Delta\theta}, \quad (D^-_\theta\Phi)(\theta) = \frac{\Phi(\theta) - \Phi(\theta - \Delta\theta)}{\Delta\theta}. \tag{53}$$

These difference operators act component-wise to vector valued functions. The above discretization is equivalent to that obtained by discretizing the elastic energy functional as

$$\mathscr{E}[\mathbf{X}] = \sum_{\boldsymbol{\xi}\in\mathscr{G}_\Omega} E(\boldsymbol{\xi}, |D_\theta^+\mathbf{X}|)\Delta\theta\Delta\eta \tag{54}$$

and taking the gradient with respect to $\mathbf{X}$.

Putting all of this together, the equations discretized in space are

$$\rho\left(\frac{\partial\mathbf{u}}{\partial t} + P_h(Q_h(\mathbf{u})\mathbf{u})\right) = \mu L_h\mathbf{u} + P_h S_h \mathbf{F}, \tag{55}$$

$$\frac{\partial\mathbf{X}}{\partial t} = S_h^*\mathbf{u}, \tag{56}$$

$$\mathbf{F} = \mathbf{F}_\mathrm{E}(\mathbf{X}) - M\frac{\partial^2\mathbf{X}}{\partial t^2}. \tag{57}$$

S_h and S_h^* act component-wise on vector valued functions. To facilitate time discretization, we would like to replace the second derivative in (57) with something more tractable. We first try the following straightforward approach:

$$\frac{\partial^2\mathbf{X}}{\partial t^2} = \frac{\partial}{\partial t}(S_h^*\mathbf{u}) = S_h^*\frac{\partial\mathbf{u}}{\partial t} + \left(\frac{\partial S_h^*}{\partial t}\right)\mathbf{u}. \tag{58}$$

We used (56) to rewrite $\frac{\partial^2\mathbf{X}}{\partial t^2}$ in the first equality. We note that the operator S_h^* is a function of time through its dependence on $\mathbf{X}$, and thus we have a term involving the time derivative of S_h^*

$$\left(\frac{\partial S_h^*}{\partial t}\right)\mathbf{u} = \left(\frac{\partial S_h^*}{\partial\mathbf{X}}\mathbf{u}\right)\cdot\frac{\partial\mathbf{X}}{\partial t} = \left(\frac{\partial S_h^*}{\partial\mathbf{X}}\mathbf{u}\right)\cdot S_h^*\mathbf{u}. \tag{59}$$

The difficulty with the above expression is that this involves the derivatives of the discrete delta function δ_h in $\frac{\partial S_h^*}{\partial\mathbf{X}}$.

Let us return to the continuous system of equations before discretization. We take the derivative of (22) in t to find that the following holds:

$$\frac{\partial^2\mathbf{X}}{\partial t^2} = \frac{\partial\mathbf{u}}{\partial t}(\mathbf{X},t) + \mathbf{u}(\mathbf{X},t)\cdot\nabla\mathbf{u}(\mathbf{X},t). \tag{60}$$

The above states the familiar fact that the acceleration of a Lagrangian marker is equal to the material derivative of the velocity field. This suggests the use of the material derivative on the Eulerian grid interpolated at $\mathbf{X}$. Namely, in place of $\frac{\partial^2\mathbf{X}}{\partial t^2}$ we use

$$S_h^*\left(\frac{\partial\mathbf{u}}{\partial t} + (Q_h(\mathbf{u})\mathbf{u})\right). \tag{61}$$

This amounts to replacing $\left(\frac{\partial S_h^*}{\partial t}\right)\mathbf{u}$ with $S_h^*(Q_h(\mathbf{u})\mathbf{u})$ in (58).

The expression $\left(\frac{\partial S_h^*}{\partial t}\right)\mathbf{u}$ involves derivatives of the discrete delta function. This would require additional computation and might bring about loss of accuracy both in space and time. On the other hand, $S_h^*(Q_h(\mathbf{u})\mathbf{u})$ does not involve derivatives of the discrete delta function. For a smooth vector field $\mathbf{u}$, $S_h^*(Q_h(\mathbf{u})\mathbf{u})$ is a second-order accurate approximation to $\mathbf{u}(\mathbf{X},t)\cdot\nabla\mathbf{u}(\mathbf{X},t)$ since

$$\begin{aligned} S_h^*(\mathbf{X})(Q_h(\mathbf{u})\mathbf{u}) &= S_h^*(\mathbf{X})(\mathbf{u}(\mathbf{x},t)\cdot\nabla\mathbf{u}(\mathbf{x},t) + \mathscr{O}(h^2)) \\ &= \mathbf{u}(\mathbf{X},t)\cdot\nabla\mathbf{u}(\mathbf{X},t) + \mathscr{O}(h^2). \end{aligned} \tag{62}$$

The first equality follows since $Q_h(\mathbf{u})\mathbf{u}$ is a second-order accurate approximation to $\mathbf{u}\cdot\nabla\mathbf{u}$. The second equality follows if S_h^* gives a second-order accurate interpolant in h, which is true so long as S_h^* can interpolate linear functions exactly. This depends on the choice of discrete delta function δ_h, but is true for most δ_h used with the immersed boundary method including (46).

Thus, in place of (57), we shall use

$$\mathbf{F} = \mathbf{F}_\mathrm{E}(\mathbf{X}) - M\left(S_h^*\frac{\partial\mathbf{u}}{\partial t} + S_h^*(Q_h(\mathbf{u})\mathbf{u})\right). \tag{63}$$

All operators in the ensuing discussion are discretized in space, and therefore, we drop the subscripts h, $\boldsymbol{\xi}$, θ on the operators unless this dependence is relevant to the discussion.

4. Temporal discretization

4.1. A first-order method

Now we discretize (55), (56) and (63) in time. We first write down a first-order method. This method was inspired by [22], but there are some important differences. The implicit scheme in [22] is a first-order scheme that makes use of operator splitting, in which the projection P and the operation $\mathscr{L}$ are performed separately. We do not perform such operator splitting here. This makes it easier to extend the first-order method to develop a second-order scheme, as we will in the next subsection. We note also that [22] does not consider nonlinear forces or additional mass on the boundary. We advance from time n to time $n+1$ in the following way:

$$\frac{\mathbf{u}^{n+1}-\mathbf{u}^n}{\Delta t} = -P(Q(\mathbf{u}^n)\mathbf{u}^n) + \frac{\mu}{\rho}L\mathbf{u}^{n+1} + \frac{1}{\rho}PS_n\mathbf{F}^{n+1}, \tag{64}$$

$$\frac{\mathbf{X}^{n+1}-\mathbf{X}^n}{\Delta t} = S_n^*\mathbf{u}^{n+1}. \tag{65}$$

Here Δt is the time step and ^{n} and ${}^{n+1}$ denote values of quantities at time $n\Delta t$ and $(n+1)\Delta t$ respectively. $S_n \equiv S(\mathbf{X}^n)$ and $S_n^* \equiv S^*(\mathbf{X}^n)$ denote the spread and interpolation operators respectively evaluated at $\mathbf{X}^n$. We have not yet specified the form of the force $\mathbf{F}^{n+1}$. If we evaluate $\mathbf{F}$ at $\mathbf{X}^{n+1}$, we will be faced in general with a nonlinear equation in $\mathbf{X}^{n+1}$. We instead make use of the linear approximation $\mathbf{F}_\mathrm{E}(\mathbf{X}^{n+1}) \approx \mathbf{F}_\mathrm{E}(\mathbf{X}^n) + \frac{\partial\mathbf{F}_\mathrm{E}}{\partial\mathbf{X}}(\mathbf{X}^{n+1}-\mathbf{X}^n)$

$$\begin{aligned} \mathbf{F}^{n+1} = {} & \mathbf{F}_\mathrm{E}(\mathbf{X}^n) + J_n(\mathbf{X}^{n+1}-\mathbf{X}^n) \\ & - M\left(S_n^*\left(\frac{\mathbf{u}^{n+1}-\mathbf{u}^n}{\Delta t}\right) + S_n^*(Q(\mathbf{u}^n)\mathbf{u}^n)\right). \end{aligned} \tag{66}$$

Here J_n is the Jacobian matrix $\frac{\partial\mathbf{F}_\mathrm{E}}{\partial\mathbf{X}}$ evaluated at $\mathbf{X}^n$. When the elastic energy has the form (26), the action of J on a vector $\mathbf{Y}$ can be obtained by taking the gradient of expression (52) with respect to $\mathbf{X}$

$$J(\mathbf{X})\mathbf{Y} = J^0(\mathbf{X})\mathbf{Y} + J^1(\mathbf{X})\mathbf{Y}, \tag{67}$$

$$J^0(\mathbf{X})\mathbf{Y} = D_\theta^+\left(T(\boldsymbol{\xi}, |D_\theta^-\mathbf{X}|)\frac{D_\theta^-\mathbf{Y}}{|D_\theta^-\mathbf{X}|}\right), \tag{68}$$

$$J^1(\mathbf{X})\mathbf{Y} = D_\theta^+\left(G(\boldsymbol{\xi}, |D_\theta^-\mathbf{X}|)\left(\frac{D_\theta^-\mathbf{X}}{|D_\theta^-\mathbf{X}|}\cdot\frac{D_\theta^-\mathbf{Y}}{|D_\theta^-\mathbf{X}|}\right)\frac{D_\theta^-\mathbf{X}}{|D_\theta^-\mathbf{X}|}\right), \tag{69}$$

$$G(\boldsymbol{\xi}, s) = \frac{\partial T}{\partial s}(\boldsymbol{\xi}, s) - T(\boldsymbol{\xi}, s). \tag{70}$$

We note that $J^1 \equiv 0$ if and only if $G \equiv 0$. By solving the above differential equation for G, one sees that this is when the elastic energy density $E(\boldsymbol{\xi}, s)$ is proportional to s^2, in which case $\mathbf{F}_E$ is linear in $\mathbf{X}$. We may thus view J^1 as being the correction to the Jacobian when the elastic force $\mathbf{F}_E$ is not linear in $\mathbf{X}$.

The above scheme is *not* fully implicit. Spreading and interpolation are performed at $\mathbf{X}^n$, not at $\mathbf{X}^{n+1}$, and the term $Q(\mathbf{u})\mathbf{u}$ is evaluated at time n. As we have discussed above, we only use a linear approximation to the elastic force $\mathbf{F}_E$. These choices make the equations linear in $\mathbf{X}^{n+1}$ and $\mathbf{u}^{n+1}$.

We can use Eq. (65) to eliminate $S_n^*\mathbf{u}^{n+1}$ in (66). Substitute the resulting expression for $\mathbf{F}^{n+1}$ into (64), solve for $\mathbf{u}^{n+1}$ and substitute this result back into (65). We obtain the following equation:

$$(I + S_n^*\mathscr{L}PS_nA_n)(\mathbf{X}^{n+1} - \mathbf{X}^n) = \mathbf{Z}^n, \tag{71}$$

$$A_n = \frac{1}{\rho}(-(\Delta t)^2J_n + M), \quad \mathscr{L} = \left(I - \frac{\mu\Delta t}{\rho}L\right)^{-1}, \tag{72}$$

where I is the identity operator. The right hand side of (71), $\mathbf{Z}^n$, is an expression involving quantities known at time n as given below

$$\mathbf{Z}^n = S_n^*\mathscr{L}P\mathbf{z}\Delta t, \tag{73}$$

$$\mathbf{z} = \mathbf{q} + \frac{1}{\rho}S_n\left(\mathbf{F}_{\mathrm{E}}(\mathbf{X}^n)\Delta t + MS_n^*\mathbf{q}\right), \tag{74}$$

$$\mathbf{q} = \mathbf{u}^n - Q(\mathbf{u}^n)\mathbf{u}^n\Delta t. \tag{75}$$

Therefore the solution to the discretized system reduces to solving the linear equation (71). The solution to (71) can be substituted into (64)–(66) to find $\mathbf{u}^{n+1}$.

It is equally possible to eliminate $\mathbf{X}^{n+1}$ from these equations to obtain a linear equation in $\mathbf{u}^{n+1}$. The resulting linear system is

$$(I + \mathscr{L}PS_nA_nS_n^*)\mathbf{u}^{n+1} = \mathbf{w}^n, \tag{76}$$

where I is the identity operator acting on $\mathbf{u}$ and $\mathbf{w}^n$ is an expression involving known quantities. $\mathbf{X}^{n+1}$ may be found by substituting $\mathbf{u}^{n+1}$ into (65).

From the point of view of using Krylov subspace methods (see Section 5), the cost of the matrix–vector product is almost the same, whether we solve for $\mathbf{X}^{n+1}$ or for $\mathbf{u}^{n+1}$. We chose to solve for $\mathbf{X}^{n+1}$ because there seems to be no simple way to form an efficient preconditioner for (76).

4.2. A second-order semi-implicit method

Based on the above first-order method, we now derive a second-order semi-implicit method.

The basis of the time stepping is the Runge–Kutta method based on the Euler method for the first fractional time step and the midpoint and trapezoidal rules for the second fractional time step. This derivation parallels the route taken in deriving a second-order accurate explicit immersed boundary method in [31]. The equations before time discretization are Eqs. (55), (56) and (63). We first take a fractional time step from time n to $n+\frac{1}{2}$. This is identical to the first-order method except that the time step is $\Delta t/2$ instead of Δt

$$\frac{\mathbf{u}^{n+\frac{1}{2}} - \mathbf{u}^n}{\Delta t/2} = -P(Q(\mathbf{u}^n)\mathbf{u}^n) + \frac{\mu}{\rho}L\mathbf{u}^{n+\frac{1}{2}} + \frac{1}{\rho}PS_n\widetilde{\mathbf{F}}^{n+\frac{1}{2}}, \tag{77}$$

$$\frac{\mathbf{X}^{n+\frac{1}{2}} - \mathbf{X}^n}{\Delta t/2} = S_n^*\mathbf{u}^{n+\frac{1}{2}}, \tag{78}$$

$$\begin{aligned}\widetilde{\mathbf{F}}^{n+\frac{1}{2}} = {} & \mathbf{F}_{\mathrm{E}}(\mathbf{X}^n) + J_n(\mathbf{X}^{n+\frac{1}{2}} - \mathbf{X}^n)\\ & - MS_n^*\left(\frac{\mathbf{u}^{n+\frac{1}{2}} - \mathbf{u}^n}{\Delta t/2} + Q(\mathbf{u}^n)\mathbf{u}^n\right).\end{aligned} \tag{79}$$

Proceeding as in the first-order case, we obtain the following linear equation:

$$(I + S_n^*\mathscr{L}PS_nA_n)\left(\mathbf{X}^{n+\frac{1}{2}} - \mathbf{X}^n\right) = \mathbf{Z}_1^n, \tag{80}$$

$$A_n = \frac{1}{\rho}\left(-\frac{(\Delta t)^2}{4}J_n + M\right), \quad \mathscr{L} = \left(I - \frac{\mu\Delta t}{2\rho}L\right)^{-1}, \tag{81}$$

where $\mathbf{Z}_1^n$ can be expressed in terms of known quantities at time n

$$\mathbf{Z}_1^n = S_n^*\mathscr{L}P\mathbf{z}\frac{\Delta t}{2}, \tag{82}$$

$$\mathbf{z} = \mathbf{q} + \frac{1}{\rho}S_n\left(\mathbf{F}_{\mathrm{E}}(\mathbf{X}^n)\frac{\Delta t}{2} + MS_n^*\mathbf{q}\right), \tag{83}$$

$$\mathbf{q} = \mathbf{u}^n - Q(\mathbf{u}^n)\mathbf{u}^n\frac{\Delta t}{2}. \tag{84}$$

Once $\mathbf{X}^{n+\frac{1}{2}}$ is known, we may find $\mathbf{u}^{n+\frac{1}{2}}$ by

$$\mathbf{u}^{n+\frac{1}{2}} = \mathscr{L}P(\mathbf{z} - S_nA_n\left(\mathbf{X}^{n+\frac{1}{2}} - \mathbf{X}^n\right)/(\Delta t/2)). \tag{85}$$

We next come to the second fractional time step. This time step is based on the midpoint and the trapezoidal rules

$$\frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t} = -P(Q(\mathbf{u}^{n+\frac{1}{2}})\mathbf{u}^{n+\frac{1}{2}}) + \frac{\mu}{\rho}L\bar{\mathbf{u}} + \frac{1}{\rho}PS_{n+\frac{1}{2}}\mathbf{F}^{n+\frac{1}{2}}, \tag{86}$$

$$\frac{\mathbf{X}^{n+1} - \mathbf{X}^n}{\Delta t} = S_{n+\frac{1}{2}}^*\bar{\mathbf{u}}, \tag{87}$$

$$\begin{aligned}\mathbf{F}^{n+\frac{1}{2}} = {} & \mathbf{F}_{\mathrm{E}}(\mathbf{X}^{n+\frac{1}{2}}) + J_{n+\frac{1}{2}}\left(\bar{\mathbf{X}} - \mathbf{X}^{n+\frac{1}{2}}\right)\\ & - MS_{n+\frac{1}{2}}^*\left(\frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t} + Q(\mathbf{u}^{n+\frac{1}{2}})\mathbf{u}^{n+\frac{1}{2}}\right),\end{aligned} \tag{88}$$

$$\bar{\mathbf{u}} \equiv \frac{\mathbf{u}^n + \mathbf{u}^{n+1}}{2}, \quad \overline{\mathbf{X}} \equiv \frac{\mathbf{X}^n + \mathbf{X}^{n+1}}{2}. \tag{89}$$

Here $S_{n+\frac{1}{2}}$, $S_{n+\frac{1}{2}}^*$ and $J_{n+\frac{1}{2}}$ are the spread, interpolation and the Jacobian operators evaluated at $\mathbf{X}^{n+\frac{1}{2}}$.

After some algebra, we see that the solution to the above system can be reduced to solving the following linear equation:

$$\left(I+S^*_{n+\frac{1}{2}}\mathscr{L}PS_{n+\frac{1}{2}}A_{n+\frac{1}{2}}\right)(\mathbf{X}^{n+1}-2\mathbf{X}^{n+\frac{1}{2}}+\mathbf{X}^n)=\mathbf{Z}^n_2, \tag{90}$$

$$A_{n+\frac{1}{2}}=\frac{1}{\rho}\left(-\frac{(\Delta t)^2}{4}J_{n+\frac{1}{2}}+M\right),\quad \mathscr{L}=\left(I-\frac{\mu\Delta t}{2\rho}L\right)^{-1}, \tag{91}$$

where $\mathbf{Z}^n_2$ can be expressed in terms of known quantities at time n and time $n+\frac{1}{2}$

$$\mathbf{Z}^n_2=(-\mathbf{W}+S^*_{n+\frac{1}{2}}\mathscr{L}P\mathbf{z})\Delta t, \tag{92}$$

$$\mathbf{z}=\mathbf{q}+\frac{1}{\rho}S_{n+\frac{1}{2}}\mathbf{F}_{\mathrm{E}}\left(\mathbf{X}^{n+\frac{1}{2}}\right)\frac{\Delta t}{2}+\frac{1}{\rho}S_{n+\frac{1}{2}}M\left(-\mathbf{W}+S^*_{n+\frac{1}{2}}\mathbf{q}\right), \tag{93}$$

$$\mathbf{q}=\mathbf{u}^n-Q(\mathbf{u}^{n+\frac{1}{2}})\mathbf{u}^{n+\frac{1}{2}}\frac{\Delta t}{2},\quad \mathbf{W}=\frac{\mathbf{X}^{n+\frac{1}{2}}-\mathbf{X}^n}{\Delta t/2}. \tag{94}$$

The velocity field $\mathbf{u}^{n+1}$ may be obtained by

$$\mathbf{u}^{n+1}=-\mathbf{u}^n+2\mathscr{L}\left(\mathbf{z}-S_{n+\frac{1}{2}}A_{n+\frac{1}{2}}\left(\mathbf{X}^{n+1}-2\mathbf{X}^{n+\frac{1}{2}}+\mathbf{X}^n\right)/\Delta t\right). \tag{95}$$

4.3. A second-order fully implicit method

The scheme proposed here resembles the second step of the Runge–Kutta scheme described above, with the important difference that values needed at the midpoint of the time step are defined as averages of the initial and final values. Note that this is only possible in an implicit scheme, and indeed we obtain a fully implicit second-order scheme by adopting this strategy

$$\frac{\mathbf{u}^{n+1}-\mathbf{u}^n}{\Delta t}=-PQ(\bar{\mathbf{u}})\bar{\mathbf{u}}+\frac{\mu}{\rho}L\bar{\mathbf{u}}+\frac{1}{\rho}PS_{\overline{\mathbf{X}}}\mathbf{F}, \tag{96}$$

$$\frac{\mathbf{X}^{n+1}-\mathbf{X}^n}{\Delta t}=S^*_{\overline{\mathbf{X}}}\bar{\mathbf{u}}, \tag{97}$$

$$\mathbf{F}=\mathbf{F}_{\mathrm{E}}(\overline{\mathbf{X}})-MS^*_{\overline{\mathbf{X}}}\left(\frac{\mathbf{u}^{n+1}-\mathbf{u}^n}{\Delta t}+Q(\bar{\mathbf{u}})\bar{\mathbf{u}}\right), \tag{98}$$

$$\bar{\mathbf{u}}\equiv\frac{\mathbf{u}^n+\mathbf{u}^{n+1}}{2},\quad \overline{\mathbf{X}}\equiv\frac{\mathbf{X}^n+\mathbf{X}^{n+1}}{2}. \tag{99}$$

Here we perform the spreading and interpolation operations S^* and S at $\overline{\mathbf{X}}=\frac{\mathbf{X}^n+\mathbf{X}^{n+1}}{2}$.

In the special case of a quadratic elastic energy function, this fully implicit scheme satisfies an energy equality which we now derive. Take the inner product of Eq. (96) with $\rho\bar{\mathbf{u}}=\rho(\mathbf{u}^n+\mathbf{u}^{n+1})/2$

$$\rho\left(\bar{\mathbf{u}},\frac{\mathbf{u}^{n+1}-\mathbf{u}^n}{\Delta t}\right)_U+\rho(\bar{\mathbf{u}},PQ(\bar{\mathbf{u}})\bar{\mathbf{u}})_U$$
$$=\mu(\bar{\mathbf{u}},L\bar{\mathbf{u}})_U+(\bar{\mathbf{u}},PS_{\overline{\mathbf{X}}}\mathbf{F})_U. \tag{100}$$

Noting that P is an orthogonal projection in the U inner product (and thus $P^*=P$) and that $P\bar{\mathbf{u}}=\bar{\mathbf{u}}$ we obtain

$$\frac{\rho}{2}\frac{\|\mathbf{u}^{n+1}\|^2_U-\|\mathbf{u}^n\|^2_U}{\Delta t}+\rho(\bar{\mathbf{u}},Q(\bar{\mathbf{u}})\bar{\mathbf{u}})_U=\mu(\bar{\mathbf{u}},L\bar{\mathbf{u}})_U+(S^*_{\overline{\mathbf{X}}}\bar{\mathbf{u}},\mathbf{F})_\Omega. \tag{101}$$

In the last term, we used (51), the fact that S^* and S are transposes of each other. Finally, using the skew symmetry property of $Q(\mathbf{u})$ (40) and (97) we obtain

$$\frac{\rho}{2}\frac{\|\mathbf{u}^{n+1}\|^2_U-\|\mathbf{u}^n\|^2_U}{\Delta t}=\mu(\bar{\mathbf{u}},L\bar{\mathbf{u}})_U+\left(\frac{\mathbf{X}^{n+1}-\mathbf{X}^n}{\Delta t},\mathbf{F}\right)_\Omega. \tag{102}$$

This expression says that the change in total kinetic energy of the fluid computed on the Eulerian grid, is equal to the sum of dissipation due to viscosity computed on the Eulerian grid, and the work supplied by the immersed elastic structure computed on the Lagrangian grid. Suppose in addition, we place no additional mass on the immersed elastic structure (i.e., assume that the immersed elastic structure is neutrally buoyant in the ambient fluid). If the elastic energy density E has the form, $E(\xi,s)=Ks^2$, that is to say, when the elastic force is linear, we have the following discrete energy conservation relation:

$$\frac{\mathscr{E}^{n+1}_{\text{total}}-\mathscr{E}^n_{\text{total}}}{\Delta t}=\mu(\bar{\mathbf{u}},L\bar{\mathbf{u}})_U, \tag{103}$$

$$\mathscr{E}^k_{\text{total}}\equiv\frac{\rho}{2}\|\mathbf{u}^k\|^2_U+\mathscr{E}[\mathbf{X}^k]. \tag{104}$$

The change in total energy $\mathscr{E}_{\text{total}}$ is equal to dissipation due to viscosity. Since the discrete Laplacian L is negative semidefinite, it follows that the total energy is monotone decreasing and is therefore bounded. This in particular means that when the forces are linear and there is no additional mass on the boundary ($M=0$), $\mathbf{u}$ and $\mathbf{X}$ remain bounded in time. Boundedness of the numerical approximation does not necessarily imply stability of the scheme for a nonlinear evolution problem, since nonlinear stability pertains to the *difference* of solutions. However, this result does imply unconditional stability for a linearized simplification of this problem, in which the convective term $\mathbf{u}\cdot\nabla\mathbf{u}$ is dropped and interpolation and spreading are performed at fixed locations. We shall computationally explore the stability properties of both the fully and semi-implicit schemes in Section 6.

We note that the above property hinges upon two features of the fully implicit scheme:

- The convection term $Q(\mathbf{u})\mathbf{u}$ has a skew symmetry property and is evaluated at $\bar{\mathbf{u}}=(\mathbf{u}^n+\mathbf{u}^{n+1})/2$.
- The interpolation and spreading operators S^* and S are transposes of each other.

The difficulty with this fully implicit scheme is that the resulting equations are nonlinear. We solve (96)–(99) using the following iteration. Let the $\mathbf{u}^{n,k}$ and $\mathbf{X}^{n,k}$ be the kth guess to $\mathbf{u}^{n+1}$ and $\mathbf{X}^{n+1}$ respectively. We shall call this iteration, indexed by k, the outer iteration. We take $\mathbf{u}^{n,0}=\mathbf{u}^n$, $\mathbf{X}^{n,0}=\mathbf{X}^n$. We define $\mathbf{u}^{n,k}$ and $\mathbf{X}^{n,k}$ recursively as follows:

$$\frac{\mathbf{u}^{n,k+1}-\mathbf{u}^{n}}{\Delta t} = -PQ(\bar{\mathbf{u}}^{n,k})\bar{\mathbf{u}}^{n,k} + \frac{\mu}{\rho}L\bar{\mathbf{u}}^{n,k+1} + \frac{1}{\rho}PS_{\overline{\mathbf{X}}^{n,k}}\mathbf{F}, \tag{105}$$

$$\frac{\mathbf{X}^{n,k+1}-\mathbf{X}^{n}}{\Delta t} = S^{*}_{\overline{\mathbf{X}}^{n,k}}\bar{\mathbf{u}}^{n,k+1}, \tag{106}$$

$$\begin{aligned}\mathbf{F} = {} & \mathbf{F}_{\mathrm{E}}(\overline{\mathbf{X}}^{n,k}) + J(\overline{\mathbf{X}}^{n,k})(\overline{\mathbf{X}}^{n,k+1}-\overline{\mathbf{X}}^{n,k}), \\ & - MS^{*}_{\overline{\mathbf{X}}^{n,k}}\left(\frac{\mathbf{u}^{n,k+1}-\mathbf{u}^{n}}{\Delta t} + Q(\bar{\mathbf{u}}^{n,k})\bar{\mathbf{u}}^{n,k}v\right),\end{aligned} \tag{107}$$

$$\bar{\mathbf{u}}^{n,j} \equiv \frac{\mathbf{u}^{n}+\mathbf{u}^{n,j}}{2}, \quad \overline{\mathbf{X}}^{n,j} \equiv \frac{\mathbf{X}^{n}+\mathbf{X}^{n,j}}{2}. \tag{108}$$

The above equations are the same as those of the second substep of the semi-implicit scheme, as can be seen by replacing $\mathbf{u}^{n+\frac{1}{2}}$, $\mathbf{X}^{n+\frac{1}{2}}$, $\mathbf{u}^{n+1}$, $\mathbf{X}^{n+1}$, $\bar{\mathbf{u}}$, $\overline{\mathbf{X}}$ in (86)–(88) with $\bar{\mathbf{u}}^{n,k}$, $\overline{\mathbf{X}}^{n,k}$, $\mathbf{u}^{n,k+1}$, $\mathbf{X}^{n,k+1}$, $\bar{\mathbf{u}}^{n,k+1}$, $\overline{\mathbf{X}}^{n,k+1}$ respectively. We solve the following to obtain $\mathbf{X}^{n,k+1}$:

$$\left(I + S^{*}_{\overline{\mathbf{X}}^{n,k}}\mathscr{L}PS_{\overline{\mathbf{X}}^{n,k}}A_{n,k}\right)(\mathbf{X}^{n,k+1}-\mathbf{X}^{n,k}) = \mathbf{Z}^{n,k}, \tag{109}$$

$$A_{n,k} = \frac{1}{\rho}\left(-\frac{(\Delta t)^{2}}{4}J(\overline{\mathbf{X}}^{n,k}) + M\right), \quad \mathscr{L} = \left(I - \frac{\mu\Delta t}{2\rho}L\right)^{-1}, \tag{110}$$

where $\mathbf{Z}^{n,k}$ is expressed in terms of known quantities as follows:

$$\mathbf{Z}^{n,k} = (-\mathbf{W} + S^{*}_{\overline{\mathbf{X}}^{n,k}}\mathscr{L}P\mathbf{z})\Delta t, \tag{111}$$

$$\begin{aligned}\mathbf{z} = {} & \mathbf{q} + \frac{1}{\rho}S_{\overline{\mathbf{X}}^{n,k}}\mathbf{F}_{\mathrm{E}}(\overline{\mathbf{X}}^{n,k})\frac{\Delta t}{2} \\ & + \frac{1}{\rho}S_{\overline{\mathbf{X}}^{n,k}}M\left(-\mathbf{W} + S^{*}_{\overline{\mathbf{X}}^{n,k}}\mathbf{q}\right),\end{aligned} \tag{112}$$

$$\mathbf{q} = \mathbf{u}^{n} - Q(\bar{\mathbf{u}}^{n,k})\bar{\mathbf{u}}^{n,k}\frac{\Delta t}{2}, \quad \mathbf{W} = \frac{\mathbf{X}^{n,k}-\mathbf{X}^{n}}{\Delta t}. \tag{113}$$

The velocity field $\mathbf{u}^{n,k+1}$ may be obtained by

$$\mathbf{u}^{n,k+1} = -\mathbf{u}^{n,k} + 2\mathscr{L}P(\mathbf{z} - S_{n,k}A_{n,k}(\mathbf{X}^{n,k+1}-\mathbf{X}^{n,k})/\Delta t). \tag{114}$$

In practice, as we discuss in the next section, we will not solve the above linear equation exactly; we only perform a few iterations of a Krylov subspace solver (GMRES), which we shall call the inner iteration. We terminate the outer iteration when the right hand side of (109) is nearly equal to zero in magnitude

$$\|(I + S^{*}_{\overline{\mathbf{X}}^{n,k}}\mathscr{L}PS_{\overline{\mathbf{X}}^{n,k}}A_{n,k})(\mathbf{X}^{n,k+1}-\mathbf{X}^{n,k})\| = \|\mathbf{Z}^{n,k}\| \approx 0. \tag{115}$$

We shall discuss the convergence criterion in the next section. We let $\mathbf{u}^{n+1} = \mathbf{u}^{n,k}$, $\mathbf{X}^{n+1} = \mathbf{X}^{n,k}$, and move on to the next time step.

In the following, we shall mainly concern ourselves with the two second-order schemes. We do not intend to reach a conclusion as to which of the two schemes, the semi-implicit or fully implicit, is the better scheme. We believe that this choice is problem and goal dependent. We note that the ingredients that make up the two schemes are exactly the same. If one has coded one of the schemes, the other scheme can be coded with little further effort.

5. Solution of the linear equations

The linear equations to be solved have the following form, in either substep of the semi-implicit method or in one outer iteration of the fully implicit method.

$$(I + S^{*}\mathscr{L}PSA)\mathbf{X} = \mathbf{Z}. \tag{116}$$

It is much too expensive to work directly with the matrix entries of the operator $(I + S^{*}\mathscr{L}PSA)$. The only non-trivial operation that can be performed at reasonable cost with this operator is the matrix–vector product. We apply the operators $\mathscr{L}$ and P by making use of the Fast Fourier Transform, which is possible thanks to the periodic boundary conditions on the Eulerian fluid grid U. Krylov subspace methods, which only require the matrix–vector product, are therefore a natural choice to solve this linear system [34,35].

The operator $(I + S^{*}\mathscr{L}PSA)$ is not symmetric. Therefore, we cannot use the Conjugate Gradient method. We chose to use the Generalized Minimum Residual method (GMRES) for both the semi-implicit and fully implicit schemes [36].

For the semi-implicit method, we tested several Krylov subspace methods including the Biconjugate Gradient Stabilized (BiCGSTAB) [37] and the Conjugate Gradient Squared (CGS) methods [38]. We found that CGS fails to show convergence in many cases tested whereas BiCGSTAB and GMRES show convergence in all cases. GMRES was slightly more efficient than BiCGSTAB in most cases. This may be expected in the present computation, given that the bulk of the computation time is spent on the matrix–vector product. The large storage requirement of GMRES did not turn out to be serious for problems tested, partly because the iteration count was typically rather small, usually less than 15.

In the fully implicit method, we chose not to perform more than N_{in} inner iterations for each outer iteration. Thus, it is desirable that the residual be guaranteed to decrease for each inner iteration. BiCGSTAB or CGS are known not to exhibit monotonic convergence in general, whereas monotonic decrease in the residual is guaranteed for GMRES [35]. We thus chose to use GMRES and fixed $N_{\mathrm{in}} = 2$ after some numerical experimentation.

A good preconditioner is essential for the present methods to be efficient. Given that one step of GMRES is spent mostly on the matrix–vector product, one (inner) iteration is roughly equal in computational work to one explicit time step.

We use the following preconditioner $\mathscr{M}$, which is similar to the preconditioner used in [22]:

$$\mathscr{M} = I + \mathscr{D}A, \tag{117}$$

where $\mathscr{D}$ is a diagonal matrix whose diagonal entries $\mathscr{D}_{\xi\xi}$ are as follows:

$$\mathscr{D}_{\xi\xi} = (S^* S \mathbf{1}_\eta)(\xi), \quad \xi = (\eta, \theta), \tag{118}$$

$$\mathbf{1}_\eta(\eta', \theta') = \begin{cases} 1 & \eta' = \eta, \\ 0 & \eta' \neq \eta. \end{cases} \tag{119}$$

The vector $\mathbf{1}_\eta$ acts as the characteristic function of each fiber. In (118), this vector is spread and immediately interpolated back at the immersed boundary points. The value $\mathscr{D}_{\xi\xi}$, therefore, approximates the density of immersed boundary points along a given fiber. We may view $\mathscr{M}$ as approximating the operator $(I + S^* \mathscr{L} PSA)$ by disregarding all fluid dynamic interactions and retaining only elastic interactions.

The sparsity structure of $\mathscr{M}$ for our model problem is as follows. Suppose we lay a $N_\eta \times N_\theta$ grid over Ω such that $N_\eta \Delta\eta = 1$ and $N_\theta \Delta\theta = 2\pi$. The operator A acts on the two components of a vector valued function defined on the grid of Ω, and thus, is a $2N_\eta N_\theta \times 2N_\eta N_\theta$ square matrix. This matrix A is composed of N_η diagonal blocks of $2N_\theta \times 2N_\theta$ matrices. These $2N_\theta \times 2N_\theta$ submatrices are circulant block tridiagonal where each block is a 2×2 matrix. Since $\mathscr{M} = I + \mathscr{D}A$ where $\mathscr{D}$ is diagonal, $\mathscr{M}$ inherits the sparsity structure of A. This structure of $\mathscr{M}$ makes it easy to apply $\mathscr{M}^{-1}$.

The above gives us a general guide for forming a preconditioner. Given the elastic interaction matrix A, one can start with $I + \mathscr{D}A$, where $\mathscr{D}$ is some diagonal matrix specifying the local "density" of the immersed boundary points. In our model problem, the elastic interactions are simple enough so that $I + \mathscr{D}A$ is easy to invert. If the elastic interaction matrix is more complex, one may try some easily invertible approximation to $I + \mathscr{D}A$. In the present model problem, for example, we may use the following instead of $I + \mathscr{D}A$. Recall from (67) that $J = J^1 + J^0$. Instead of using the full operator J in A, we may replace J by J^0

$$\mathscr{M}^0 = I + \mathscr{D}\left(\frac{1}{\rho}\left(-\frac{(\Delta t)^2}{4} J^0 + M\right)\right). \tag{120}$$

The preconditioner $\mathscr{M}^0$ has the benefit of acting independently on the two components of a vector-valued function. Thus, $\mathscr{M}^0$ is effectively a $N_\eta N_\theta \times N_\eta N_\theta$ matrix, making it less expensive to form and apply than $\mathscr{M} = I + \mathscr{D}A$. When the elastic forces are close to linear, i.e. when J^1 is small, we have found that $\mathscr{M}^0$ is almost as effective as $\mathscr{M}$ in reducing the iteration count.

We next consider the convergence criterion for the GMRES iterations. Let $\mathscr{K} = I + S^* \mathscr{L} PSA$. Suppose we are stepping from time n to time $n + 1$, and are concerned with the equation

$$\mathscr{K}(\mathbf{X}^{n+1} - \mathbf{X}^n) = \mathbf{Z}^n, \tag{121}$$

where $\mathbf{Z}^n$ is the right-hand side. Let $\widetilde{\mathbf{X}}^{n+1} \approx \mathbf{X}^{n+1}$ be the approximate solution we obtain using GMRES. We take our initial guess $\widetilde{\mathbf{X}}_0^{n+1} = \mathbf{X}^n$. We would like $\widetilde{\mathbf{X}}^{n+1}$ to satisfy

$$\begin{aligned} \|\mathscr{M}^{-1}\mathscr{K}(\widetilde{\mathbf{X}}^{n+1} - \mathbf{X}^{n+1})\| &\leqslant \epsilon_{\mathrm{tol}} \|\mathscr{M}^{-1}\mathscr{K}(\mathbf{X}^{n+1} - \mathbf{X}^n)\| \\ &= \epsilon_{\mathrm{tol}} \|\mathscr{M}^{-1}\mathbf{Z}^n\|. \end{aligned} \tag{122}$$

Note that though $\mathbf{X}^{n+1}$ is a quantity to be solved for, the expression $\mathscr{K}(\mathbf{X}^{n+1} - \mathbf{X}^n) = \mathbf{Z}^n$ is a known quantity. In (122), we compare the residual $\mathscr{M}^{-1}\mathscr{K}(\widetilde{\mathbf{X}}^{n+1} - \mathbf{X}^{n+1})$ to $\mathscr{M}^{-1}\mathscr{K}(\mathbf{X}^{n+1} - \mathbf{X}^n)$ and not to $\mathscr{M}^{-1}\mathscr{K}\mathbf{X}^{n+1}$. Comparison with the latter will not yield a translation invariant criterion; the convergence criterion will depend on the position of the origin of the Eulerian coordinate system.

We now consider each linear equation in turn.

For Eq. (80), the first substep of the semi-implicit scheme, we take our initial guess $\widetilde{\mathbf{X}}_0^{n+\frac{1}{2}}$ to be $\mathbf{X}^n$. As the convergence criterion, we use (122) without modification

$$\begin{aligned} \|\mathscr{M}_n^{-1}\mathscr{K}_n(\widetilde{\mathbf{X}}^{n+\frac{1}{2}} - \mathbf{X}^{n+\frac{1}{2}})\| &\leqslant \epsilon_{\mathrm{tol}} \|\mathscr{M}_n^{-1}\mathscr{K}_n(\mathbf{X}^{n+\frac{1}{2}} - \mathbf{X}^n)\| \\ &= \epsilon_{\mathrm{tol}} \|\mathscr{M}_n^{-1}\mathbf{Z}_1^n\|, \end{aligned} \tag{123}$$

$$\mathscr{K}_n = I + S_n^* \mathscr{L} PS_n A_n. \tag{124}$$

Here $\widetilde{\mathbf{X}}^{n+\frac{1}{2}}$ is the GMRES approximation to $\mathbf{X}^{n+\frac{1}{2}}$ and $\mathscr{M}_n$ denotes the preconditioner corresponding to $\mathscr{K}_n$.

For Eq. (90), the second substep of the semi-implicit scheme, we take our initial guess $\widetilde{\mathbf{X}}_0^{n+1}$ to be $2\mathbf{X}^{n+\frac{1}{2}} - \mathbf{X}^n$, in line with the form of (90). We use the following modification of (122) as the convergence criterion:

$$\begin{aligned} \|\mathscr{M}_{n+\frac{1}{2}}^{-1}\mathscr{K}_{n+\frac{1}{2}}(\widetilde{\mathbf{X}}^{n+1} - \mathbf{X}^{n+1})\| &\leqslant 2\epsilon_{\mathrm{tol}} \|\mathscr{M}_n^{-1}\mathscr{K}_n(\mathbf{X}^{n+\frac{1}{2}} - \mathbf{X}^n)\| \\ &= 2\epsilon_{\mathrm{tol}} \|\mathscr{M}_n^{-1}\mathbf{Z}_1^n\|, \\ &\approx \epsilon_{\mathrm{tol}} \|\mathscr{M}_{n+\frac{1}{2}}^{-1}\mathscr{K}_{n+\frac{1}{2}}(\mathbf{X}^{n+1} - \mathbf{X}^n)\|, \end{aligned} \tag{125}$$

$$\mathscr{K}_{n+\frac{1}{2}} = \left(I + S_{n+\frac{1}{2}}^* \mathscr{L} PS_{n+\frac{1}{2}} A_{n+\frac{1}{2}}\right). \tag{126}$$

Here $\widetilde{\mathbf{X}}^{n+1}$ is the GMRES approximation to $\mathbf{X}^{n+1}$ and $\mathscr{M}_{n+\frac{1}{2}}$ denotes the preconditioner corresponding to $\mathscr{K}_{n+\frac{1}{2}}$. The reason we take this indirect convergence criterion is that $\mathscr{K}_{n+\frac{1}{2}}(\mathbf{X}^{n+1} - \mathbf{X}^n)$ is not a quantity that is computed within the scheme. Computing this quantity is expensive involving matrix vector multiplication with $\mathscr{K}_{n+\frac{1}{2}}$.

In solving (109) in the fully implicit scheme, as our initial guess $\widetilde{\mathbf{X}}_0^{n,k+1}$ we take $\mathbf{X}^{n,k}$. The GMRES iteration is terminated when the iteration count reaches $N_{\mathrm{in}}(=2)$ or when the following convergence criterion is met:

$$\begin{aligned} \|\mathscr{M}_{n,k}^{-1}\mathscr{K}_{n,k}(\widetilde{\mathbf{X}}^{n,k+1} - \mathbf{X}^{n,k+1})\| &\leqslant \epsilon_{\mathrm{tol}} \|\mathscr{M}_{n,0}^{-1}\mathscr{K}_{n,0}(\mathbf{X}^{n,1} - \mathbf{X}^n)\| \\ &= \epsilon_{\mathrm{tol}} \|\mathscr{M}_{n,0}^{-1}\mathbf{Z}^{n,0}\|, \\ &\approx \epsilon_{\mathrm{tol}} \|\mathscr{M}_{n,k}^{-1}\mathscr{K}_{n,k}(\mathbf{X}^{n,k+1} - \mathbf{X}^n)\|, \end{aligned} \tag{127}$$

$$\mathscr{K}_{n,k} = \left(I + S_{\overline{\mathbf{X}}_{n,k}}^* \mathscr{L} PS_{\overline{\mathbf{X}}_{n,k}} A_{n,k}\right). \tag{128}$$

Here $\widetilde{\mathbf{X}}^{n,k+1}$ is the GMRES approximation to $\mathbf{X}^{n,k+1}$ and $\mathscr{M}_{n,k}$ denotes the preconditioner corresponding to $\mathscr{K}_{n,k}$. The reason we take the above indirect criterion is the same as for the second substep of the semi-implicit scheme.

We terminate the outer iteration when the initial guess satisfies the above convergence criterion, i.e., the inner iteration count drops to zero

$$\begin{aligned}&\|\mathscr{M}_{n,k}^{-1}\mathscr{K}_{n,k}(\widetilde{\mathbf{X}}_0^{n,k+1}-\mathbf{X}^{n,k+1})\|\\&\quad=\|\mathscr{M}_{n,k}^{-1}\mathscr{K}_{n,k}(\mathbf{X}^{n,k}-\mathbf{X}^{n,k+1})\|=\|\mathscr{M}_{n,k}^{-1}\mathbf{Z}^{n,k}\|\\&\quad\leqslant\epsilon_{\text{tol}}\|\mathscr{M}_{n,0}^{-1}\mathscr{K}_{n,0}(\mathbf{X}^{n,1}-\mathbf{X}^n)\|=\epsilon_{\text{tol}}\|\mathscr{M}_{n,0}^{-1}\mathbf{Z}^{n,0}\|.\end{aligned}\qquad(129)$$

Thus, the termination criterion for the outer iteration is

$$\|\mathscr{M}_{n,k}^{-1}\mathbf{Z}^{n,k}\|\leqslant\epsilon_{\text{tol}}\|\mathscr{M}_{n,0}^{-1}\mathbf{Z}^{n,0}\|.\qquad(130)$$

When this criteria is met, we let $\mathbf{X}^{n+1}$ be $\mathbf{X}^{n,k}$ and $\mathbf{u}^{n+1}$ be $\mathbf{u}^{n,k}$.

The value of N_{in} is a parameter that we can adjust to attain optimal computational results. Taking $N_{\text{in}}=2$ seemed to produce good results. Increasing N_{in} led to a modest decrease in outer iteration count but this decrease was often not enough to justify this increase.

The value of the relative tolerance ϵ_{tol} will be taken small enough so that we observe second-order convergence, as we shall discuss in the next section.

6. Numerical results

6.1. Setup

For purposes of the numerical study, we further specialize the elastic energy density E to have the following form:\openup3

$$E(\boldsymbol{\xi},s)=\frac{1}{3}K_3(\eta)s^3+\frac{1}{2}K_2(\eta)s^2.\qquad(131)$$

We take the mass density of the elastic structure $M(\boldsymbol{\xi})$ to be a function only of η, and uniform in θ.

We shall use the following functional form for the stiffness constant and mass density for the immersed elastic structure. Following one of the computational examples in [27], the material properties are tapered toward the edges of the immersed structure:

$$K_i(\eta)=\kappa_i(1-\cos(2\pi\eta)),\qquad(132)$$

$$M(\eta)=\frac{M_0}{2\pi}(1-\cos(2\pi\eta)),\qquad(133)$$

where κ_i and M_0 are constants. We note that in this paper, we deal with additional mass on the elastic fibers, which was not handled in [27]. We have normalized the second equation so that M_0 is the total mass of the immersed elastic structure.

The initial configuration of the immersed elastic structure is the following:

$$\mathbf{X}(\eta,\theta,0)=\begin{pmatrix}l/2\\l/2\end{pmatrix}+\begin{pmatrix}l(\alpha+(\gamma-1/2)\eta)\cos(\theta)\\l(\beta+(\gamma-1/2)\eta)\sin(\theta)\end{pmatrix}.\qquad(134)$$

We recall that l is the length of the domain U. Here we take $\alpha=1/3,\ \beta=1/4,\ \gamma=1/12$. We take the initial velocity field $\mathbf{u}(\mathbf{x},0)$ to be 0 everywhere. Therefore, we start with a configuration in which a circular elastic shell, stretched in the x-coordinate direction, is immersed in a static fluid.

Dimensional analysis shows that the following dimensionless constants completely specify the problem:

$$\frac{M_0}{\rho l^2},\quad\frac{\rho\kappa_2 l^4}{\mu^2},\quad\frac{\kappa_3 l}{\kappa_2}.\qquad(135)$$

The first is the ratio between the mass of the immersed elastic structure and that of the fluid, and the second is an analogue of the Reynolds number where the representative velocity is taken to be $\kappa_2 l^3/\mu$. The second constant in (135) may also be considered a ratio between the relative importance of elastic and viscous effects. This is to be distinguished from the Reynolds number for a computational run, calculated with the computed velocity field. The last constant is the ratio between the cubic and quadratic terms in the energy density E. We shall take

$$\rho=1,\quad\kappa_2=1,\quad\kappa_3=1,\quad l=1\qquad(136)$$

and vary μ and M_0 in the numerical experiments. We are thus varying the first two dimensionless constants in (135) while keeping the last constant fixed at a value of 1. We take a Cartesian grid both on U and on Ω. For U, we use a uniform $N\times N$ grid, where the grid spacing is $h=l/N=1/N$. Following [27], we take the following grid for Ω:

$$(\eta_i,\theta_j)=\left(\frac{\Delta\eta}{2},\frac{\Delta\theta}{2}\right)+(i\Delta\eta,j\Delta\theta).\qquad(137)$$

We note that $N_\eta\Delta\eta=1$ and $N_\theta\Delta\theta=2\pi$. We utilize an offset of $\Delta\eta/2$ in order to avoid elastic fibers on the edges of the elastic shell.

We set $N_\eta=N/4$, $N_\theta=4N$ so that there are approximately 2–3 immersed boundary points per mesh width.

We performed our numerical study in Matlab, where computationally intensive components were coded in C.

6.2. Convergence

We use the following parameter values:

$$\mu=0.05,0.005,\quad M_0=0,1.\qquad(138)$$

When $M_0=1$, the total mass of the immersed elastic structure equals the total mass of fluid. We then have four situations in which we assess convergence. Let the number of time steps be N_{T} and the total time $T=N_{\text{T}}\Delta t$. When $M_0=0$ and $M_0=1$, we set $T=1$ and $T=1.5$ respectively. In both cases, the elastic shell completes approximately three quarters of an oscillatory cycle at time T (Fig. 2). We check for convergence at time T.

We analyze convergence in time and space separately. We note that this is only possible because we are using an implicit scheme. The stability of the method allows us to refine the mesh width h for fixed Δt, which would lead to numerical instability with an explicit scheme.

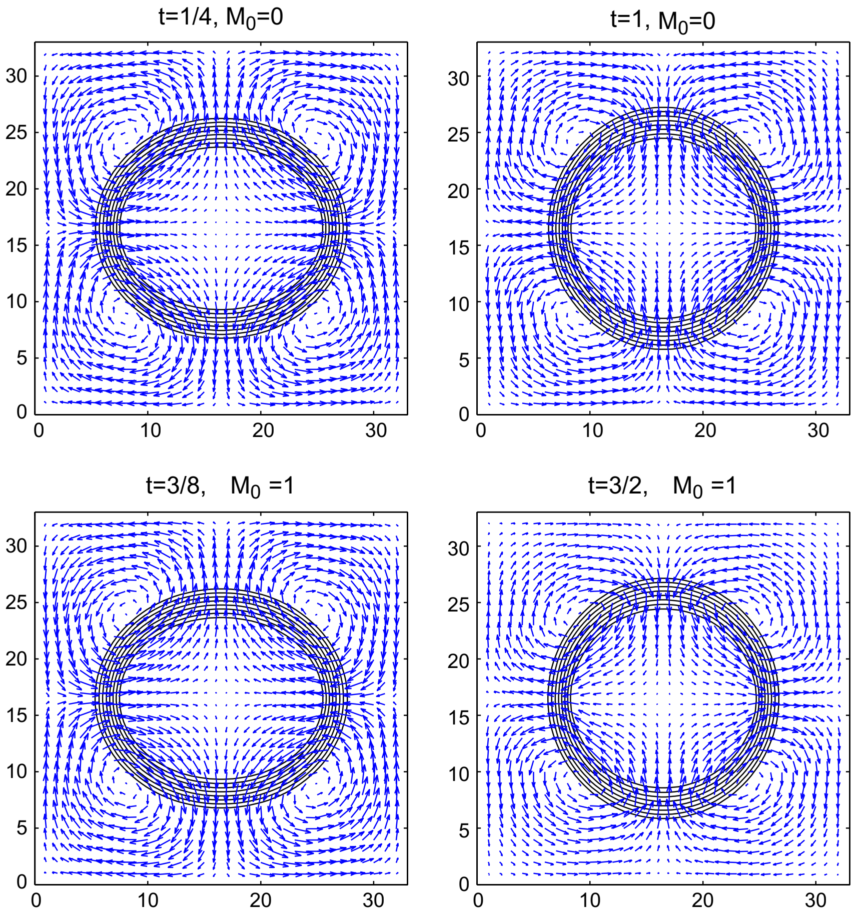

Fig. 2. Sample figure of simulation. The black lines indicate fiber positions. This figure was computed using the semi-implicit scheme with $\mu = 0.05$ and $N = 32$. When $M_0 = 0$, $T = 1$ is used and when $M_0 = 1$, $T = 1.5$ is used. When there is additional mass on the boundary, the oscillation is slower and hence the more time is required to complete three quarters of an oscillation.

6.2.1. Convergence in space

Spatial convergence is tested with $\Delta t = 1/16$. Thus, $N_{\mathrm{T}} = 16$ when $M_0 = 0$ and $N_{\mathrm{T}} = 24$ when $M_0 = 1$. We progressively refine the mesh in powers of 2 from $N = 2^6 = 64$ to $N = 2^9 = 512$. We use a relative tolerance $\epsilon_{\mathrm{tol}} = 1.0 \times 10^{-4}$ for the linear solver. We do *not* refine ϵ_{tol} as the spatial mesh is refined, unlike the temporal convergence study to be discussed later. We follow [27] in assessing spatial convergence.

For a vector field $\mathbf{w}(\mathbf{x}) = (w_1(\mathbf{x}), w_2(\mathbf{x}))$ defined on the cartesian coordinate U, we define the discrete L^p norm as follows:

$$\|\mathbf{w}\|_p = \left(\sum_{i,j}(w_1^2(x_i,y_j) + w_2^2(x_i,y_j))^{p/2}h^2\right)^{1/p}, \tag{139}$$

$$\|\mathbf{w}\|_\infty = \max_{i,j}(w_1^2(x_i,y_j) + w_2^2(x_i,y_j))^{1/2}, \tag{140}$$

where p in (139) satisfies $1 \leqslant p < \infty$. Likewise, we define the discrete L^p norm for a vector field $\mathbf{W} = (W_1, W_2)$ defined on the curvilinear coordinate Ω as follows:

$$\|\mathbf{W}\|_p = \left(\sum_{i,j}(W_1^2(\eta_i,\theta_j) + W_2^2(\eta_i,\theta_j))^{p/2}\Delta\eta\Delta\theta\right)^{1/p}, \tag{141}$$

$$\|\mathbf{W}\|_\infty = \max_{i,j}(W_1^2(\eta_i,\theta_j) + W_2^2(\eta_i,\theta_j))^{1/2}, \tag{142}$$

where p in (141) satisfies $1 \leqslant p < \infty$. For a computed quantity q, let the quantity computed with an $N \times N$ mesh or the corresponding curvilinear be denoted by q^N. We define a measure of error $e_p^s[q;N]$ (where the superscript s denotes *spatial* convergence) as follows:

$$e_p^s[q;N] = \|q^N - \mathscr{I}^{2N\to N}q^{2N}\|_p. \tag{143}$$

Here $\mathscr{I}^{2N\to N}$ is an interpolation operator from the finer to the coarser grid. As an empirical measure of convergence rate in space, we use

$$r_p^s[q;N] = \log_2\left(\frac{e_p^s[q;N]}{e_p^s[q;2N]}\right). \tag{144}$$

Empirical convergence rates are given in Table 1 as well as in Fig. 3. We see approximate second-order convergence for both $\mathbf{u}$ and $\mathbf{X}$.

We note here that we did not need to refine ϵ_{tol} to obtain second-order convergence, which is in contrast to the temporal convergence study to be presented below. This is presumably because the spatially discrete problem with $\epsilon_{\mathrm{tol}} = 1 \times 10^{-4}$ is approaching the spatially continuous

Table 1
Empirical convergence rates in space ($N = 128$) are shown for $\mathbf{u}$ and $\mathbf{X}$ in the L^1, L^2 and L^∞ norms

M_0	μ	Re	q	Semi-implicit			Fully implicit		
				$r_1^s[q,N]$	$r_2^s[q,N]$	$r_\infty^s[q,N]$	$r_1^s[q,N]$	$r_2^s[q,N]$	$r_\infty^s[q,N]$
0	μ_1	3.5	$\mathbf{u}$	1.97	1.97	1.98	1.97	1.97	1.98
			$\mathbf{X}$	1.90	1.87	1.80	1.90	1.87	1.80
	μ_2	65	$\mathbf{u}$	2.13	2.18	2.23	2.14	2.20	2.24
			$\mathbf{X}$	1.88	1.81	1.64	1.86	1.81	1.65
1	μ_1	30	$\mathbf{u}$	1.91	1.95	1.93	2.11	2.08	2.03
			$\mathbf{X}$	1.84	1.83	1.70	1.90	1.86	1.75
	μ_2	420	$\mathbf{u}$	1.84	1.80	1.47	1.86	1.87	1.64
			$\mathbf{X}$	1.36	1.45	1.45	1.85	1.76	1.62

We take $T = 1$, $N_T = 16$ when $M_0 = 0$ and $T = 1.5$, $N_T = 24$ when $M_0 = 1$. In the table, $\mu_1 = 0.05$ and $\mu_2 = 0.005$. Re denotes the approximate Reynolds number for each of the four cases, calculated using (145). Convergence rate is calculated at $t = T$. si: semi-implicit, fi: fully implicit.

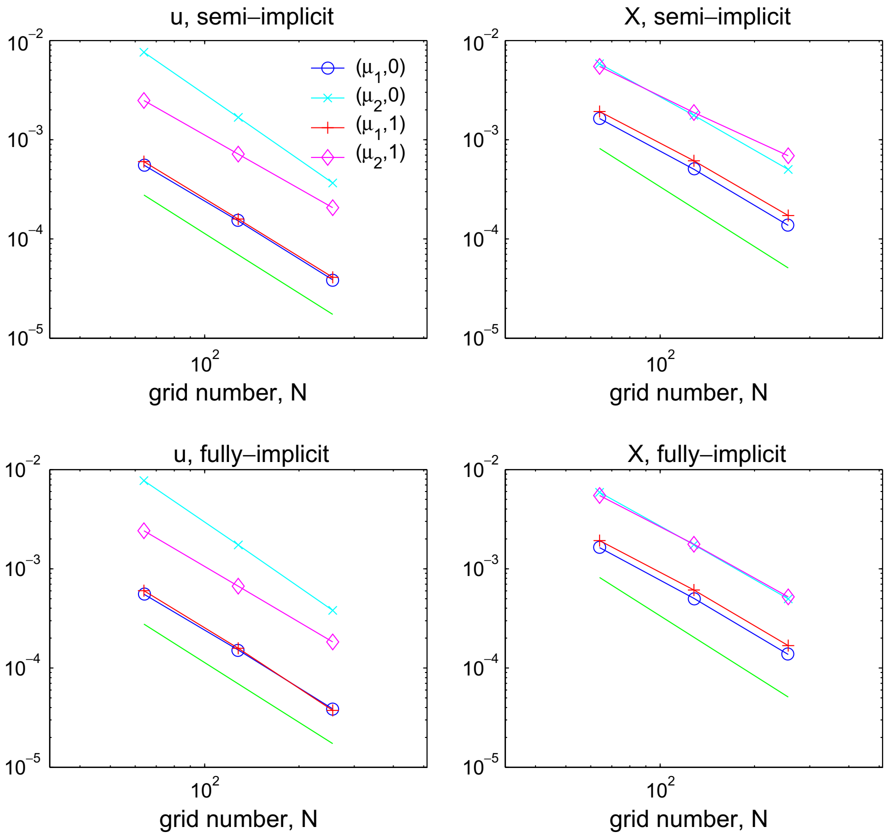

Fig. 3. Plot of the L^2 errors in space, $e_2^s[q,N]$, for the semi-implicit and fully implicit schemes. The legend denotes (μ, M_0) where $\mu_1 = 0.05$, $\mu_2 = 0.005$. The line at the bottom of each graph is a reference line to check for second-order accuracy.

(but time discrete) problem with the same tolerance. Each and every iterate of the solution procedure is approaching its continuous counterpart. A consequence of this is that the iteration count should be constant regardless of the spatial mesh if the mesh is fine enough, which is exactly what we find in Section 6.4.

The velocity field $\mathbf{u}$ seems to attain second-order convergence over all parameter ranges. The immersed structure configuration $\mathbf{X}$ attains an order of accuracy of about 1.8–1.9 on average, and does not reach the theoretical order of 2. The reason for this is not clear. The deterioration of the convergence rate for $(\mu, M_0) = (0.005, 1)$, especially for the semi-implicit scheme, is most likely due to the high Reynolds number of this computation. We compute the Reynolds number for each computational run as follows:

$$Re = \max_{t=k\Delta t} \|(\rho + S_h M)\mathbf{u}(\mathbf{x},t)\|_\infty \frac{l}{\mu}. \tag{145}$$

The inclusion of the term $S_h M$ in the above accounts for the presence of additional mass on the immersed elastic boundaries. As is listed in Table 1, the Reynolds number

$Re = 420$ when $(\mu, M_0) = (0.005, 1)$ is relatively high compared with the other three computational runs for which we observe approximate second-order convergence.

6.2.2. Convergence in time

To assess convergence in time, we set $N = 128$ and vary N_{T} in powers of 2 from 2^3 to 2^6. We use the same parameters for μ and M_0. We take the relative tolerance for the linear solver to be

$$\epsilon_{\mathrm{tol}} = \epsilon_0 \times 4^{2-k} \quad \text{for } \Delta t = 2^{-k}, \tag{146}$$

where ϵ_0 is the tolerance used for $\Delta t = 1/16$. We use the above tolerance for the following reason. There are two sources of error at each time step: the error due to discretization of the partial differential equations and the error in the solution of the linear equations. In order to observe second-order convergence in time, we must make sure that the error incurred in the solution of the linear equations decreases at least at the same rate as the discretization error. Since we expect second-order convergence, the discretization error at each time step should scale as $(\Delta t)^{2+1}$. In order for (123), (125), (127) to scale as $(\Delta t)^{2+1}$, we need ϵ_{tol} to scale like $(\Delta t)^2$.

The error of the method was analyzed similarly to spatial convergence. The error at time t is computed as

$$e^t_p[q; N_T] = \|q^{N_T}(t) - q^{2N_T}(t)\|_p. \tag{147}$$

We calculate the convergence rates at $t = T$. As an empirical measure of convergence rate, we use

$$r^t_p[q; N_T] = \log_2\left(\frac{e^t_p[q; N_T]}{e^t_p[q; 2N_T]}\right). \tag{148}$$

We took $\epsilon_0 = 1 \times 10^{-4}$. Empirical convergence rates are given in Table 2 as well as in Fig. 4. We see approximate second-order convergence in all parameters ranges considered here.

6.2.3. Convergence in space and time

To assess convergence in space and time, we set $h = 2^{-3-k}(N = 2^{k+3})$, $\Delta t = 2^{-k}$ for $k = 0, 1, 2, 3$. Both Δt and h are taken to approach 0 so that the ratio $\Delta t/h$ is held constant. We use the same parameters for μ and M_0. Similarly to the temporal convergence study, we take the relative tolerance for the linear solver to be

$$\epsilon_{\mathrm{tol}} = \epsilon_0 \times 4^{2-k} \quad \text{for } \Delta t = 2^{-k}, \tag{149}$$

where ϵ_0 is the tolerance used for $\Delta t = 1/16$. We took $\epsilon_0 = 1 \times 10^{-4}$.

The error of the method was analyzed similarly to spatial convergence. We define the space time error e^{st} and its rate r^{st} similarly to (143) and (144). Empirical convergence rates are given in Table 3 as well as in Fig. 5. We see approximate second-order convergence in all parameters ranges considered here.

6.3. Stability

We study the stability properties of the implicit methods and compare this with those of an explicit method. We shall use the second-order explicit method detailed in [19], which is a slight modification of the scheme introduced in [31].

Stability is a concept that applies to a numerical scheme and not to a single computational run. However, for purposes of investigating stability computationally, it is convenient to have some definition of stability that applies to each computational run. We take the following simple definition. For the semi-implicit and explicit schemes, we shall call a computation stable if the immersed boundary points $\mathbf{X}$ always lie within the computational domain at the end of a substep or a time step. For the fully-implicit scheme, we shall call a computation stable if the iterative scheme converges at each time step and the resulting immersed boundary positions lie within the computational domain. Otherwise, we shall call the computation unstable. These operational definitions work because we know from experience that when instability occurs it is so violent that the immersed boundary points leave the computational domain within a few time steps.

We let

$$\begin{aligned}(\mu, M_0) = {} & (0.05, 0), (0.005, 0), (0.0005, 0) \\ & \times (0.05, 1), (0.005, 1), (0.0005, 1)\end{aligned} \tag{150}$$

Table 2
Empirical convergence rates in time are shown for $\mathbf{u}$ and $\mathbf{X}$ in the L^1, L^2 and L^∞ norms

M_0	μ	q	Semi-implicit			Fully implicit		
			$r^t_1[q, N_T]$	$r^t_2[q; N_T]$	$r^t_\infty[q; N_T]$	$r^t_1[q; N_T]$	$r^t_2[q; N_T]$	$r^t_\infty[q; N_T]$
0	μ_1	$\mathbf{u}$	1.99	2.00	2.32	2.03	2.05	2.33
		$\mathbf{X}$	2.03	2.03	2.02	2.04	2.04	2.05
	μ_2	$\mathbf{u}$	2.14	2.31	2.17	2.26	2.39	2.26
		$\mathbf{X}$	1.97	1.94	1.75	1.89	1.84	1.72
1	μ_1	$\mathbf{u}$	1.92	1.93	1.84	1.94	1.95	1.95
		$\mathbf{X}$	1.96	1.90	1.68	2.03	2.03	2.07
	μ_2	$\mathbf{u}$	1.98	1.99	1.93	2.00	2.03	2.09
		$\mathbf{X}$	1.91	1.89	1.68	2.03	2.01	1.96

We take $T = 1$, $\Delta t = 1/16(N_{\mathrm{T}} = 16)$ when $M_0 = 0$ and $T = 1.5$, $\Delta t = 1/16(N_{\mathrm{T}} = 24)$ when $M_0 = 1$. The mesh was taken to be $N = 128$. In the table, $\mu_1 = 0.05$ and $\mu_2 = 0.005$. Convergence rate is calculated at $t = T$. si: semi-implicit, fi: fully implicit.

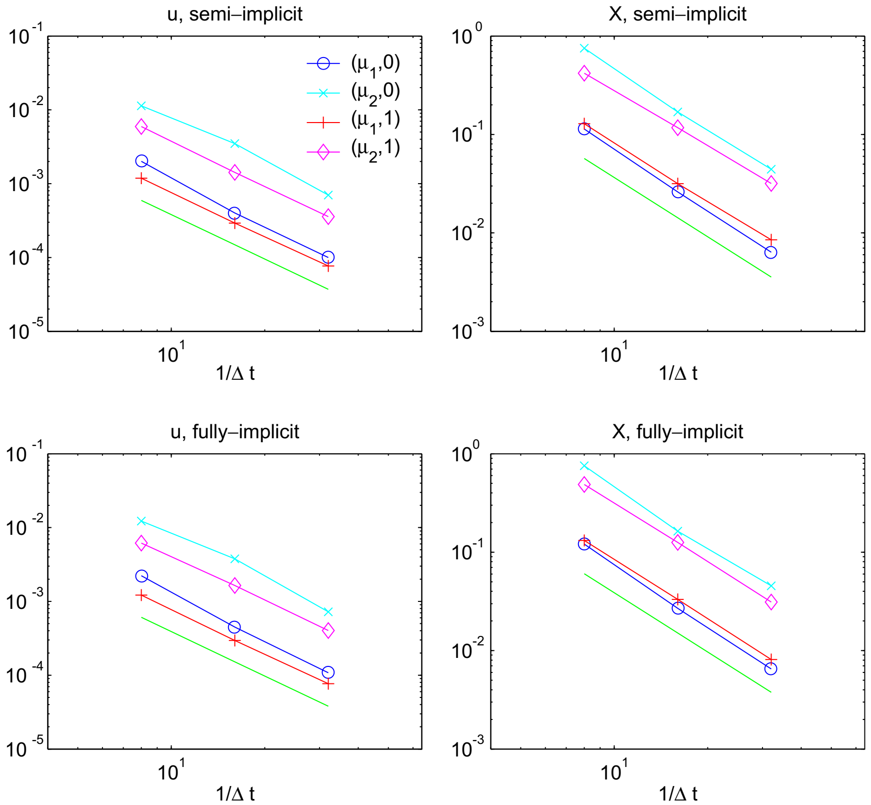

Fig. 4. Plot of the L^2 errors in time, $e_2^t[q, N_T]$, for the semi-implicit and fully implicit schemes. The legend denotes (μ, M_0) where $\mu_1 = 0.05$, $\mu_2 = 0.005$. The line at the bottom of each graph is a reference line to check for second-order accuracy.

Table 3
Empirical convergence rates in space and time are shown for **u** and **X** in the L^1, L^2 and L^∞ norms

M_0	μ	q	Semi-implicit			Fully implicit		
			$r_1^{st}[q,N]$	$r_2^{st}[q,N]$	$r_\infty^{st}[q;N]$	$r_1^{st}[q;N]$	$r_2^{st}[q;N]$	$r_\infty^{st}[q;N]$
0	μ_1	**u**	1.98	2.00	2.12	2.03	2.04	2.14
		X	1.95	1.92	1.81	1.94	1.92	1.82
	μ_2	**u**	2.11	2.34	2.21	2.20	2.38	2.14
		X	1.97	1.95	1.82	1.91	1.91	1.80
1	μ_1	**u**	1.95	1.96	1.99	1.85	1.87	1.94
		X	1.93	1.90	1.76	1.92	1.90	1.79
	μ_2	**u**	2.04	2.05	2.02	1.96	1.98	1.89
		X	1.85	1.81	1.80	1.89	1.83	1.80

We take $T = 1, N_T = 16$ when $M_0 = 0$ and $T = 1.5$, $N_T = 24$ when $M_0 = 1$. The mesh was taken to be $N = 128$. In the table, $\mu_1 = 0.05$ and $\mu_2 = 0.005$. Convergence rate is calculated at $t = T$. si: semi-implicit, fi: fully implicit.

and consider a total of six cases. We run the simulation for a total time of $T = 1$ for each of the above cases, during which time the immersed elastic structure goes through 1/2–3/4 of an oscillatory cycle. Experience indicates that if the computation is stable up to this point, it is stable for subsequent oscillatory cycles. This is presumably because it is in the first oscillatory cycle that the velocity of the immersed boundary points is the greatest. The Reynolds number for each of the six cases, computed as using (145), with $N = 128$ was approximately 3.5, 65, 850, 30, 420, 4500 respectively. We thus examine stability over a wide range of Reynolds numbers. We vary the spatial mesh in the following fashion:

$$N = 64, 128, 256, 512. \tag{151}$$

As the mesh size N is increased, we expect that the number of time steps N_T must be increased in order to avoid unstable computations. For each N, we increase the number of time steps from $N_T = 4$ in powers of 2, until we obtain stable computations in the sense described above. We note that $N_T = 4$ is a very large time step; the immersed elastic

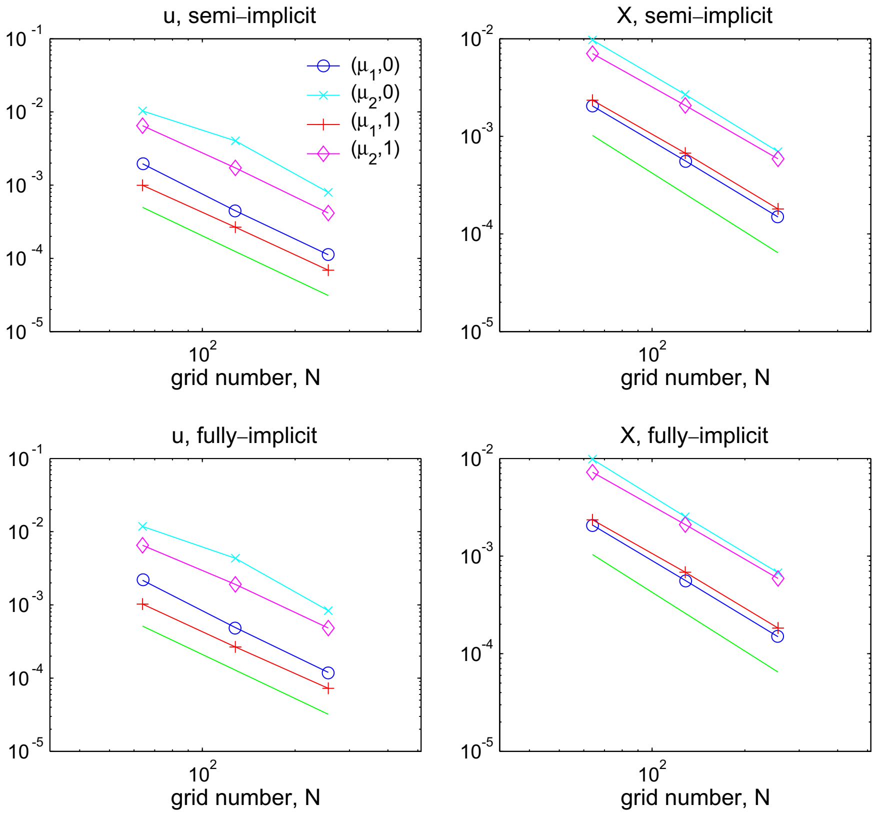

Fig. 5. Plot of the L^2 errors in space and time, $e_2^{st}[q,N]$, for the semi-implicit and fully implicit schemes. The legend denotes (μ, M_0) where $\mu_1 = 0.05$, $\mu_2 = 0.005$. The line at the bottom of each graph is a reference line to check for second-order accuracy.

Table 4
The smallest value of the number of time steps $N_T(\geqslant 4)$, in powers of 2, such that stable computations are possible

μ		0.05			0.005			0.0005		
M_0	N	ex	si	fi	ex	si	fi	ex	si	fi
0	2^6	2^6	4	4	2^7	4	4	2^7	4	4
	2^7	2^7	4	4	2^8	4	4	2^9	4	8
	2^8	2^8	4	4	2^9	4	4	2^{10}	4	16
	2^9	2^9	4	4	2^{10}	4	4	2^{11}	4	16
1	2^6	–	4	4	–	4	4	–	4	8
	2^7	–	4	4	–	4	4	–	4	16
	2^8	–	4	4	–	4	8	–	4	64
	2^9	–	4	4	–	4	16	–	4	64

When $M_0 = 1$, explicit computations were not performed for reasons explained in text. ex: explicit, si: semi-implicit, fi: fully implicit.

structure must go through more than half an oscillatory cycle in four time steps.

In Table 4, we list for each N the smallest value of N_T at which we obtain stable computations. For the semi-implicit method, computations could be performed stably at $N_T = 4$ for all 6 combinations of μ and M_0. For the fully-implicit method, computations were stable at $N_T = 4$ when $(\mu, M_0) = (0.05, 0), (0.005, 0), (0.05, 1)$, but not so for the other cases. We also list these values for the explicit scheme when $M_0 = 0$. It is difficult to make a comparison between the implicit and explicit calculations when $M_0 \neq 0$, since it is not possible to incorporate fiber mass in the explicit scheme in the same manner as is done here for the implicit methods.

We see that the stability restrictions of the implicit methods are far less severe than those of the explicit method. The explicit scheme requires finer time steps as the spatial mesh is refined, whereas this does not seem to be necessary for the semi-implicit scheme. The fully implicit scheme does require smaller time steps as N is increased when μ is small or M_0 is large, but the number of time steps required is nevertheless far smaller than is required for the explicit scheme.

The semi-implicit method seems to possess better stability properties than the fully-implicit method. One reason for this may be that the fully-implicit method requires solution of a nonlinear algebraic equation. In unstable computations, the residual for each iteration of the nonlinear algebraic solver either stagnated indefinitely or took an upward turn after a considerable reduction from its value at the beginning of the iteration. The semi-implicit scheme is not plagued by this problem, since the algebraic equations are linear and the GMRES iteration is guaranteed to terminate. Unstable computations may thus be a result of the poor performance of the iterative algorithm to solve the nonlinear algebraic equations and not of the nature of the discretization.

A theoretical analysis of the above stability properties is desirable but probably difficult. The most detailed theoretical analysis to date of stability of the immersed boundary method is probably that of Stockie and Wetton [20]. Their study, though insightful, only applies in the linear regime.

6.4. Performance

We saw in the above that the implicit methods have superior stability properties compared with the explicit method. However, these methods would be of limited use if the additional computational cost of having to solve algebraic equations outweighed the computational advantage of being able to take larger time steps.

We compare the performance of the implicit methods with the second-order explicit method used above in the stability study. We use two measures, the number of $\mathscr{L}P$ operations $N_{\mathscr{L}P}$ and execution time. The $\mathscr{L}P$ operation is asymptotically the most expensive of all operations in one computational cycle, for both the explicit and implicit methods. The operation $\mathscr{L}P$ requires 4 two-dimensional Fast Fourier Transforms whose asymptotic cost is $\mathcal{O}(N^2\log(N))$, and $\mathcal{O}(N^2)$ operations in Fourier space. The number of spreading and interpolation operations is also equal to $N_{\mathscr{L}P}$. This gives us an implementation independent comparison of the explicit and implicit schemes.

We use the model problem $(\mu, M_0) = (0.05, 0), (0.005, 0)$. We restrict our comparison to $M_0 = 0$, because it is difficult to perform a comparative study with the explicit scheme when $M_0 \neq 0$ as remarked earlier.

In order to make a fair comparison between the explicit and implicit schemes, we must run the implicit schemes with time steps small enough so that the computed solution is a reasonable approximation to the true solution. We take $N_{\mathrm{T}} = 8{,}16$ and $N = 64, 128, 256, 512$ for the implicit computations. In this range of (N, N_T), we know from the convergence study that the computed solution is approaching the true solution at approximately the asymptotic rate.

Table 5
The number of $\mathscr{L}P$ operations, $N_{\mathscr{L}P}$, performed for each computation

	$\mu = 0.05$					$\mu = 0.005$				
N	exp.	$N_{\mathrm{T}} = 8$		$N_{\mathrm{T}} = 16$		exp.	$N_{\mathrm{T}} = 8$		$N_{\mathrm{T}} = 16$	
		si	fi	si	fi		si	fi	si	fi
64	128	70	81	122	144	256	108	117	183	198
128	256	70	83	117	146	512	117	125	193	213
256	512	70	83	117	146	1024	118	128	199	216
512	1024	70	84	116	146	2048	118	128	202	218

ex: explicit, si: semi-implicit, fi: fully implicit.

The tolerance of GMRES was taken to be $\epsilon_0 = 1 \times 10^{-4}$. Therefore, we use $\epsilon_{\mathrm{tol}} = 4 \times 10^{-4}$ for $N_{\mathrm{T}} = 8$ and $\epsilon_{\mathrm{tol}} = 1 \times 10^{-4}$ for $N_{\mathrm{T}} = 16$ in accordance with (146).

In Table 5 is listed the number of $\mathscr{L}P$ operations. In the explicit computation, when $\mu = 0.05$ and $\mu = 0.005$, we had to take 64 and 128 time steps respectively when $N = 128$. Halving the mesh width required halving the time step in order to avoid numerical instabilities. In the implicit computation, no such time step refinement was necessary. Whatever the mesh number N, we could take the number of time steps to be 8 or 16.

In the explicit scheme, two $\mathscr{L}P$ operations are performed for each time step. Since the time step increases proportionally to N, so does $N_{\mathscr{L}P}$. In implicit computations, we see that $N_{\mathscr{L}P}$ stays approximately constant as one refines the mesh width. Thus, the ratios between the computational cost for the explicit and implicit methods increase by roughly a factor of 2 as the mesh width is halved. The fully implicit scheme requires 10–20% more $\mathscr{L}P$ operations than the semi-implicit scheme. This comes from the overhead associated with calculating the right hand side of the linear equation (109) for each outer iteration in the fully implicit scheme.

The above difference between the implicit and explicit methods is reflected in the execution time for each computational run as shown in Table 6. We see that the benefit of using the implicit scheme increases significantly when a finer mesh width is used.

7. Conclusions

In the above, we described a semi-implicit and a fully implicit second-order accurate immersed boundary method. We saw that these implicit methods give us a natural way to deal with mass on the immersed elastic fibers.

The virtue of an implicit scheme is that one does not have to take small time steps to avoid numerical instability. We have demonstrated that the proposed implicit methods allow us to take very large time steps compared with expli-

Table 6
Execution time for each computation in seconds rounded to three significant digits

	$\mu = 0.05$					$\mu = 0.005$				
N	exp.	$N_{\mathrm{T}} = 8$		$N_{\mathrm{T}} = 16$		exp.	$N_{\mathrm{T}} = 8$		$N_{\mathrm{T}} = 16$	
		si	fi	si	fi		si	fi	si	fi
64	3.59	2.27	2.84	3.91	4.91	6.98	3.16	4.00	5.17	6.61
128	23.9	6.89	9.08	10.8	15.5	46.5	10.0	13.1	16.0	21.3
256	255	30.4	42.8	52.5	73.9	508	48.2	62.3	81.6	105
512	3030	161	225	274	392	6050	253	331	435	561

ex: explicit, si: semi-implicit, fi: fully implicit.

cit computations. One hopes that this benefit of being able to take larger time steps offsets the complication of having to solve algebraic equations at each time step. We have shown by way of computational experiments that this is indeed the case with the present implicit methods. As one halves the grid spacing, the iteration count of the implicit methods stays approximately constant. The explicit scheme, on the other hand, requires on average twice as many time steps when the grid spacing is halved, and therefore, incurs twice as much computational cost. We believe that the current methods will be efficient alternatives to explicit schemes especially when one has stiff elastic forces or a fine spatial grid.

The present method also gives us an alternative way to deal with mass on the immersed elastic fibers. The present methods make use of the d'Alembert force (i.e., $-M\frac{\partial^2 X}{\partial t^2}$) to incorporate mass of the fibers as is done in [30]. The use of the d'Alembert force in the context of an explicit method is restricted to small fiber mass to avoid numerical instability [30]. In the present methods, large fiber mass (in our computational example, the presence of the immersed structure doubled the total mass in the computational domain) does not cause serious problems. The two other methods to deal with fiber mass are the "mass spreading method" [28,30] and the "penalty method" [29,30]. The "mass spreading method" spreads the fiber mass on to the fluid grid. The Fast Fourier Transform can no longer be used to perform the requisite linear algebra, due to the non-uniformity of the fluid mass density. The "penalty method" introduces a spring with a large stiffness constant whose role is to couple the massless immersed boundary to a collection of point masses. The present implicit methods avoid such complications.

The success of the implicit methods relies in large part on the availability of a good preconditioner. The elastic properties of the immersed elastic structure considered here allowed the use of an efficient preconditioner. It would be a future challenge to develop preconditioners for immersed structures with more general elastic properties.

Acknowledgements

The authors would like to thank Aaron Fogelson, Luca Heltai, Elijah Newren and Xiaodong Wang for helpful discussion. Y.M. was supported by the Henry McCracken fellowship of New York University during the course of this work.

References

[1] D.M. McQueen, C.S. Peskin, Heart simulation by an immersed boundary method with formal second order accuracy and reduced numerical viscosity, in: Mechanics for a New Millennium, Proceedings of the International Conference on Theoretical and Applied Mechanics (ICTAM) 2000, Kluwer Academic Publishers, 2001.

[2] R.P. Beyer, A computational model of the cochlea using the immersed boundary method, J. Comput. Phys. 98 (1992) 145–162.

[3] E. Givelberg, Modeling elastic shells immersed in fluid, Ph.D. thesis, Courant Institute of Mathematical Sciences, New York University, 1997.

[4] L.J. Fauci, A.L. Fogelson, Truncated Newton methods and the modeling of complex immersed elastic structures, Comm. Pure Appl. Math. 46 (1993) 787–816.

[5] N.T. Wang, A.L. Fogelson, Computational methods or continuum models of platelet aggregation, J. Comput. Phys. 151 (2) (1999) 649–675.

[6] H. Yu, Three-dimensional computational modeling and simulation of platelet aggregation on parallel computers, Ph.D. thesis, University of Utah, 2000.

[7] L.J. Fauci, C.S. Peskin, A computational model of aquatic animal locomotion, J. Comput. Phys. 77 (1988) 85–108.

[8] L.J. Fauci, Interaction of oscillating filaments – a computational study, J. Comput. Phys. 86 (1990) 294–313.

[9] M. Hopkins, L.J. Fauci, A computational model of the collective fluid dynamics of motile microorganisms, J. Fluid Mech. 455 (2002) 149–174.

[10] R. Cortez, N. Cowen, R. Dillon, L.J. Fauci, Simulation of swimming organisms: coupling internal mechanics with external fluid dynamics, Comput. Sci. Engrg. 6 (3) (2004) 38–45.

[11] L.A. Miller, C.S. Peskin, When vortices stick: an aerodynamic transition in tiny insects, J. Exp. Biol. 207 (2004) 3073–3088.

[12] L.A. Miller, C.S. Peskin, A computational fluid dynamics of 'clap and fling' in small insects, J. Exp. Biol. 208 (2005) 195–212.

[13] A.L. Fogelson, C.S. Peskin, A fast numerical method for solving three-dimensional Stokes' equations in the presence of suspended particles, J. Comput. Phys. 79 (1988) 50–69.

[14] J.M. Stockie, S.I. Green, Simulating the motion of flexible pulp fibres using the immersed boundary method, J. Comput. Phys. 147 (1998) 147–165.

[15] E.N. Jung, C.S. Peskin, Two-dimensional simulations of valveless pumping using the immersed boundary method, SIAM J. Sci. Comput. 23 (2001) 19–45.

[16] M.E. Rosar, C.S. Peskin, Fluid flow in collapsible elastic tubes: a three-dimensional numerical method, New York J. Math. 7 (2001) 281–302.

[17] S. Lim, C.S. Peskin, Simulations of the whirling instability by the immersed boundary method, SIAM J. Sci. Comput. 25 (2004) 2066–2083.

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

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

[20] J.M. Stockie, B.R. Wetton, Analysis of stiffness in the immersed boundary method and implications for time-stepping schemes, J. Comput. Phys. 154 (1999) 41–64.

[21] C. Tu, C.S. Peskin, Stability and instability in the computation of flows with moving immersed boundaries: a comparison of three methods, SIAM J. Sci. Statist. Comput. 13 (1992) 1361–1376.

[22] A.A. Mayo, C.S. Peskin, An implicit numerical method for fluid dynamics problems with immersed elastic boundaries, in: Fluid Dynamics in Biology: Proceedings of the AMS-IMS-SIAM Joint Summer Research Conference of Biofluid dynamics, Contemporary Mathematics, vol. 141, AMS, 1993, pp. 261–277.

[23] E.P. Newren, A. Fogelson, R. Guy, R. Kirby, Unconditionally stable discretizations of the immersed boundary equations, J. Comput. Phys. 222 (2007) 702–719.

[24] X. Wang, From immersed boundary method to immersed continuum method, Int. J. Multiscale Comput. Engrg. 4 (2006) 127–145.

[25] D. Boffi, L. Gastaldi, L. Heltai, Stability results for the finite element approach to the immersed boundary method, in: K.J. Bathe (Ed.), Proceeding of the Third M.I.T Conference on Computational Fluid and Solid Mechanics, 2005, pp. 93–96.

[26] L. Heltai, The finite element immersed boundary method, Ph.D. thesis, Università di Pavia, Dipartimento di Matematica "F. Casorati", submitted.

[27] 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 (2005) 75–105.

[28] L. Zhu, C.S. Peskin, Simulation of a flexible flapping filament in a flowing soap film by the immersed boundary method, J. Comput. Phys. 79 (2) (2002) 452–468.

[29] Y. Kim, The penalty immersed boundary method and its application to aerodynamics, Ph.D. thesis, New York University, 2003.

[30] Y. Kim, L. Zhu, X. Wang, C. Peskin, On various techniques for computer simulation of boundaries with mass, in: K. Bathe (Ed.), Computational Fluid and Solid Mechanics 2003, Elsevier Science Ltd., 2003, pp. 1746–1750.

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

[32] A.J. Chorin, Numerical solution of the Navier–Stokes equations, Math. Comput. 22 (1968) 745–762.

[33] A.J. Chorin, On the convergence of discrete approximations to the Navier–Stokes equations, Math. Comput. 23 (1969) 341–353.

[34] Y. Saad, Iterative Methods for Sparse Linear Systems, second ed., SIAM, 2003.

[35] H.A. van der Vorst, Iterative Krylov methods for large linear systems, Cambridge Monographs on Applied and Computational Mathematics, vol. 13, Cambridge University Press, 2003.

[36] Y. Saad, M.H. Schultz, GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems, SIAM J. Sci. Statist. Comput. 7 (1986) 856–869.

[37] H.A. van der Vorst, Bi-CGSTAB: a fast smoothly converging variant of Bi-CG for the solution of non-symmetric linear systems, SIAM J. Sci. Statist. Comput. 13 (1992) 631–644.

[38] P. Sonneveld, CGS, a fast lanczos-type solver for nonsymmetric linear systems, SIAM J. Sci. Statist. Comput. 10 (1989) 36–52.