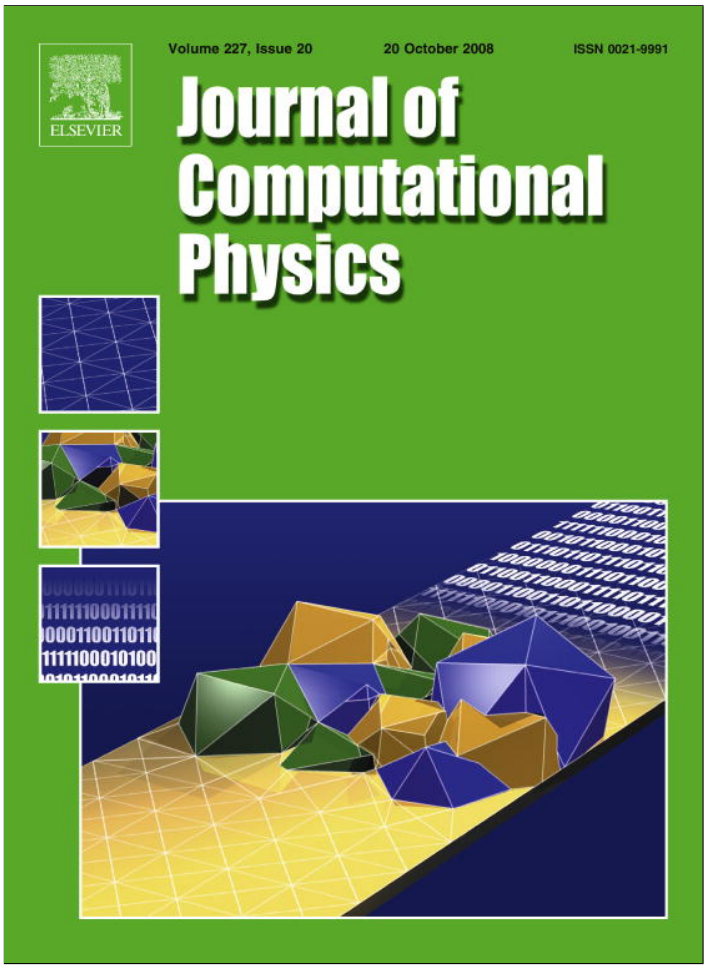

Journal of Computational Physics 227 (2008) 8968–8991

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

An efficient semi-implicit immersed boundary method for the Navier–Stokes equations

Thomas Y. Hou [a,*], Zuoqiang Shi [a,b]

[a] Applied and Computational Mathematics, 1200 California Blvd., MC 217-50, Caltech, Pasadena, CA 91125, United States
[b] Zhou Pei-Yuan Center for Applied Mathematics, Tsinghua University, Beijing 100084, China

ARTICLE INFO

Article history:
Received 13 January 2008
Received in revised form 7 July 2008
Accepted 8 July 2008
Available online 17 July 2008

Keywords:
Immersed boundary method
Navier–Stokes equations
Implicit discretization

ABSTRACT

The immersed boundary method is one of the most useful computational methods in studying fluid structure interaction. On the other hand, the Immersed Boundary method is also known to require small time steps to maintain stability when solved with an explicit method. Many implicit or approximately implicit methods have been proposed in the literature to remove this severe time step stability constraint, but none of them give satisfactory performance. In this paper, we propose an efficient semi-implicit scheme to remove this stiffness from the immersed boundary method for the Navier–Stokes equations. The construction of our semi-implicit scheme consists of two steps. First, we obtain a semi-implicit discretization which is proved to be unconditionally stable. This unconditionally stable semi-implicit scheme is still quite expensive to implement in practice. Next, we apply the small scale decomposition to the unconditionally stable semi-implicit scheme to construct our efficient semi-implicit scheme. Unlike other implicit or semi-implicit schemes proposed in the literature, our semi-implicit scheme can be solved explicitly in the spectral space. Thus the computational cost of our semi-implicit schemes is comparable to that of an explicit scheme. Our extensive numerical experiments show that our semi-implicit scheme has much better stability property than an explicit scheme. This offers a substantial computational saving in using the immersed boundary method.

1. Introduction

The immersed boundary method was originally introduced by Peskin in the 1970s to model the flow around heart valves. The method uses a uniform Eulerian grid to discretize the fluid velocity and a Lagrangian description for the immersed elastic structure. The interaction between the fluid and the elastic structure is expressed in terms of the spreading and interpolation operations by use of smoothed Delta functions. This formulation allows a single set of fluid dynamics equations to hold in the entire domain with no internal boundary conditions. The immersed boundary method has now evolved into a general useful method and has been used in a wide variety of applications, particularly in biofluid dynamics problems where complex geometries and immersed elastic membranes are present. Examples include blood flow in the heart [27,18–22,28], vibrations of the cochlear basilar membrane [3,11], platelet aggregation during clotting [10,37], aquatic locomotion [7,9,14,38,4], flow with suspended particles [8,33], and inset flight [23,24], We refer to [29] for an extensive list of applications.

Despite of its considerable success, the immersed boundary method is known to suffer from a severe time step restriction to maintain stability if an explicit or semi-implicit method is used [29,34,32]. This restriction is typically much more severe than the one that would be imposed from using an explicit discretization for the convection term in the Navier–Stokes equa-

* Corresponding author.
E-mail addresses: hou@acm.caltech.edu (T.Y. Hou), shi@acm.caltech.edu (Z. Shi).

0021-9991/$ - see front matter © 2008 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2008.07.005

tions. The instability is also known to arise from large elastic force and small viscosity [34]. Much effort has been made to remove this restriction. Some implicit and approximately implicit methods have been proposed in the literature [35,25,17]. However, none of them give satisfactory performance. The computational cost of using these implicit or approximately implicit schemes is still too high to be effective in a practical computation. To date, almost all practical computations using the immersed boundary method have been performed using an explicit discretization.

In this paper, we develop an efficient semi-implicit scheme to remove the time step restriction of the immersed boundary method in a two-dimensional, incompressible Navier–Stokes flow. There are two important ingredients in deriving our semi-implicit scheme. The first one is to obtain a semi-implicit discretization for the immersed boundary problem which is proved to be unconditionally stable. More precisely, we prove that the energy norm of the numerical solution is a non-increasing function of time. This is a weaker result than proving that the difference between two solutions in the energy norm can be bounded in terms of the energy norm of their difference at time zero. The second one is to perform the small scale decomposition to this unconditionally stable semi-implicit discretization to obtain our efficient semi-implicit scheme. An important feature of our small scale decomposition is that the leading order term, which is discretized implicitly, can be expressed as a convolution operator. This property enables us to solve for the implicit solution explicitly using the Fourier transformation. Thus, the computational cost of our semi-implicit scheme is comparable to that of an explicit method. This offers a significant computational saving in using the immersed boundary method.

The semi-implicit scheme that we present in this paper is a generalization of a semi-implicit scheme for the 2D Stokes equations recently introduced by the authors in [15]. One of the main contributions of this paper is to find an unconditionally stable semi-implicit discretization of the immersed boundary method for the 2D Navier–Stokes equations. Using this unconditionally stable semi-implicit scheme as a building block, we obtain an efficient semi-implicit scheme that is free from the usual CFL stability restriction when the convection term is discretized explicitly. We note that when viscosity is small and elastic force is large, the velocity field could be quite large. In this case, removing the CFL stability constraint could provide significant computational saving. This is also confirmed by our numerical experiments. The semi-implicit discretization that we use for the convection term is a variant of the alternating directional implicit (ADI) scheme [30] which can be solved efficiently. This property allows us to implement our semi-implicit scheme very efficiently.

As we mentioned earlier, the small scale decomposition plays an essential role in our construction of an efficient semi-implicit scheme. This method was first developed by Hou et al. [12,13] to remove the stiffness from interfacial flow with surface tension. The coupling between the elastic boundary and the fluid makes it much more difficult to remove the stiffness induced by the elastic force in the Immersed Boundary method. We overcome this difficulty by designing a unconditionally stable semi-implicit discretization that decouples the stiffness induced by the elastic force from the fluid flow. The decoupling of the stiffness induced by the elastic force from the fluid flow makes it possible for us to apply the small scale decomposition to this unconditionally stable semi-implicit discretization. This leads to an efficient semi-implicit scheme with the desirable stability property.

We remark that very recently Newren et al. have obtained an unconditionally stable method for the 2D Stokes flow with linear force in [26]. Their study sheds new light to the stability property of the immersed boundary method and clarifies some of the confusions regarding the stability property of the immersed boundary method. On the other hand, since they treat the convection term explicitly, their semi-implicit discretization is *not* unconditionally stable for the Navier–Stokes equations. Moreover, since they do not use the Small Scale Decomposition in their method, the computational cost of their semi-implicit method is still quite expensive. The gain of their semi-implicit method over an explicit discretization is rather limited.

To illustrate the stability property of our semi-implicit scheme, we apply our method to several prototype problems and test our scheme for a wide range of elastic coefficients and viscosity coefficients. Our extensive computational experiments confirm that our semi-implicit scheme removes the high frequency stability constraint induced by the elastic force. The stability of our semi-implicit scheme is essentially independent of the meshsize. The computational saving over an explicit scheme is very substantial. The computational gain is even bigger as the stiffness of the Immersed Boundary method becomes more severe. In the most severe case we have tested with the elastic coefficient $S_b = 10^5$ and the Reynolds number $Re = 60,000$ on a 512×512 grid, the maximum time step of our semi-implicit scheme is 1160 times larger than the explicit scheme. Even after we take into account the extra cost in inverting the semi-implicit solution, our semi-implicit scheme still runs 683 times faster than the explicit scheme. The saving is even larger as we increase the resolution, or the elastic coefficient, and/or the Reynolds number.

As an application, we apply our semi-implicit scheme to compute the vortex sheet problem in a viscous fluid with Reynolds number $Re = 10,000$, which is very challenging computationally. We study the large time behavior of the vortex sheet solution for different values of Weber numbers. For a modest value of Weber number $We = 40/\pi$, the interface does not roll up. Instead it deforms into elongated fingers that penetrate each fluid into the other. This is similar to the result of [13,36]. But for a larger Weber number $We = 400/\pi$, we observe that the vortex sheet rolls up and forms a thin neck near the tip of the finger. The neck below the tip becomes thinner in time. Unlike the inviscid vortex sheet with surface tension which forms a finite time pinching singularity in the neck of its finger, the thickness of the neck does not appear to approach to zero in a finite time for the viscous vortex sheet. It is likely that the neck is being stabilized by the viscosity of the fluid, and the vortex sheet rollup may continue indefinitely. This confirms the result previously obtained by Ceniceros and Roma in [6]. The behavior of the viscous vortex sheet seems to be qualitatively different from the inviscid vortex sheet with surface tension [13,6].

This paper is organized as follows. First, we review the classical formulation of the immersed boundary method in Section 2 and introduce the arclength and tangent angle formulation. In Section 3, we introduce our unconditionally stable semi-implicit discretization and prove its unconditional stability. The small scale decomposition is applied to the unconditionally stable scheme introduced in Section 4. We give a complete description of our semi-implicit scheme in Section 5. In Section 6, we apply our semi-implicit scheme to the viscous vortex sheet problem. In Section 7, we present our extensive numerical experiments to study the stability of our semi-implicit scheme and compare the performance of our method with the explicit scheme and the unconditionally stable semi-implicit scheme. Numerical results for the viscous vortex sheet are also given in this section. Some concluding remarks are given in Section 8.

2. Formulation

In this section, we will review the classical formulation of the Immersed Boundary method and describe its spatial discretization. We will also introduce the arclength and tangent angle formulation for the Immersed Boundary method. The use of the arclength and tangent angle formulation simplifies the construction of our semi-implicit scheme and makes it easier to apply the small scale decomposition to our unconditionally stable discretization.

2.1. Review of the immersed boundary method

We consider a viscous incompressible fluid in a two-dimensional domain Ω. An immersed massless elastic boundary is a closed simple curve Γ contained in Ω. We assume that the elastic boundary is parameterized by $\mathbf{X}(\alpha,t), 0 \leqslant \alpha \leqslant L_b$, satisfying $\mathbf{X}(0,t) = \mathbf{X}(L_b,t)$. Here α is a Lagrangian variable. The governing equations are the incompressible Navier–Stokes equations which interact with the elastic boundary through the elastic force $\mathbf{f}(\mathbf{x},t)$:

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u}\right) = -\nabla p + \mu\triangle\mathbf{u} + \mathbf{f}(\mathbf{x},t), \tag{1}$$

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

$$\frac{\partial \mathbf{X}}{\partial t}(\alpha,t) = \mathbf{u}(\mathbf{X}(\alpha,t),t), \tag{3}$$

where $\mathbf{u}$ is the fluid velocity, p is the pressure, ρ and μ are constant fluid density and viscosity, respectively. The force density is typically modeled as a Dirac delta function along the boundary as follows:

$$\mathbf{f}(\mathbf{x},t) = \int_0^{L_b} \mathbf{F}(\alpha,t)\delta(\mathbf{x}-\mathbf{X}(\alpha,t))\mathrm{d}\alpha, \tag{4}$$

where δ is the two-dimensional Dirac delta function and

$$\mathbf{F}(\alpha,t) = \frac{\partial}{\partial\alpha}(T\boldsymbol{\tau}), \tag{5}$$

$$T = T\left(\left|\frac{\partial \mathbf{X}}{\partial\alpha}\right|\right). \tag{6}$$

The function T is chosen to satisfy the Hook's law:

$$T = S_b\left(\left|\frac{\partial \mathbf{X}}{\partial\alpha}\right| - 1\right), \tag{7}$$

where S_b is the elastic coefficient of the boundary, and τ is the unit tangent vector along the boundary. This choice of force density has been used widely in the previous studies of the immersed boundary method, see e.g. [16,31,35].

The interaction of the fluid and the elastic force is done through the spreading and interpolation operations, which are defined as follows:

$$L(\mathbf{X})(g(\alpha))(\mathbf{x}) = \int_\Gamma g(\alpha)\delta(\mathbf{x}-\mathbf{X}(\alpha,t))\mathrm{d}\alpha, \tag{8}$$

$$L^*(\mathbf{X})(u(\mathbf{x}))(\alpha) = \int_\Omega u(\mathbf{x})\delta(\mathbf{x}-\mathbf{X}(\alpha,t))\mathrm{d}\mathbf{x}. \tag{9}$$

It is easy to show that L and L^* are adjoint operators [26,15]:

$$\langle \mathbf{u}(\mathbf{x}), L(\mathbf{X})(g(\alpha))\rangle_\Omega = \langle L^*(\mathbf{X})(u(\mathbf{x})), g(\alpha)\rangle_\Gamma, \tag{10}$$

where the inner product are defined as follows:

$$\langle u,v\rangle_\Omega = \int_\Omega u(\mathbf{x})v(\mathbf{x})\mathrm{d}\mathbf{x}, \quad \langle f,g\rangle_\Gamma = \int_\Gamma f(\alpha)g(\alpha)\mathrm{d}\alpha. \tag{11}$$

Using the interpolation operator, we can rewrite (3) in the following way:

$$\frac{\partial \mathbf{X}}{\partial t}(\alpha,t) = L^*(\mathbf{X})(\mathbf{u}(\mathbf{x},t))(\alpha,t). \tag{12}$$

Eqs. 3,4 represent the interaction of the fluid and the elastic boundary. At a given time, the elastic boundary interacts with the fluid through its force along the boundary and the immersed boundary is convected by the fluid velocity. The force density is then updated by the new configuration of the boundary.

2.2. Spatial discretization

We consider the fluid in an unit square domain with doubly periodic boundary conditions. We use the spectral method to discretize the diffusion term in the Navier–Stokes equations with a uniform $N \times N$ Cartesian grid. A finite difference method can also be used if non-periodic boundary conditions are used [29]. The discretization of the convection term will be described in Section 3.

Next, we describe the discretization of the immersed boundary. We employ a Lagrangian grid with gridsize $\Delta\alpha$. The number of grid points along the boundary is N_b. When the interface is closed, the solution is periodic along the interface. Thus it makes sense to use the spectral method to discretize the solution along the immersed boundary. We remark that a finite difference discretization can be also used [29]. To discretize the spreading and interpolation operators, we need to introduce a discrete delta function. The discrete delta function we use was introduced by Peskin in [29]:

$$\delta_h(x,y) = \frac{1}{h^2}\phi\left(\frac{x}{h}\right)\phi\left(\frac{y}{h}\right), \tag{13}$$

where

$$\phi(r) = \begin{cases} \frac{1}{8}\left(3 - 2|r| + \sqrt{1+4|r|-4r^2}\right), & |r| \leqslant 1, \\ \frac{1}{8}\left(5 - 2|r| - \sqrt{-7+12|r|-4r^2}\right), & 1 \leqslant |r| \leqslant 2, \\ 0, & |r| > 2. \end{cases} \tag{14}$$

Using the above discrete delta function, we discretize the spreading and interpolation operators as follows

$$L_h(\mathbf{X})(g(\alpha))(\mathbf{x}) = \sum_{\alpha \in \mathscr{G}_\Gamma} g(\alpha)\delta_h(\mathbf{x} - \mathbf{X}(\alpha,t))\Delta\alpha, \tag{15}$$

$$L_h^*(\mathbf{X})(u(\mathbf{x}))(\alpha) = \sum_{\mathbf{x} \in \mathscr{G}_\Omega} u(\mathbf{x})\delta_h(\mathbf{x} - \mathbf{X}(\alpha,t))h^2. \tag{16}$$

The above summation is over grid points on the immersed interface Γ in (15) and over grid points in Ω in (16). We can show that L_h and L_h^* are still adjoint operators [15]:

$$\langle u(\mathbf{x}), L(\mathbf{X})(g(\alpha))\rangle_{\Omega_h} = \langle L_h^*(\mathbf{X})(u(\mathbf{x})), g(\alpha)\rangle_{\Gamma_h}, \tag{17}$$

where the discrete inner products are defined as follows:

$$\langle f, g\rangle_{\Gamma_h} = \sum_{\alpha \in \mathscr{G}_\Gamma} f(\alpha)g(\alpha)\Delta\alpha, \tag{18}$$

$$\langle u, v\rangle_{\Omega_h} = \sum_{\mathbf{x} \in \mathscr{G}_\Omega} u(\mathbf{x})v(\mathbf{x})h^2. \tag{19}$$

2.3. The arclength-tangent angle formulation

The small scale decomposition plays an important role in the construction of our efficient semi-implicit scheme. To make it easier to perform the small scale decomposition, we reformulate the immersed boundary method using the arclength-tangent angle formulation. The arclength-tangent angle formulation has been used successfully by Hou et al. [12] to remove the stiffness of interfacial flows with surface tension.

We parameterize the interface by $\mathbf{X}(\alpha,t), \alpha \in [0, L_b]$. The arclength derivative, s_α, and the tangent vector, θ are defined as follows

$$s_\alpha(\alpha,t) = |\mathbf{X}_\alpha(\alpha,t)|, \tag{20}$$

$$(x_\alpha(\alpha,t), y_\alpha(\alpha,t)) = s_\alpha(\alpha,t)(\cos\theta(\alpha,t), \sin\theta(\alpha,t)). \tag{21}$$

Let U and V be the normal and tangent components of the velocity field. We can rewrite the interface equation as follows:

$$\frac{\partial \mathbf{X}}{\partial t} = \mathbf{u}(\mathbf{X},t) = U\mathbf{n} + V\boldsymbol{\tau}, \tag{22}$$

where $\boldsymbol{\tau}$ and $\mathbf{n}$ are the unit tangent and normal vectors of the interface, respectively. By using the Frenèt formula, $\frac{\partial \boldsymbol{\tau}}{\partial s} = k\mathbf{n}, \frac{\partial \mathbf{n}}{\partial s} = -k\boldsymbol{\tau}$, one can derive the equivalent evolution equations for s_α and θ [12]:

$$(s_\alpha)_t = V_\alpha - \theta_\alpha U, \tag{23}$$

$$\theta_t = \frac{U_\alpha}{s_\alpha} + \frac{V\theta_\alpha}{s_\alpha}. \tag{24}$$

Given s_α, θ and a reference point on the interface, we can reconstruct the interface Γ by integrating (21) with respect to α. Using the $s_\alpha - \theta$ formulation, we can reformulate the immersed boundary problem as follows:

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u}\right) = -\nabla p + \mu\triangle\mathbf{u} + L(\mathbf{X})(\mathbf{F}(s_\alpha,\theta)), \tag{25}$$

$$\nabla\cdot\mathbf{u} = 0, \tag{26}$$

$$U = L^*(\mathbf{X})(\mathbf{u}(\mathbf{x}))\cdot\mathbf{n}, \tag{27}$$

$$V = L^*(\mathbf{X})(\mathbf{u}(\mathbf{x}))\cdot\boldsymbol{\tau}, \tag{28}$$

$$(s_\alpha)_t = V_\alpha - \theta_\alpha U, \tag{29}$$

$$\theta_t = \frac{U_\alpha}{s_\alpha} + \frac{V\theta_\alpha}{s_\alpha}, \tag{30}$$

where

$$\mathbf{F}(s_\alpha,\theta) = \frac{\partial}{\partial\alpha}(T\boldsymbol{\tau}) = S_b(s_{\alpha,\alpha}\boldsymbol{\tau} + (s_\alpha - 1)\theta_\alpha\mathbf{n}). \tag{31}$$

3. A unconditionally stable semi-implicit discretization

In this section, we will describe our unconditionally stable semi-implicit discretization of the Immersed Boundary method for the incompressible Navier–Stokes equations. We first introduce the method in Section 3.1 and then prove its unconditional stability in Section 3.2.

3.1. The description of the method

In this subsection, we describe our unconditionally stable semi-implicit scheme. We discretize the Immersed Boundary method by using a time splitting method. In the first step, we only discretize the convection term. In the second step, we discretize the Immersed Boundary method for the Stokes equations. As we mentioned before, we can use a spectral method or a finite difference method to discretize the solution in space. Below we describe the algorithm of our unconditionally stable semi-implicit scheme from t^n to t^{n+1}. Assume the velocity field and the interface position are already known at t^n. We update the solution from t^n to t^{n+1} using the following three steps:

Step 1: Discretization of the convection term.

$$\frac{\tilde{\mathbf{u}}^{n+1,1} - \mathbf{u}^n}{\Delta t} + \frac{1}{2}u_1^n D_{h,1}^0\tilde{\mathbf{u}}^{n+1,1} + \frac{1}{2}D_{h,1}^0(u_1^n\tilde{\mathbf{u}}^{n+1,1}) = 0, \tag{32}$$

$$\frac{\tilde{\mathbf{u}}^{n+1} - \tilde{\mathbf{u}}^{n+1,1}}{\Delta t} + \frac{1}{2}u_2^n D_{h,2}^0\tilde{\mathbf{u}}^{n+1} + \frac{1}{2}D_{h,2}^0(u_2^n\tilde{\mathbf{u}}^{n+1}) = 0, \tag{33}$$

where $D_{h,\beta}^0$ is the central difference approximation of the derivative operator along the x_β direction,

$$(D_{h,\beta}^0\ \phi)(\mathbf{x}) = \frac{\phi(\mathbf{x} + h\mathbf{e}_\beta) - \phi(\mathbf{x} - h\mathbf{e}_\beta)}{2h}, \tag{34}$$

where $\mathbf{e}_\beta$ is the unit vector along the coordinate axis with $\beta = 1, 2$. The above semi-implicit discretization of the convection term is a variant of the alternating directional implicit (ADI) scheme [30]. This special form of the discretization of the convection term is inspired by a similar explicit discretization of the convection term introduced in [29].

Step 2: Update of $\mathbf{u}^{n+1}, p^{n+1}$ and s_α^{n+1}.

$$\rho\frac{\mathbf{u}^{n+1} - \tilde{\mathbf{u}}^{n+1}}{\Delta t} = -\nabla_h p^{n+1} + \mu\nabla_h^2\mathbf{u}^{n+1} + L_{h,n}(\mathbf{F}(s_\alpha^{n+1},\theta^n;\boldsymbol{\tau}^n,\mathbf{n}^n)), \tag{35}$$

$$\nabla_h^2 p^{n+1} = \frac{1}{\Delta t}\nabla_h\cdot(L_{h,n}(\mathbf{F}(s_\alpha^{n+1},\theta^n;\boldsymbol{\tau}^n,\mathbf{n}^n))\Delta t + \rho\tilde{\mathbf{u}}^{n+1}), \tag{36}$$

$$V^{n+1} = L_{h,n}^*(\mathbf{u}^{n+1})\cdot\boldsymbol{\tau}^n \tag{37}$$

$$U^{n+1} = L_{h,n}^*(\mathbf{u}^{n+1})\cdot\mathbf{n}^n, \tag{38}$$

$$\frac{s_\alpha^{n+1} - s_\alpha^n}{\Delta t} = D_{\Delta\alpha}V^{n+1} - D_{\Delta\alpha}\theta^n U^{n+1}, \tag{39}$$

where $\boldsymbol{\tau}^n = (\cos(\theta^n), \sin(\theta^n))$, $\mathbf{n}^n = (-\sin(\theta^n), \cos(\theta^n))$, $L_{h,n} = L_h(\mathbf{X}^n)$, $L^*_{h,n} = L^*_h(\mathbf{X}^n)$, ∇_h and $D_{\Delta\alpha}$ are discrete derivative operators for the Eulerian grid and the Lagrangian grid, respectively, and

$$\mathbf{F}(s_\alpha^{n+1}, \theta^n; \boldsymbol{\tau}^n, \mathbf{n}^n) = S_b(D_{\Delta\alpha} s_\alpha^{n+1} \boldsymbol{\tau}^n + (s_\alpha^{n+1} - 1) D_{\Delta\alpha}\theta^n \mathbf{n}^n). \tag{40}$$

Step 3: Update of θ^{n+1}.

After we have obtained $\mathbf{u}^{n+1}, p^{n+1}$ and s_α^{n+1}, we update θ at t^{n+1} using the following semi-implicit scheme:

$$\rho\frac{\bar{\mathbf{u}}^{n+1} - \tilde{\mathbf{u}}^{n+1}}{\Delta t} = -\nabla_h \bar{p}^{n+1} + \mu\nabla_h^2 \bar{\mathbf{u}}^{n+1} + L_{h,n}\big(\mathbf{F}\big(s_\alpha^{n+1}, \theta^{n+1}; \boldsymbol{\tau}^n, \mathbf{n}^n\big)\big), \tag{41}$$

$$\nabla_h^2\, \bar{p}^{n+1} = \frac{1}{\Delta t}\nabla_h \cdot (L_{h,n}(\mathbf{F}(s_\alpha^{n+1}, \theta^{n+1}; \boldsymbol{\tau}^n, \mathbf{n}^n))\Delta t + \rho\tilde{\mathbf{u}}^{n+1}), \tag{42}$$

$$\overline{V}^{n+1} = L^*_{h,n}(\bar{\mathbf{u}}^{n+1}) \cdot \boldsymbol{\tau}^n \tag{43}$$

$$\overline{U}^{n+1} = L^*_{h,n}(\bar{\mathbf{u}}^{n+1}) \cdot \mathbf{n}^n, \tag{44}$$

$$\frac{\theta^{n+1} - \theta^n}{\Delta t} = \frac{1}{s_\alpha^{n+1}}(D_{\Delta\alpha}\overline{U}^{n+1} + D_{\Delta\alpha}\theta^n \overline{V}^{n+1}). \tag{45}$$

where

$$\mathbf{F}\big(s_\alpha^{n+1}, \theta^{n+1}; \boldsymbol{\tau}^n, \mathbf{n}^n\big) = S_b\big(D_{\Delta\alpha} s_\alpha^{n+1} \boldsymbol{\tau}^n + \big(s_\alpha^{n+1} - 1\big) D_{\Delta\alpha}\theta^{n+1} \mathbf{n}^n\big). \tag{46}$$

It is important to note that the above discretization is not fully implicit. In fact, both the spreading and interpolation operators are evaluated at the interface $\mathbf{X}^n$ from the previous time step. Moreover, when solve the s_α^{n+1} and $\mathbf{u}^{n+1}$, in (35)–(39), we use θ^n instead of θ^{n+1} to evaluate the force density. This makes our semi-implicit discretization linear with respect to the implicit solution variables, $\mathbf{u}^{n+1}, \theta^{n+1}$, and s_α^{n+1}. The above semi-implicit discretization essentially decouples the stiffness induced by the elastic force from the fluid equations. This enables us to remove the stiffness of the immersed boundary method effectively by applying the small scale decomposition and arclength-tangent angle formulation as was done in [12].

3.2. Stability analysis

In this section we will analyze the stability of the semi-implicit discretization (32)–(45) in the energy norm. We will prove that the above semi-implicit discretization is unconditionally stable in the sense that the total energy is non-increasing.

First, we define the total energy of the physical system. The total energy includes the kinetic energy K and the potential energy P, which are defined below:

$$K = \frac{1}{2}\rho\langle\mathbf{u}, \mathbf{u}\rangle_{\Omega_h} = \frac{\rho}{2}\sum_{i,j=1}^{N} \mathbf{u}_{ij} \cdot \mathbf{u}_{ij} h^2, \tag{47}$$

$$P = \frac{1}{2} S_b \langle s_\alpha - 1, s_\alpha - 1\rangle_{\Gamma_h} = \frac{S_b}{2}\sum_{j=1}^{N_b} (s_{\alpha,j} - 1)^2 \Delta\alpha. \tag{48}$$

The total energy is then defined as

$$E = K + P. \tag{49}$$

Theorem 1. *The semi-implicit scheme (32)–(45) is unconditionally stable in the sense that the total energy is a non-increasing function of time, i.e. $E^{n+1} \leqslant E^n$ for all $n \geqslant 0$.*

Proof of Theorem 1. We first introduce an intermediate kinetic energy as follows:

$$\widetilde{K}^{n+1} = \frac{1}{2}\rho\langle\tilde{\mathbf{u}}^{n+1}, \tilde{\mathbf{u}}^{n+1}\rangle_{\Omega_h} = \frac{\rho}{2}\sum_{i,j=1}^{N} \tilde{\mathbf{u}}_{ij}^{n+1} \cdot \tilde{\mathbf{u}}_{ij}^{n+1} h^2. \tag{50}$$

To simplify the presentation, we denote the discrete spectral derivative $D_{\Delta\alpha} g$ of a function g as g_α. Taking the discrete inner product of (35) with $\mathbf{u}^{n+1} + \tilde{\mathbf{u}}^{n+1}$, we obtain

$$\begin{aligned}
K^{n+1} - \widetilde{K}^{n+1} &= \frac{\rho}{2}\langle\mathbf{u}^{n+1} + \tilde{\mathbf{u}}^{n+1}, \mathbf{u}^{n+1} - \tilde{\mathbf{u}}^{n+1}\rangle_{\Omega_h} \\
&= \frac{\rho}{2}\langle -\mathbf{u}^{n+1} + \tilde{\mathbf{u}}^{n+1}, \mathbf{u}^{n+1} - \tilde{\mathbf{u}}^{n+1}\rangle_{\Omega_h} + \rho\langle\mathbf{u}^{n+1}, \mathbf{u}^{n+1} - \tilde{\mathbf{u}}^{n+1}\rangle_{\Omega_h} \\
&= -\frac{\rho}{2}\langle\mathbf{u}^{n+1} - \tilde{\mathbf{u}}^{n+1}, \mathbf{u}^{n+1} - \tilde{\mathbf{u}}^{n+1}\rangle_{\Omega_h} \\
&\quad + \Delta t\Big\langle\mathbf{u}^{n+1}, -\nabla_h p^{n+1} + \mu\nabla_h^2\mathbf{u}^{n+1} + L_{h,n}\big(\mathbf{F}\big(s_\alpha^{n+1}, \theta^n; \boldsymbol{\tau}^n, \mathbf{n}^n\big)\big)\Big\rangle_{\Omega_h} \\
&= -\frac{\rho}{2}\langle\mathbf{u}^{n+1} - \tilde{\mathbf{u}}^{n+1}, \mathbf{u}^{n+1} - \tilde{\mathbf{u}}^{n+1}\rangle_{\Omega_h} + \Delta t\langle\nabla_h \cdot \mathbf{u}^{n+1}, p^{n+1}\rangle_{\Omega_h} \\
&\quad - \mu\Delta t\langle\nabla_h\mathbf{u}^{n+1}, \nabla_h\mathbf{u}^{n+1}\rangle_{\Omega_h} \\
&\quad + \Delta t\langle L^*_{h,n}(\mathbf{u}^{n+1}), \mathbf{F}(s_\alpha^{n+1}, \theta^n; \boldsymbol{\tau}^n, \mathbf{n}^n)\rangle_{\Gamma_h}.
\end{aligned} \tag{51}$$

The second term on the right hand side of (51) is zero because the discrete velocity field is divergence free, i.e. $\nabla_h \cdot \mathbf{u}^{n+1} = 0$. The fourth term can be rewritten as

$$\begin{aligned}\left\langle L^*_{h,n}(\mathbf{u}^{n+1}), \mathbf{F}(s_\alpha^{n+1}, \theta^n; \boldsymbol{\tau}^n, \mathbf{n}^n)\right\rangle_{\Gamma_h} &= \left\langle V^{n+1}\boldsymbol{\tau}^n + U^{n+1}\mathbf{n}^n, S_b\left(s_{\alpha,\alpha}^{n+1}\boldsymbol{\tau}^n + (s_\alpha^{n+1}-1)\theta_\alpha^n\mathbf{n}^n\right)\right\rangle_{\Gamma_h} \\ &= S_b\left(\left\langle V^{n+1}, s_{\alpha,\alpha}^{n+1}\right\rangle_{\Gamma_h} + \left\langle U^{n+1}, (s_\alpha^{n+1}-1)\theta_\alpha^n\right\rangle_{\Gamma_h}\right),\end{aligned} \tag{52}$$

where we have used 37, 38, and 40. Combining (51) and (52), we get

$$\begin{aligned}K^{n+1} - \widetilde{K}^{n+1} &= -\frac{\rho}{2}\langle \mathbf{u}^{n+1} - \tilde{\mathbf{u}}^{n+1}, \mathbf{u}^{n+1} - \tilde{\mathbf{u}}^{n+1}\rangle_{\Omega_h} - \mu\Delta t\langle \nabla_h \mathbf{u}^{n+1}, \nabla_h \mathbf{u}^{n+1}\rangle_{\Omega_h} \\ &\quad + S_b\Delta t\left(\left\langle V^{n+1}, s_{\alpha,\alpha}^{n+1}\right\rangle_{\Gamma_h} + \left\langle U^{n+1}, (s_\alpha^{n+1}-1)\theta_\alpha^n\right\rangle_{\Gamma_h}\right).\end{aligned} \tag{53}$$

Similarly, using (39) and performing summation by parts, we get

$$\begin{aligned}P^{n+1} - P^n &= \frac{S_b}{2}\langle s_\alpha^{n+1} + s_\alpha^n - 2, s_\alpha^{n+1} - s_\alpha^n\rangle_{\Gamma_h} \\ &= \frac{S_b}{2}\langle -s_\alpha^{n+1} + s_\alpha^n, s_\alpha^{n+1} - s_\alpha^n\rangle_{\Gamma_h} + S_b\langle s_\alpha^{n+1} - 1, s_\alpha^{n+1} - s_\alpha^n\rangle_{\Gamma_h} \\ &= -\frac{S_b}{2}\langle s_\alpha^{n+1} - s_\alpha^n, s_\alpha^{n+1} - s_\alpha^n\rangle_{\Gamma_h} + S_b\Delta t\left\langle s_\alpha^{n+1} - 1, V_\alpha^{n+1} - \theta_\alpha^n U^{n+1}\right\rangle_{\Gamma_h} \\ &= -\frac{S_b}{2}\langle s_\alpha^{n+1} - s_\alpha^n, s_\alpha^{n+1} - s_\alpha^n\rangle_{\Gamma_h} - S_b\Delta t\left(\left\langle s_{\alpha,\alpha}^{n+1}, V^{n+1}\right\rangle_{\Gamma_h} + \left\langle s_\alpha^{n+1} - 1, \theta_\alpha^n U^{n+1}\right\rangle_{\Gamma_h}\right).\end{aligned} \tag{54}$$

Adding (53) to (54), we have

$$E^{n+1} - P^n - \widetilde{K}^{n+1} = -\frac{\rho}{2}\langle \mathbf{u}^{n+1} - \tilde{\mathbf{u}}^{n+1}, \mathbf{u}^{n+1} - \tilde{\mathbf{u}}^{n+1}\rangle_{\Omega_h} - \mu\Delta t\langle \nabla_h \mathbf{u}^{n+1}, \nabla_h \mathbf{u}^{n+1}\rangle_{\Omega_h} - \frac{S_b}{2}\langle s_\alpha^{n+1} - s_\alpha^n, s_\alpha^{n+1} - s_\alpha^n\rangle_{\Gamma_h} \leqslant 0 \tag{55}$$

To prove that the total energy is non-increasing, we need to prove that

$$\widetilde{K}^{n+1} \leqslant K^n. \tag{56}$$

The key in proving (56) is the following observation:

$$\left\langle \mathbf{v}, \frac{1}{2}u_i^n D_{h,i}^0 \mathbf{v} + \frac{1}{2}D_{h,i}^0(u_i^n \mathbf{v})\right\rangle_{\Omega_h} = 0, \quad i = 1, 2, \tag{57}$$

for any vector $\mathbf{v}$. To prove (57), we use summation by parts to the second term:

$$\left\langle \mathbf{v}, \frac{1}{2}u_i^n D_{h,i}^0 \mathbf{v} + \frac{1}{2}D_{h,i}^0(u_i^n \mathbf{v})\right\rangle_{\Omega_h} = \left\langle u_i^n \mathbf{v}, \frac{1}{2}D_{h,i}^0 \mathbf{v}\right\rangle_{\Omega_h} - \left\langle \frac{1}{2}D_{h,i}^0 \mathbf{v}, u_i^n \mathbf{v}\right\rangle_{\Omega_h} = 0. \tag{58}$$

In the above summation by parts, there is no contribution from the boundary term since we use periodic boundary conditions. Using (57), we have

$$\begin{aligned}\widetilde{K}^{n+1,1} - K^n &= \frac{1}{2}\rho\langle \tilde{\mathbf{u}}^{n+1,1} + \mathbf{u}^n, \tilde{\mathbf{u}}^{n+1,1} - \mathbf{u}^n\rangle_{\Omega_h} \\ &= -\frac{1}{2}\rho\langle \tilde{\mathbf{u}}^{n+1,1} - \mathbf{u}^n, \tilde{\mathbf{u}}^{n+1,1} - \mathbf{u}^n\rangle_{\Omega_h} + \rho\langle \tilde{\mathbf{u}}^{n+1,1}, \tilde{\mathbf{u}}^{n+1,1} - \mathbf{u}^n\rangle_{\Omega_h} \\ &= -\frac{1}{2}\rho\langle \tilde{\mathbf{u}}^{n+1,1} - \mathbf{u}^n, \tilde{\mathbf{u}}^{n+1,1} - \mathbf{u}^n\rangle_{\Omega_h} - \rho\left\langle \tilde{\mathbf{u}}^{n+1,1}, \frac{1}{2}u_1^n D_{h,1}^0 \tilde{\mathbf{u}}^{n+1,1} + \frac{1}{2}D_{h,1}^0(u_1^n \tilde{\mathbf{u}}^{n+1,1})\right\rangle_{\Omega_h}.\end{aligned}$$

The second term of the right hand side vanishes using (57). Thus we obtain

$$\widetilde{K}^{n+1,1} - K^n = -\frac{1}{2}\rho\langle \tilde{\mathbf{u}}^{n+1,1} - \mathbf{u}^n, \tilde{\mathbf{u}}^{n+1,1} - \mathbf{u}^n\rangle_{\Omega_h} \leqslant 0, \tag{59}$$

where $\widetilde{K}^{n+1,1} = \frac{\rho}{2}\langle \tilde{\mathbf{u}}^{n+1,1}, \tilde{\mathbf{u}}^{n+1,1}\rangle_{\Omega_h}$. Similarly, we have

$$\widetilde{K}^{n+1} - \widetilde{K}^{n+1,1} = -\frac{1}{2}\rho\langle \tilde{\mathbf{u}}^{n+1} - \tilde{\mathbf{u}}^{n+1,1}, \tilde{\mathbf{u}}^{n+1} - \tilde{\mathbf{u}}^{n+1,1}\rangle_{\Omega_h} \leqslant 0. \tag{60}$$

Adding (59) to (60) gives

$$\widetilde{K}^{n+1} - K^n \leqslant 0. \tag{61}$$

Combining (55) and (61), we prove that the total energy is non-increasing

$$E^{n+1} \leqslant P^n + \widetilde{K}^{n+1} \leqslant P^n + K^n = E^n. \tag{62}$$

This proves that the semi-implicit scheme (32)–(45) is unconditionally stable in the sense that the total energy is non-increasing.

Remark 1. In our proof presented above, we have used three important properties of our semi-implicit discretization. The first property is that the discrete spreading and interpolation operators are adjoint. The second property is that the velocity field satisfies the discrete divergence free condition. The third property is identity (57).

Remark 2. We can also use other schemes to discretize the convection equation

$$\tilde{\mathbf{u}}_t + \tilde{\mathbf{u}} \cdot \nabla \tilde{\mathbf{u}} = 0, \tag{63}$$

in the first step of our discretization described in (32) and (33). As long as we use a time discretization with the following stability property

$$\widetilde{K}^{n+1} \leqslant (1 + C\Delta t)K^n, \tag{64}$$

then we can easily modify the above proof to show that $E^n \leqslant C(T)E^0$ for all $t^n \leqslant T$, where $C(T) = \exp(CT)$. In the case of the ADI discretization that we use for the convection term, we have $\widetilde{K}^{n+1} \leqslant K^n$ and the total energy is non-increasing. If we use an explicit discretization for the convection term, a CFL stability constraint is required to satisfy (64).

4. Small scale decomposition

In this section, we apply the Small Scale Decomposition to the unconditionally stable semi-implicit scheme introduced in Section 3. First, we solve for the velocity field (still denoted as $\mathbf{u}^{n+1}$) from the space-continuous version of (35) and (36) using an integral representation:

$$\begin{aligned}
\mathbf{u}^{n+1}(\mathbf{x}) &= \left(1 - \frac{\mu\Delta t}{\rho}\nabla^2\right)^{-1}\left(\tilde{\mathbf{u}}^{n+1} + \frac{\Delta t}{\rho}L_n\left(\mathbf{F}\left(s_\alpha^{n+1}, \theta^n\right)\right) - \nabla(\nabla^2)^{-1}\nabla\cdot\left(\frac{\Delta t}{\rho}L_n\left(\mathbf{F}\left(s_\alpha^{n+1}, \theta^n\right)\right) + \tilde{\mathbf{u}}^{n+1}\right)\right)\\
&= \left(1 - \frac{\mu\Delta t}{\rho}\nabla^2\right)^{-1}\left(\tilde{\mathbf{u}}^{n+1} + \nabla(\nabla^2)^{-1}\nabla\cdot\tilde{\mathbf{u}}^{n+1}\right) + \frac{\Delta t}{\rho}\left(1 - \frac{\mu\Delta t}{\rho}\nabla^2\right)^{-1}L_n\left(\mathbf{F}\left(s_\alpha^{n+1}, \theta^n\right)\right)\\
&\quad - \frac{\Delta t}{\rho}\left(1 - \frac{\mu\Delta t}{\rho}\nabla^2\right)^{-1}(\nabla^2)^{-1}\left(\nabla\nabla\cdot L_n\left(\mathbf{F}\left(s_\alpha^{n+1}, \theta^n\right)\right)\right)
\end{aligned}$$

Let E_1 and E_2 be the free space fundamental solutions in two dimensions of the following differential operators:

$$\left(1 - \frac{\mu\Delta t}{\rho}\nabla^2\right)E_1 = \delta(\mathbf{x} - \mathbf{x}'), \tag{65}$$

$$\nabla^2\left(1 - \frac{\mu\Delta t}{\rho}\nabla^2\right)E_2 = \delta(\mathbf{x} - \mathbf{x}'). \tag{66}$$

They can be expressed in terms of the modified Bessel function of the second kind [1]:

$$E_1 = \frac{\lambda^2}{2\pi}K_0(\lambda|\mathbf{x} - \mathbf{x}'|), \tag{67}$$

$$E_2 = \frac{1}{2\pi}(K_0(\lambda|\mathbf{x} - \mathbf{x}'| + \ln(|\mathbf{x} - \mathbf{x}'|)), \tag{68}$$

where $\lambda^2 = \frac{\rho}{\mu\Delta t}$ and K_0 is the modified Bessel function of the second kind. Performing integration by parts, we can further rewrite velocity $\mathbf{u}^{n+1}$ as follows:

$$\begin{aligned}
\mathbf{u}^{n+1}(\mathbf{x}) &= \left(1 - \frac{\mu\Delta t}{\rho}\nabla^2\right)^{-1}\left(\tilde{\mathbf{u}}^{n+1} + \nabla(\nabla^2)^{-1}\nabla\cdot\tilde{\mathbf{u}}^{n+1}\right) + \frac{1}{2\pi}\frac{\Delta t}{\rho}\int_{\Gamma^n}\lambda^2 K_0(\lambda|\mathbf{x} - \mathbf{X}^n(\alpha')|)\mathbf{F}\left(s_\alpha^{n+1}, \theta^n\right)\mathrm{d}\alpha'\\
&\quad - \frac{1}{2\pi}\frac{\Delta t}{\rho}\int_{\Gamma^n}G(\mathbf{x} - \mathbf{X}^n(\alpha'))\cdot\mathbf{F}\left(s_\alpha^{n+1}, \theta^n\right)\mathrm{d}\alpha',
\end{aligned} \tag{69}$$

where Γ^n is the immersed boundary at time t^n, and

$$G_{ij}(\mathbf{r}) = \frac{\delta_{ij}}{|\mathbf{r}|^2} - \frac{2r_ir_j}{|\mathbf{r}|^4} + \frac{1}{2}\lambda^2(K_0(\lambda|\mathbf{r}|) + K_2(\lambda|\mathbf{r}|))\frac{r_ir_j}{|\mathbf{r}|^2} - \lambda K_1(\lambda|\mathbf{r}|)\left(\frac{\delta_{ij}}{|\mathbf{r}|} - \frac{r_ir_j}{|\mathbf{r}|^3}\right), \tag{70}$$

K_0, K_1, K_2 are the modified Bessel functions of the second kind.

As we can see, the singular velocity integral is a complicated nonlocal integral operator. It is difficult to solve for the implicit solution if we treat the velocity integral fully implicitly. The discretization we introduce in Section 3 has made an important first step in constructing an efficient semi-implicit method by producing an unconditionally stable discretization which is linear in terms of the implicit solution at t^{n+1}. However, the resulting integral equation is still quite expensive to

solve. To derive an efficient semi-implicit scheme, we apply the small scale decomposition to the unconditionally stable discretization we obtain in Section 3.

The main idea of the small scale decomposition technique introduced in [12] is to decompose the singular velocity integral into the sum of a linear convolution operator and a remainder operator which is regular. Since the remaining operator, which could be nonlinear and nonlocal, is regular, it does not contribute to the stiffness of the problem to the leading order. Thus we do not need to treat it implicitly. The leading order singular operator, which captures accurately the high frequency spectral property of the velocity integral, can be further simplified as a linear convolution integral operator. Thus, if we treat only the leading order convolution operator implicitly, but keep the regular remainder operator explicitly, we can effectively remove the stiffness of the velocity field induced by the high frequency modes of the solution.

Below we will show how to perform such small scale decomposition for the immersed boundary method. Using the Taylor expansion and neglecting the explicit part of the integral expression (69), we obtain the following decomposition:

$$V^{n+1}(\alpha) = \mathbf{u}^{n+1}(\mathbf{X}^n(\alpha))\cdot\boldsymbol{\tau}^n(\alpha) \sim \frac{S_b\Delta t}{2\pi\rho}\int_{\Gamma^n}\lambda^2 K_0\big(\lambda s_\alpha^n|\alpha-\alpha'|\big)s_{\alpha,\alpha'}^{n+1}\,\mathrm{d}\alpha' - \frac{S_b\Delta t}{2\pi\rho}\int_{\Gamma^n}\left(\frac{1}{2}\lambda^2\big(K_0\big(\lambda s_\alpha^n|\alpha-\alpha'|\big)+K_2\big(\lambda s_\alpha^n|\alpha-\alpha'|\big)\big)-\frac{1}{(s_\alpha^n)^2(\alpha-\alpha')^2}\right)s_{\alpha,\alpha'}^{n+1}\,\mathrm{d}\alpha'. \tag{71}$$

Note that [1]

$$\frac{\mathrm{d}^2}{\mathrm{d}\alpha'^2}\left(\frac{1}{(s_\alpha^n(\alpha))^2}K_0\big(\lambda s_\alpha^n|\alpha-\alpha'|\big)\right) = \frac{1}{2}\lambda^2\big(K_0\big(\lambda s_\alpha^n|\alpha-\alpha'|\big)+K_2\big(\lambda s_\alpha^n|\alpha-\alpha'|\big)\big). \tag{72}$$

Integrating the right hand side of (71) by parts twice, we get

$$V^{n+1}(\alpha) \sim \frac{S_b\Delta t}{2\pi\rho}\int_{\Gamma^n}\lambda^2 K_0\big(\lambda s_\alpha^n|\alpha-\alpha'|\big)s_{\alpha,\alpha'}^{n+1}\,\mathrm{d}\alpha' - \frac{S_b\Delta t}{2\pi\rho(s_\alpha^n)^2}\int_{\Gamma^n}\big(K_0\big(\lambda s_\alpha^n|\alpha-\alpha'|\big)-\ln(\alpha-\alpha')\big)s_{\alpha,\alpha'\alpha'\alpha'}^{n+1}\,\mathrm{d}\alpha'. \tag{73}$$

Next, we solve for the velocity field $\bar{\mathbf{u}}^{n+1}$ from the space-continuous version of (41) and (42) using an integral representation. Following a similar procedure, we obtain the leading order term for $\overline{U}^{n+1}$ as follows:

$$\overline{U}^{n+1}(\alpha) = \bar{\mathbf{u}}^{n+1}(\mathbf{X}^n(\alpha))\cdot\mathbf{n}^n(\alpha) \sim \frac{S_b\Delta t}{2\pi\rho(s_\alpha^n)^2}\int_{\Gamma^n}\big(K_0\big(\lambda s_\alpha^n|\alpha-\alpha'|\big)-\ln(\alpha-\alpha')\big)\big((s_\alpha^{n+1}-1)\theta_{\alpha'}^{n+1}\big)_{\alpha'\alpha'}\,\mathrm{d}\alpha'. \tag{74}$$

We note that the singular operator is linear since s_α^{n+1} is updated first, but one part of the operator $K_0\big(\lambda s_\alpha^n(\alpha')|\alpha-\alpha'|\big)$ still can not be expressed as a convolution operator. Thus, we need to simplify the kernel further. First, we approximate $s_\alpha^n(\alpha)$ by $\min_\alpha s_\alpha^n(\alpha)$. Let $\beta = \lambda\min_\alpha s_\alpha^n(\alpha)$ and denote by $\mathscr{F}$ the Fourier transform. In [15], we have shown that

$$\mathscr{F}\left(\frac{1}{\pi}\int_{-\infty}^{+\infty}K_0(\beta|\alpha-\alpha'|)f(\alpha')\mathrm{d}\alpha'\right) = \frac{\widehat{f}(k)}{\sqrt{\beta^2+k^2}}, \tag{75}$$

where $\widehat{f}(k) = \mathscr{F}(f)(k) = \int_{-\infty}^{\infty}f(\alpha)\mathrm{e}^{\mathrm{i}k\alpha}\mathrm{d}\alpha$. Using (75), replacing $s_\alpha^n(\alpha)$ by $\min_\alpha s_\alpha^n(\alpha)$ and $(s_\alpha^{n+1}-1)$ by $\max_\alpha(s_\alpha^{n+1}-1)$, we obtain the following simple expressions of the leading order term for V_α^{n+1} and $\overline{U}_\alpha^{n+1}$ in the Fourier transform:

$$\widehat{V}_\alpha^{n+1}(\alpha) \sim \widehat{T}\big(s_\alpha^{n+1}\big) \equiv -\frac{S_b\Delta t}{2\rho(\min_\alpha s_\alpha^n)^2}\left(\frac{\big(\lambda\min_\alpha s_\alpha^n\big)^2k^2+k^4}{\sqrt{\big(\lambda\min_\alpha s_\alpha^n\big)^2+k^2}}-|k|^3\right)\widehat{s}_\alpha^{n+1}, \tag{76}$$

$$\widehat{\overline{U}}_\alpha^{n+1}(\alpha) \sim \widehat{S}(\theta^{n+1}) \equiv -\frac{S_b\Delta t\max_\alpha\big(s_\alpha^{n+1}-1\big)}{2\rho\big(\min_\alpha s_\alpha^n\big)^2}\left(|k|^3-\frac{k^4}{\sqrt{\big(\lambda\min_\alpha s_\alpha^n\big)^2+k^2}}\right)\widehat{\theta}^{n+1}, \tag{77}$$

Since the above small scale decomposition captures the leading order behavior of the solution operator, we can use it to obtain the stability constraint for the explicit scheme near the equilibrium. A simple calculation shows that the stability constraint for the explicit scheme is given by

$$\Delta t < C(S_b,\mu)h^\beta, \tag{78}$$

where $1\leqslant\beta\leqslant 3/2$. The value of β depends on μ. If $\mu\ll 1$, then we have $\beta\approx 3/2$. On the other hand, if $\mu\gg 1$, we have $\beta\approx 1$.

5. Summary of the efficient semi-implicit algorithm

To develop our efficient semi-implicit scheme, we will apply the Small Scale Decomposition that we developed in the previous section for the tangential an normal velocity fields to the unconditionally stable semi-implicit discretization intro-

duced in Section 3. We note that the leading contribution to the stiffness of the s_α equation comes from the derivative of the tangential velocity. The second term involving the normal velocity field is a lower order term. Thus it is sufficient to treat the leading order term of the derivative of the tangential velocity implicitly and treat the remaining terms explicitly. Once we have updated s_α, the leading order contribution to the stiffness of the θ equation comes from the derivative of the normal velocity. Therefore, we just need to treat the leading order contribution from the the derivative of the normal velocity implicitly when we discretize the θ equation. As we see from the previous section (see (76) and (77)), the leading order contributions from $\overline{U}_\alpha$ and V_α can be expressed as convolution operators. Thus we can invert the implicit solution explicitly by using the Fourier transform. Based on the above consideration, we propose the following semi-implicit scheme for the Immersed Boundary method:

Step 1: Discretization of the convection term.

$$\frac{\tilde{\mathbf{u}}^{n+1,1}-\mathbf{u}^n}{\Delta t}+\frac{1}{2}u_1^n D_{h,1}^0\tilde{\mathbf{u}}^{n+1,1}+\frac{1}{2}D_{h,1}^0\left(u_1^n\tilde{\mathbf{u}}^{n+1,1}\right)=0, \tag{79}$$

$$\frac{\tilde{\mathbf{u}}^{n+1}-\tilde{\mathbf{u}}^{n+1,1}}{\Delta t}+\frac{1}{2}u_2^n D_{h,2}^0\tilde{\mathbf{u}}^{n+1}+\frac{1}{2}D_{h,2}^0\left(u_2^n\tilde{\mathbf{u}}^{n+1}\right)=0. \tag{80}$$

Step 2: Update of $\mathbf{u}^{n+1}, p^{n+1}$ and s_α^{n+1}.

$$\frac{s_\alpha^{n+1}-s_\alpha^n}{\Delta t}=T(s_\alpha^{n+1})+\left(D_{\Delta\alpha}V^{*,n+1}-D_{\Delta\alpha}\theta^n U^{*,n+1}-T\left(s_\alpha^n\right)\right), \tag{81}$$

$$\rho\frac{\mathbf{u}^{n+1}-\tilde{\mathbf{u}}^{n+1}}{\Delta t}=-\nabla_h p^{n+1}+\mu\nabla_h^2\mathbf{u}^{n+1}+L_{h,n}\left(\mathbf{F}\left(s_\alpha^{n+1},\theta^n;\boldsymbol{\tau}^n,\mathbf{n}^n\right)\right), \tag{82}$$

$$\nabla_h^2 p^{n+1}=\frac{1}{\Delta t}\nabla_h\cdot\left(L_{h,n}\left(\mathbf{F}\left(s_\alpha^{n+1},\theta^n;\boldsymbol{\tau}^n,\mathbf{n}^n\right)\right)\Delta t+\rho\tilde{\mathbf{u}}^{n+1}\right), \tag{83}$$

where

$$\widehat{T}(s_\alpha^{n+1})=-\frac{S_b\Delta t}{2\rho(\min\limits_\alpha s_\alpha^n)^2}\left(\frac{(\lambda\min\limits_\alpha s_\alpha^n)^2k^2+k^4}{\sqrt{(\lambda\min\limits_\alpha s_\alpha^n)^2+k^2}}-|k|^3\right)\widehat{s}_\alpha^{n+1}, \tag{84}$$

$\lambda^2=\frac{\rho}{\mu\Delta t}$, and $\mathbf{u}^{*,n+1}$ is the intermediate velocity at t^{n+1} which is calculated explicitly using the following algorithm:

$$\rho\frac{\mathbf{u}^{*,n+1}-\tilde{\mathbf{u}}^{n+1}}{\Delta t}=-\nabla_h p^{*,n+1}+\mu\nabla_h^2\mathbf{u}^{*,n+1}+L_{h,n}(\mathbf{F}(s_\alpha^n,\theta^n;\boldsymbol{\tau}^n,\mathbf{n}^n)), \tag{85}$$

$$\nabla_h^2 p^{*,n+1}=\frac{1}{\Delta t}\nabla_h\cdot(L_{h,n}(\mathbf{F}(s_\alpha^n,\theta^n;\boldsymbol{\tau}^n,\mathbf{n}^n))\Delta t+\rho\tilde{\mathbf{u}}^{n+1}), \tag{86}$$

$$V^{*,n+1}=L_{h,n}^*(\mathbf{u}^{*,n+1})\cdot\boldsymbol{\tau}^n, \tag{87}$$

$$U^{*,n+1}=L_{h,n}^*(\mathbf{u}^{*,n+1})\cdot\mathbf{n}^n. \tag{88}$$

Step 3: Update of θ^{n+1}. Once we have updated $\mathbf{u}$, p, and s_α at t^{n+1}, we update θ^{n+1} using the following semi-implicit scheme:

$$\frac{\theta^{n+1}-\theta^n}{\Delta t}=\frac{S(\theta^{n+1})}{\min\limits_\alpha s_\alpha^{n+1}}+\left(\frac{1}{s_\alpha^{n+1}}(D_{\Delta\alpha}U^{n+1}+D_{\Delta\alpha}\theta^n V^{n+1})-\frac{S(\theta^n)}{\min\limits_\alpha s_\alpha^{n+1}}\right), \tag{89}$$

where

$$V^{n+1}=L_{h,n}^*(\mathbf{u}^{n+1})\cdot\boldsymbol{\tau}^n, \tag{90}$$

$$U^{n+1}=L_{h,n}^*(\mathbf{u}^{n+1})\cdot\mathbf{n}^n, \tag{91}$$

$$\widehat{S}(\theta^{n+1})=-\frac{S_b\Delta t\max\limits_\alpha(s_\alpha^n-1)}{2\rho(\min\limits_\alpha s_\alpha^n)^2}\left(|k|^3-\frac{k^4}{\sqrt{(\lambda\min\limits_\alpha s_\alpha^n)^2+k^2}}\right)\widehat{\theta}^{n+1}. \tag{92}$$

This is our semi-implicit scheme for the Immersed Boundary method for the Navier-Stokes equations. Since this scheme is derived by using Small Scale Decomposition, we call it the SSD semi-implicit scheme. In the semi-implicit scheme described above, we treat only the leading order term implicitly and discretize the lower order terms explicitly. As a result, the stability of the SSD semi-implicit scheme is weaker than the original unconditionally stable scheme and the SSD semi-implicit scheme may not have an associated non-increasing energy. A near equilibrium analysis shows that the stabil-

ity condition is $\Delta t < C(S_b, \mu)$. Since we treat the convection term implicitly, there is no CFL stability condition for the time step. Our numerical study also confirms this. Although the leading order term is discretized implicitly, we can solve for the implicit solution explicitly using the Fourier transform. Moreover, the linear system resulting from the discretization of the convection term in (79),(80) can be solved efficiently by two tridiagonal solves. Therefore the overall computational cost of the SSD semi-implicit scheme is comparable to that of an explicit method.

In our scheme, the reference point to reconstruct the interface $\mathbf{X}^{n+1}$ is computed explicitly. In order to reduce the stiffness introduced by the single reference point, we update two points $\mathbf{X}_1, \mathbf{X}_{N_b/2}$, then take average of them to determine the position of the interface at next time step.

Remark 3. The leading order term we derive above is calculated analytically using the space-continuous formulation with an unsmoothed Dirac delta function. As Stockie and Wetton pointed out in [34], this analysis over-predicts the stiffness of the Immersed Boundary method in a practical computation. If we use the leading order approximation directly, the semi-implicit scheme with the leading order terms derived above tends to over-dissipate the solution. To alleviate this effect in the practical implementation, we rescale the leading order term by a coefficient which is calculated at the first time step in the following way:

$$C_V = \frac{\max_\alpha V_\alpha^{1,*}}{\max_\alpha T(s_\alpha^0)},$$

$$C_U = \frac{\max_\alpha U^1}{\max_\alpha S_U(\theta^0)},$$

where $S_U(\theta^0)$ is the leading order term of $\overline{U}^1$, which can be computed from $S(\theta^0)$ via the Fourier transform. The leading order term we use in a practical computation is actually $C_V T(s_\alpha^{n+1})$ and $C_U S(\theta^{n+1})$.

Remark 4. We remark that if we exclude the source term from the above algorithm, we can get a unconditionally stable method for the incompressible Navier–Stokes in the sense of the total energy is non-increasing. The accuracy of this method would be first order in time and second order in space.

6. Viscous vortex sheet

In this section, we derive an efficient semi-implicit scheme for the viscous vortex sheet with surface tension. The motion of an interface, Γ, with surface tension, separating incompressible, viscous fluids can also be formulated by the immersed boundary method:

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u}\right) = -\nabla p + \mu\triangle\mathbf{u} + L(\mathbf{X})(\mathbf{F}(s_\alpha,\theta)), \tag{93}$$

$$\nabla\cdot\mathbf{u} = 0, \tag{94}$$

$$U = L^*(\mathbf{X})(\mathbf{u}(\mathbf{x}))\cdot\mathbf{n}, \tag{95}$$

$$V = L^*(\mathbf{X})(\mathbf{u}(\mathbf{x}))\cdot\boldsymbol{\tau}, \tag{96}$$

$$s_{\alpha t} = V_\alpha - \theta_\alpha U, \tag{97}$$

$$\theta_t = \frac{U_\alpha}{S_\alpha} + \frac{V\theta_\alpha}{S_\alpha}, \tag{98}$$

where

$$\mathbf{F}(s_\alpha,\theta) = \frac{\partial}{\partial\alpha}(T\boldsymbol{\tau}) = T\theta_\alpha\mathbf{n}, \tag{99}$$

and T is the surface tension coefficient. The far field boundary condition is

$$\mathbf{u}(x,y) \to (\pm V_0, 0) \text{ as } y \to \pm\infty. \tag{100}$$

Note that in this case the force density depends only on the curvature and the configuration of the boundary. If we change the integration variable from the Lagrangian variable α to the arclength variable s, and use the relationship, $\theta_\alpha = \kappa s_\alpha$, we obtain the following expression for the force density:

$$\mathbf{f}(\mathbf{x},t) = \int_0^{L(t)} \kappa(s,t)\delta(\mathbf{x}-\mathbf{X}(s,t))\mathrm{d}s, \tag{101}$$

where $L(t)$ is the total arclength of the interface at time t. Since the force is independent of s_α, the leading order term of the tangential velocity becomes zero. In fact, the tangential velocity does not influence the evolution of the interface. The choice of the tangential velocity only affects the parameterization of the interface. We can use any tangential velocity, V, to evolve the interface. In particular, we can choose V such that s_α is independent of α at each time step. In this case, the arclength

derivative, s_α, can be replaced by the total arclength of the interface, $L(t)$. This is the so-called $\theta - L$ formulation. This leads to the following choice of V [12]:

$$V(\alpha,t) = V(0,t) + \int_0^\alpha \theta'_\alpha U \,\mathrm{d}\alpha' - \alpha \int_0^1 \theta'_\alpha U \,\mathrm{d}\alpha'. \tag{102}$$

For simplicity, we can simply set $V(0,t)$ to be 0.

If we use the tangential velocity above, then s_α is independent of α, and we have

$$s_\alpha(\alpha,t) = L(t) = \int_0^1 s_{\alpha'}(\alpha',t)\mathrm{d}\alpha'. \tag{103}$$

The evolution of the interface is now given in terms of L and θ

$$L_t = -\int_0^1 \theta'_\alpha U \,\mathrm{d}\alpha', \tag{104}$$

$$\theta_t = \frac{1}{L}(U_\alpha + \theta_\alpha V). \tag{105}$$

We remark that Ceniceros and Roma have also used the arclength and tangent angle formulation to alleviate the stiffness of the viscous vortex sheet with surface tension in [6].

Next, we will derive our semi-implicit scheme for the viscous vortex sheet problem based on the $\theta - L$ formulation. In order to resolve the configuration of the sheet, we develop a second order semi-implicit scheme. To simplify the presentation, we will only describe the semi-discrete algorithm. We use the second order ENO scheme to discretize the convection term [5]. The numerical scheme consists of two steps. In the first step, we update the solution at an intermediate time step $t^{n+\frac{1}{2}}$:

$$\frac{L^{n+\frac{1}{2}} - L^n}{\Delta t/2} = -\int_0^1 \theta^n_{\alpha'} U^n \,\mathrm{d}\alpha', \tag{106}$$

$$\rho\left(\frac{\mathbf{u}^{n+\frac{1}{2}} - \mathbf{u}^n}{\Delta t/2} + \mathbf{u}^n \cdot \nabla \mathbf{u}^n\right) = -\nabla p^{n+\frac{1}{2}} + \mu \nabla^2 \mathbf{u}^{n+\frac{1}{2}} + L_n(\mathbf{F}(\theta^n;\mathbf{n}^n)), \tag{107}$$

$$\nabla^2 p^{n+\frac{1}{2}} = \nabla \cdot (L_n(\mathbf{F}(\theta^n;\mathbf{n}^n)) - \rho \mathbf{u}^n \cdot \nabla \mathbf{u}^n), \tag{108}$$

$$U^{n+\frac{1}{2}} = L_n^*(\mathbf{u}^{n+\frac{1}{2}}) \cdot \mathbf{n}^n, \tag{109}$$

$$V^{n+\frac{1}{2}} = \int_0^\alpha \theta^n_{\alpha'} U^{n+\frac{1}{2}} \mathrm{d}\alpha' - \alpha \int_0^1 \theta^n_{\alpha'} U^{n+\frac{1}{2}} \mathrm{d}\alpha', \tag{110}$$

$$\frac{\theta^{n+\frac{1}{2}} - \theta^n}{\Delta t/2} = \frac{S_p(\theta^{n+\frac{1}{2}})}{L^{n+\frac{1}{2}}} + \left(\frac{1}{L^{n+\frac{1}{2}}}\left(U_\alpha^{n+\frac{1}{2}} + \theta^n_\alpha V^{n+\frac{1}{2}}\right) - \frac{S_p(\theta^n)}{L^{n+\frac{1}{2}}}\right). \tag{111}$$

where $\mathbf{n}^n = (-\sin\theta^n, \cos\theta^n)$ and

$$\mathbf{F}(\theta^n;\mathbf{n}^n) = T\ \theta^n_\alpha\ \mathbf{n}^n. \tag{112}$$

In the second step, we update the solution at t^{n+1}:

$$\frac{L^{n+1} - L^n}{\Delta t} = -\int_0^1 \theta^{n+\frac{1}{2}}_{\alpha'} U^{n+\frac{1}{2}} \mathrm{d}\alpha' \tag{113}$$

$$\rho\left(\frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t} + \mathbf{u}^{n+\frac{1}{2}} \cdot \nabla \mathbf{u}^{n+\frac{1}{2}}\right) = -\nabla \bar{p} + \mu \nabla^2 \bar{\mathbf{u}} + L_{n+\frac{1}{2}}\left(\mathbf{F}\left(\theta^{n+\frac{1}{2}};\mathbf{n}^{n+\frac{1}{2}}\right)\right) \tag{114}$$

$$\nabla^2 \bar{p} = \nabla \cdot \left(L_{n+\frac{1}{2}}\left(\mathbf{F}\left(\theta^{n+\frac{1}{2}};\mathbf{n}^{n+\frac{1}{2}}\right)\right) - \rho \mathbf{u}^{n+\frac{1}{2}} \cdot \nabla \mathbf{u}^{n+\frac{1}{2}}\right) \tag{115}$$

$$\overline{U} = L^*_{n+\frac{1}{2}}(\bar{\mathbf{u}}) \cdot \mathbf{n}^{n+\frac{1}{2}} \tag{116}$$

$$\overline{V} = \int_0^\alpha \theta^{n+\frac{1}{2}}_{\alpha'} \overline{U} \,\mathrm{d}\alpha' - \alpha \int_0^1 \theta^{n+\frac{1}{2}}_{\alpha'} \overline{U} \,\mathrm{d}\alpha' \tag{117}$$

$$\frac{\theta^{n+1} - \theta^n}{\Delta t} = \frac{S_m(\overline{\theta})}{2\overline{L}} + \left(\frac{1}{\overline{L}}\left(\overline{U}_\alpha + \overline{\theta}_\alpha \overline{V}\right) - \frac{S_m(\theta^{n+\frac{1}{2}})}{2\overline{L}}\right). \tag{118}$$

where $\mathbf{n}^{n+\frac{1}{2}} = \left(-\sin\left(\theta^{n+\frac{1}{2}}\right), \cos\left(\theta^{n+\frac{1}{2}}\right)\right)$,

$$\overline{L} = \frac{L^{n+1} + L^n}{2}, \quad \bar{\mathbf{u}} = \frac{\mathbf{u}^{n+1} + \mathbf{u}^n}{2}, \quad \overline{\theta} = \frac{\theta^{n+1} + \theta^n}{2}$$

and

$$\mathbf{F}(\theta^{n+\frac{1}{2}};\mathbf{n}^{n+\frac{1}{2}}) = T\ \theta^{n+\frac{1}{2}}_\alpha\ \mathbf{n}^{n+\frac{1}{2}}.$$

Now, the leading order terms, $\widehat{S}_p$ and $\widehat{S}_m$, become simpler. They are defined below:

$$\widehat{S}_p\left(\theta^{n+\frac{1}{2}}\right) = -\frac{T\Delta t}{4\rho(L^n)^2}\left(|k|^3 - \frac{k^4}{\sqrt{(\lambda L^n)^2 + k^2}}\right)\widehat{\theta}^{n+\frac{1}{2}}, \tag{119}$$

$$\widehat{S}_m(\bar{\theta}) = -\frac{T\Delta t}{2\rho\bar{L}^2}\left(|k|^3 - \frac{k^4}{\sqrt{(\lambda\bar{L})^2 + k^2}}\right)\widehat{\bar{\theta}}, \tag{120}$$

where $\lambda^2 = \frac{2\rho}{\mu\Delta t}$.

The configuration of the interface, $\mathbf{X}(\alpha, t)$, can be obtained by integrating the formula

$$x_\alpha = L\cos\theta, \quad y_\alpha = L\sin\theta. \tag{121}$$

Unfortunately, integrating (121) directly can not guarantee that the interface is periodic. The loss of periodicity of the interface can have a serious consequence on the accuracy and stability of the numerical computation. To overcome this difficulty, we force the interface to be periodic by integrating the following equivalent formula:

$$x(\alpha, t) = x(0, t) + \alpha\left(1 - L\int_0^1 \cos(\theta(\alpha'))\mathrm{d}\alpha'\right) + L\int_0^\alpha \cos(\theta(\alpha'))\mathrm{d}\alpha', \tag{122}$$

$$y(\alpha, t) = y(0, t) - \alpha L\int_0^1 \sin(\theta(\alpha'))\mathrm{d}\alpha' + L\int_0^\alpha \sin(\theta(\alpha'))\mathrm{d}\alpha'. \tag{123}$$

From the periodicity of the problem, the velocity at $x(0, t)$, $y(0, t)$ is always zero, so we fix $x(0, t) = x(0, 0) = 0$, $y(0, t) = y(0, 0) = 0$ in our computations.

7. Numerical results

In this section, we perform a number of numerical experiments to study the stability property of our semi-implicit scheme for the immersed boundary problem. To illustrate the stability property of our semi-implicit scheme, we apply our method to a prototype test problem and test our scheme for a wide range of elastic coefficients and viscosity coefficients. We also compare the performance of our method with that of an explicit scheme and the unconditionally stable scheme. Our computational experiments confirm that our semi-implicit scheme removes the high frequency stability constraint induced by the elastic force. The computational saving over an explicit scheme is very substantial. The computational gain is even bigger as the stiffness of the Immersed Boundary method becomes more severe. We also apply our semi-implicit scheme to compute the vortex sheet problem in a viscous fluid with Reynolds number of order 10,000, and observe some interesting phenomena.

7.1. The immersed boundary method for 2D Navier–Stokes equations

We first describe the set-up of our numerical test problem. The test problem we use is one typically seen in the literature, in which the immersed boundary is a closed curve initially in the shape of an ellipse. We choose an ellipse initially aligned in the coordinate directions with horizontal semi-axis $a = 0.32$ and vertical semi-axis $b = 0.24$:

$$\begin{cases} x(\alpha, 0) = 0.5 + 0.32\cos\alpha, \\ y(\alpha, 0) = 0.5 + 0.24\sin\alpha. \end{cases} \tag{124}$$

The rest state of the boundary is a circle with radius $r = 0.2$. The fluid is initially at rest in a doubly periodic domain $\Omega = [0, 1] \times [0, 1]$. The boundary conditions are periodic in both directions. For this test problem, the boundary oscillates around a circular equilibrium state with the same area as that of the original ellipse.

We discretize Ω using a uniform $N \times N$ grid. We set the mesh size of the immersed boundary to be $N_b = 2N$, so that there are approximately 2 immersed boundary points per mesh width. We use the spectral method to discretize the diffusion term and the spatial derivatives in the domain Ω and along the immersed boundary. The leading order singular integral is also discretized by the spectral method. On the other hand, the convection term is discretized by using the center difference approximation.

We focus on the tests with large elastic coefficient S_b and relatively small viscous coefficient μ. This is also the most challenging case in practical computations. In our numerical study, we use a wide range parameter values:

$$\rho = 1, \quad S_b = 10^3, 10^4, 10^5, \quad \mu = 1, 0.1, 0.01.$$

We also use different spatial resolutions with

$$N = 128,\ 256,\ 512,$$

to study the stability of our SSD semi-implicit scheme and to compare its performance with an explicit method.

It is useful to rewrite the model in the nondimensional form. To this end, we define the following dimensionless variables:

$$t' = \frac{t}{t_0}, \quad \mathbf{x}' = \frac{\mathbf{x}}{L}, \quad \mathbf{u}' = \frac{\mathbf{u}t_0}{L}, \quad p' = \frac{pt_0}{\mu}, \quad \mathbf{f}' = \frac{\mathbf{f}Lt_0}{\mu},$$

where L is the size of computational domain, t_0 is characteristic time. Using these new variables, we have

$$\frac{\partial \mathbf{u}'}{\partial t'} + \mathbf{u}' \cdot \nabla \mathbf{u}' = \frac{\mu t_0}{\rho L^2}(-\nabla p' + \triangle \mathbf{u}' + \mathbf{f}'(\mathbf{x}', t')), \tag{125}$$

$$0 = \nabla \cdot \mathbf{u}'. \tag{126}$$

For the equations of the elastic boundary, the dimensionless variables are

$$\mathbf{X}' = \frac{\mathbf{X}}{L}, \quad s'_\alpha = \frac{s_\alpha}{L}, \quad \theta' = \theta, \quad \alpha' = \frac{\alpha}{L}, \quad T' = \frac{T}{S_b}, \quad \mathbf{F}' = \frac{\mathbf{F}L}{S_b}, \quad \boldsymbol{\tau}' = \boldsymbol{\tau}, \quad \mathbf{n}' = \mathbf{n}.$$

Then the equations describe the interaction of the boundary and the fluid become

$$U' = \mathbf{u}'(\mathbf{X}'(\alpha', t'), t') \cdot \mathbf{n}', \tag{127}$$

$$V' = \mathbf{u}'(\mathbf{X}'(\alpha', t'), t') \cdot \boldsymbol{\tau}', \tag{128}$$

$$s'_{\alpha,t'} = V'_{\alpha'} - \theta'_{\alpha'} U', \tag{129}$$

$$\theta'_{t'} = \frac{1}{s'_\alpha}(U'_{\alpha'} + V'\theta'_{\alpha'}), \tag{130}$$

where

$$\mathbf{f}'(\mathbf{x}', t') = \frac{S_b t_0}{\mu L} \int_0^{L_b/L} \mathbf{F}'(\alpha', t')\delta(\mathbf{x}' - \mathbf{X}'(\alpha', t'))\mathrm{d}\alpha', \tag{131}$$

$$\mathbf{u}'(\mathbf{X}'(\alpha', t'), t') = \int_\Omega \mathbf{u}'(\mathbf{x}', t')\delta(\mathbf{x}' - \mathbf{X}'(\alpha', t'))\mathrm{d}\mathbf{x}'. \tag{132}$$

From the nondimensional analysis, we can see that there are three nondimensional parameters in this problem:

$$\frac{S_b t_0}{\mu L}, \quad \frac{\mu t_0}{\rho L^2}, \quad \frac{L_b}{L}.$$

Let $U_0 = \frac{L}{t_0}$ be the characteristic velocity. In our test problem, L_b and L are fixed and depend on the initial condition only. Then there are only two parameters left:

$$\frac{S_b}{\mu U_0}, \quad \frac{\rho L U_0}{\mu}.$$

We note that the second parameter is the Reynolds number.

If we choose the characteristic velocity to be the maximum velocity, the range of these two parameters in our numerical tests is,

$$\frac{\rho L U_0}{\mu} : 10 \sim 6 \times 10^4,$$

$$\frac{S_b}{\mu U_0} : 10^2 \sim 1.6 \times 10^4.$$

In the case of $S_b = 10^5$ and $\mu = 0.01$, the Reynolds number $\frac{\rho L U_0}{\mu}$ is equal to 6×10^4 and the nondimensionalized elastic coefficient $\frac{S_b}{\mu U_0}$ is equal to 1.6×10^4.

7.2. Accuracy of our semi-implicit scheme

In this subsection, we perform a convergence study for our semi-implicit scheme. First, we study the convergence rate in time. We fix $N = 256$ and vary the time steps in powers of 2 from $\frac{1}{16}$ to $\frac{1}{128}$. The elastic coefficient S_b is fixed to be 1. We consider a sequence of viscosity coefficients: $\mu = 0.1, 0.01, 0.005$. Following [25], we compute the time discretization error at time T as follows:

$$e_T(v; \Delta t) = \|v(T; \Delta t) - v(T; \Delta t/2)\|_{l^2}. \tag{133}$$

For a vector field $\mathbf{u}(\mathbf{x}) = (u_1(\mathbf{x}), u_2(\mathbf{x}))$ defined on the Cartesian grid with $x_i = ih, y_j = jh$, the discrete l^2 norm is defined as follows

$$\|\mathbf{u}\|_{l^2} = \left(\sum_{i,j} (u_1^2(x_i, y_j) + u_2^2(x_i, y_j))h^2\right)^{\frac{1}{2}}. \tag{134}$$

Similarly, the discrete l^2 norm for a vector field $\mathbf{w}(\alpha) = (w_1(\alpha), w_2(\alpha))$ defined on the interface Γ is defined below:

$$\|\mathbf{w}\|_{l^2} = \left(\sum_i \left(w_1^2(\alpha_i) + w_2^2(\alpha_i) \right) \Delta\alpha \right)^{\frac{1}{2}}. \tag{135}$$

We compute the solution up to $T = 1$ and evaluate the convergence rate based on the numerical solution at $T = 1$ with different temporal resolutions. The results are shown in Table 1. As we can see, the convergence rate is better than first order but it does not achieve second order.

Now, we study the convergence rate in space. The time step is fixed to be $\frac{1}{512}$ and the meshsizes are varied in powers of 2 from 32 to 256. The elastic coefficient S_b is fixed to be 1 and we consider a sequence of viscosity coefficients: $\mu = 0.1, 0.01, 0.005$. The numerical error is computed in the same way as before:

$$e_T(u; h) = \|u(T; h) - u(T; h/2)\|_{l^2}. \tag{136}$$

The solution is computed up to $T = 1$. The results are shown in Table 2. Again, we observe that the convergence rate is better than first order. As the viscosity coefficient decreases, the velocity field becomes more and more singular, and the convergence rate also decreases.

7.3. Stability property of our semi-implicit scheme

In this subsection, we will perform some extensive numerical studies to investigate the stability property of our semi-implicit scheme and compare its performance with an explicit scheme and the unconditionally stable semi-implicit scheme. In the explicit scheme that we use, we discretize the convection term using the upwinding scheme and update the elastic boundary explicitly, but the diffusion term is discretized implicitly. This is similar to the Forward Euler/Backward Euler method used by Stockie and Wetton in [34].

In Fig. 1 we plot the total energy as a function of time for the explicit scheme and our SSD semi-implicit scheme with different time steps 2×10^{-6} and 2×10^{-5}. When $\Delta t = 2 \times 10^{-6}$, both the explicit and SSD semi-implicit schemes are stable. When $\Delta t = 2 \times 10^{-5}$, the explicit scheme becomes unstable, but our SSD semi-implicit scheme is still stable.

To further investigate the stability property of our SSD semi-implicit scheme, we compute the maximum time steps with different meshsizes $N = 128,\ 256,\ 512$. The total time we run is $t = 0.01$. For each method we run at least 100 steps. The results are shown in Table 3. As we can see from Table 3, the maximum time steps that we can use for our semi-implicit scheme is much larger than that for the explicit scheme. Moreover, the larger the elastic coefficient, or the Reynolds number, or the numerical resolution is, the larger the ratio between the maximum time step of our semi-implicit scheme and that of the explicit scheme becomes. In the most severe case we have tested with $S_b = 10^5, N = 512$ and $\mu = 0.01$, the maximum time step of our semi-implicit scheme is 1160 times larger than that of the explicit scheme. Even after we take into account the extra cost in inverting the semi-implicit solution, our semi-implicit scheme still runs 683 times faster than the explicit scheme.

It is interesting to compare the performance of our semi-implicit scheme for the Navier–Stokes equations with that for the Stokes equations. In [15], we proposed an efficient semi-implicit scheme for the Stokes equations, and showed that our

Table 1
Numerical error of **X** and **u** with different timestep

		$\Delta t = 1/16$	$\Delta t = 1/32$	$\Delta t = 1/64$	Convergence rate
X	$\mu = 0.1$	3.32×10^{-4}	9.73×10^{-5}	2.85×10^{-5}	1.77
	$\mu = 0.01$	2.42×10^{-3}	7.81×10^{-4}	2.82×10^{-4}	1.55
	$\mu = 0.005$	2.57×10^{-3}	1.06×10^{-3}	3.69×10^{-4}	1.40
u	$\mu = 0.1$	7.59×10^{-4}	3.72×10^{-4}	1.66×10^{-4}	1.09
	$\mu = 0.01$	2.22×10^{-3}	1.06×10^{-3}	4.62×10^{-4}	1.13
	$\mu = 0.005$	4.27×10^{-3}	2.05×10^{-3}	9.18×10^{-4}	1.11

Table 2
Numerical error of **X** and **u** with different spatial meshsize

		$h = 1/32$	$h = 1/64$	$h = 1/128$	Convergence rate
X	$\mu = 0.1$	1.52×10^{-4}	3.57×10^{-5}	9.31×10^{-6}	2.01
	$\mu = 0.01$	3.92×10^{-3}	1.23×10^{-3}	3.71×10^{-4}	1.70
	$\mu = 0.005$	6.33×10^{-3}	2.19×10^{-3}	6.78×10^{-4}	1.61
u	$\mu = 0.1$	3.01×10^{-4}	1.06×10^{-5}	4.03×10^{-5}	1.45
	$\mu = 0.01$	2.86×10^{-3}	7.25×10^{-4}	3.86×10^{-4}	1.44
	$\mu = 0.005$	7.77×10^{-3}	2.06×10^{-3}	1.06×10^{-3}	1.43

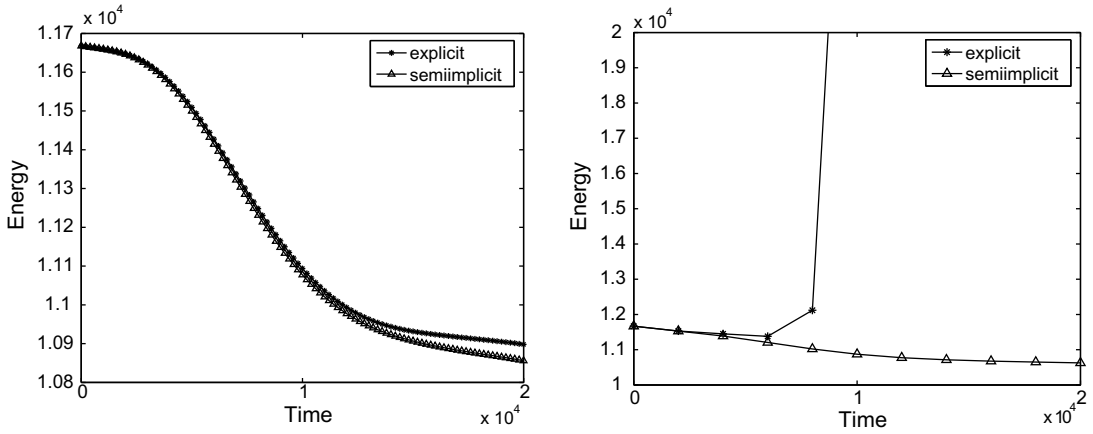

Fig. 1. Energy of the Navier–Stokes system for semi-implicit and explicit methods with different time steps, using the implicit scheme (32) and (33) to discretize the convection term. Left: $\Delta t = 2 \times 10^{-6}$; Right: $\Delta t = 2 \times 10^{-5}$. $S_b = 10^5$, $\mu = 0.01$.

Table 3
Maximum time steps for our SSD semi-implicit scheme for the Navier–Stokes equations with different meshsizes

		$S_b = 10^3$		$S_b = 10^4$		$S_b = 10^5$	
		SSD s,i	exp	SSD s,i	exp	SSD s,i	exp
$\mu = 1$	$N = 128$	1.21×10^{-1}	3.29×10^{-4}	2.26×10^{-2}	6.33×10^{-5}	1.68×10^{-3}	1.62×10^{-5}
	$N = 256$	1.19×10^{-1}	1.49×10^{-4}	2.26×10^{-2}	2.53×10^{-5}	1.68×10^{-3}	5.72×10^{-6}
	$N = 512$	1.19×10^{-1}	6.73×10^{-5}	2.26×10^{-2}	1.01×10^{-5}	1.68×10^{-3}	2.19×10^{-6}
$\mu = 0.1$	$N = 128$	1.67×10^{-2}	1.61×10^{-4}	4.05×10^{-3}	4.57×10^{-5}	1.12×10^{-3}	1.14×10^{-5}
	$N = 256$	1.67×10^{-2}	5.95×10^{-5}	4.05×10^{-3}	1.64×10^{-5}	1.09×10^{-3}	4.79×10^{-6}
	$N = 512$	1.67×10^{-2}	2.21×10^{-5}	4.05×10^{-3}	5.72×10^{-6}	1.09×10^{-3}	1.63×10^{-6}
$\mu = 0.01$	$N = 128$	1.13×10^{-2}	1.17×10^{-4}	3.38×10^{-3}	1.92×10^{-5}	1.01×10^{-3}	3.48×10^{-6}
	$N = 256$	1.11×10^{-2}	4.57×10^{-5}	3.13×10^{-3}	1.07×10^{-5}	8.86×10^{-4}	1.69×10^{-6}
	$N = 512$	1.09×10^{-2}	1.62×10^{-5}	2.92×10^{-3}	3.50×10^{-6}	8.02×10^{-4}	6.91×10^{-7}

The legend "exp" stands for the explicit scheme, "SSD s,i" the SSD semi-implicit scheme.

Table 4
Maximum time steps for our SSD semi-implicit method for the unsteady Stokes equations with different meshsizes

		$S_b = 10^3$		$S_b = 10^4$		$S_b = 10^5$	
		SSD s,i	exp	SSD s,i	exp	SSD s,i	exp
$\mu = 1$	$N = 128$	1.22×10^{-1}	3.29×10^{-4}	2.26×10^{-2}	6.33×10^{-5}	3.73×10^{-3}	1.61×10^{-5}
	$N = 256$	1.19×10^{-1}	1.48×10^{-4}	2.26×10^{-2}	2.52×10^{-5}	3.25×10^{-3}	5.72×10^{-6}
	$N = 512$	1.19×10^{-1}	6.72×10^{-5}	2.26×10^{-2}	1.01×10^{-5}	3.13×10^{-3}	2.19×10^{-6}
$\mu = 0.1$	$N = 128$	3.75×10^{-2}	1.61×10^{-4}	4.49×10^{-3}	4.42×10^{-5}	8.94×10^{-4}	1.14×10^{-5}
	$N = 256$	3.24×10^{-2}	5.95×10^{-5}	4.39×10^{-3}	1.58×10^{-5}	8.89×10^{-4}	4.20×10^{-6}
	$N = 512$	3.13×10^{-2}	2.21×10^{-5}	4.34×10^{-3}	5.53×10^{-6}	8.89×10^{-4}	1.38×10^{-6}
$\mu = 0.01$	$N = 128$	8.78×10^{-3}	1.19×10^{-4}	2.34×10^{-3}	2.02×10^{-5}	6.22×10^{-4}	3.84×10^{-6}
	$N = 256$	8.72×10^{-3}	4.11×10^{-5}	2.29×10^{-3}	1.01×10^{-5}	6.12×10^{-4}	1.59×10^{-6}
	$N = 512$	8.68×10^{-3}	1.39×10^{-5}	2.26×10^{-3}	2.68×10^{-6}	5.61×10^{-4}	4.94×10^{-7}

The legend "exp" stands for the explicit scheme, "SSD s,i" the SSD semi-implicit scheme.

semi-implicit scheme gave a substantial improvement over an explicit scheme or a fully implicit scheme. One may be interesting to see how the inclusion of the convection term may affect the performance of our semi-implicit scheme. In Table 4, we list the maximum time steps obtained by using our semi-implicit scheme for the Stokes equations and compare them with those obtained using the explicit scheme. The comparison is conducted using the same test problem with the same range of parameters. By comparing Table 3 with Table 4, we can see that the ratio between the maximum time steps of our semi-implicit scheme and those of the explicit scheme for the Navier–Stokes equations is comparable to that for the unsteady Stokes equations. This is very encouraging. It shows that our semi-implicit scheme is as effective for the Navier–Stokes equations as it is for the unsteady Stokes equations.

The results we present here confirm that our SSD semi-implicit scheme can eliminate the stability constraint from both the convection term and the elastic force. The stability of our SSD semi-implicit scheme is much better than that of the explicit scheme. We note that the maximum time steps for both the Navier–Stokes equations and the unsteady Stokes equations are essentially independent of the meshsize. On the other hand, we also observe that the maximum time steps for our SSD semi-implicit scheme still have some mild dependence on the elastic coefficient and the Reynolds number. This is because the small scale decomposition does not remove the stiffness of the immersed boundary problem in the low to intermediate frequencies. The large elastic coefficient S_b is multiplied to the solution in all frequencies. Thus the stiffness also contributes to the low to intermediate frequencies of the solution. This explains why we can not completely remove the stiffness of the elastic force in the immersed boundary method by using the small scale decomposition. Nonetheless, given the simplicity and the efficiency of our semi-implicit scheme, the computational gain we obtain over the explicit scheme is already very substantial.

7.4. Performance

Next, we compare the computational cost of our SSD semi-implicit scheme with that of the unconditionally stable semi-implicit scheme and the explicit scheme. Specifically, we compare the CPU time that is required for each of these schemes to obtain an accurate numerical solution at a given time. Since there is extra computational cost in implementing our semi-implicit scheme, it is important to compare how much saving we actually obtain when performing a large scale computational with realistic physical parameters. We document the CPU times required for each scheme if we compute the solution up to $t = 0.05$ in the most severe case of $S_b = 10^5$ and $\mu = 0.01$. We use $\Delta t = 5 \times 10^{-4}$ for our semi-implicit scheme and the unconditionally stable semi-implicit scheme. This choice of $\Delta t = 5 \times 10^{-4}$ is due to the accuracy consideration, not that of stability. We choose this time step to ensure that the interface is resolved with a reasonable accuracy. Note that the corresponding time step required by the explicit scheme is much smaller. In the case of $N = 512$, the maximum time step that is required by the time step stability constraint is 6.91×10^{-7}.

In Table 5, we show the CPU times in seconds for these three schemes. The computational results presented here are performed by using a Matlab code in a Dell OPTIPLEX GX620 computer (64-bit-capable 3.6 GHz, Pentium 4, 660 CPU, 512 MB of RAM). We observe that the computational cost of the unconditionally stable semi-implicit scheme is roughly of the same order as that of the explicit scheme for a modest resolution. The computational gain over the explicit scheme is very limited. On the other hand, our SSD semi-implicit scheme runs 427 times faster than the explicit scheme in the case of $N = 512$. As the elastic coefficient or the Reynolds number or the resolution increases, the gain over the explicit scheme is even more substantial.

Recall that the convection term is discretized by using a variant of the semi-implicit ADI scheme, (32) and (33). To obtain the intermediate velocity field $\tilde{\mathbf{u}}$, we need to invert a $N \times N$ linear system $4N$ times in each time step. Fortunately, the matrix for this linear system is a cycled tridiagonal matrix, which can be solved with a linear complexity. Thus, the complexity for solving $\tilde{\mathbf{u}}$ is essentially the same as that for discretizing the convection term by the upwinding scheme. This shows that the computational cost of our SSD semi-implicit scheme per time step is comparable to that of the explicit scheme.

As we mentioned before, we can also discretize the convection term by using an explicit scheme, such as the upwinding scheme. If we do so, we obtain a new semi-implicit scheme by replacing the discretization (32) and (33) by the upwinding scheme. We call this modified scheme the upwind semi-implicit scheme. The upwind semi-implicit scheme needs to satisfy

Table 5
The CPU time in seconds for semi-implicit and explicit method

	Explicit	SSD s,i	Stable s,i
$N = 128$	660	13	660
$N = 256$	4970	44	3330
$N = 512$	96961	227	18300

The total time is 0.05, $S_b = 10^5$, $\mu = 0.01$. The legend "SSD s,i" stands for the SSD semi-implicit scheme, "stable s,i" the unconditionally stable semi-implicit scheme. We use the maximum time step listed in Table 3 for the explicit scheme. For the SSD semi-implicit scheme and unconditionally stable semi-implicit scheme, we use $\Delta t = 5 \times 10^{-4}$.

Table 6
The CPU time in seconds for three semi-implicit schemes with different treatments of the convection term

	SSD s,i	Upwind s,i (uniform)	Upwind s,i (adaptive)
$N = 128$	13	39	23
$N = 256$	44	474	194
$N = 512$	227	9912	2672

The total time is 0.05, $S_b = 10^5$, $\mu = 0.01$. The legend "SSD s,i" stands for the SSD semi-implicit scheme, "upwind s,i (uniform)" the semi-implicit scheme with an upwinding discretization of the convection term and a uniform time stepping, "upwind s,i (adaptive)" the semi-implicit scheme with an upwinding discretization of the convection term and an adaptive time stepping. For the SSD semi-implicit scheme, we use $\Delta t = 5 \times 10^{-4}$.

the CFL condition to maintain stability. We have performed numerical experiments to compare the performance of the upwind semi-implicit scheme with that of the SSD semi-implicit scheme introduced in Section 5. We have used both a uniform time step and an adaptive time step to satisfy the CFL condition. The results are shown in Table 6. In the case of $N = 512$ using the adaptive time stepping, we need to run 1226 steps to solve the solution at $t = 0.05$ for the upwind semi-implicit scheme. The CPU time is 11 times of that of our SSD semi-implicit scheme. If we use a uniform time step, the number of time steps would increase to 4386. The CPU time becomes 9912 seconds, which is about 44 times of that of our SSD semi-implicit scheme. Therefore the additional saving by using the ADI semi-implicit scheme to the convection term is quite significant.

7.5. Area conservation

It is known that the immersed boundary method does not conserve the area enclosed by the immersed boundary although the velocity field on the Eulerian grid satisfies a discrete divergence free condition. Our semi-implicit method does not fix this problem. However, we find if the velocity field is updated by a spectral method instead of a finite difference method, the area is conserved much better. This observation is illustrated in Fig. 2. For the same explicit method, if we use the spectral method to solve for the velocity, the area is almost conservative with only 0.07% area loss. On the other hand, if we use a finite difference method to solve for the velocity field, then the area loss is as large as 23%. It has been observed that the larger the time step is, the more severe the area loss becomes [26]. Note that our SSD semi-implicit scheme uses a much larger time step than the explicit scheme (by a factor as large as 1000), the use of our semi-implicit scheme may result in a greater area loss than an explicit scheme. Fortunately, since we use the spectral method to solve the velocity field, our SSD semi-implicit scheme still gives a much smaller area loss than the explicit scheme using a finite difference method to solve the velocity field.

7.6. The immersed boundary method for the viscous vortex sheet

In this section, we apply the semi-implicit scheme (106)–(120) to compute the viscous vortex sheet problem with surface tension. The interface is nearly flat initially

$$x(\alpha, 0) = \alpha, \quad y(\alpha, 0) = 0.05 \sin 2\pi\alpha, \quad \gamma(\alpha, 0) = 2, \tag{137}$$

where γ is the vortex sheet strength.

The computational domain is a rectangular domain, $[0, 1] \times [-1, 1]$. The horizontal boundaries are periodic and the top and bottom boundaries are rigid, moving walls. The velocity fields at the top and bottom boundaries are $(1,0)$ and $(-1, 0)$, respectively. This generates a shear to the fluid flow. The density and viscosity of the fluids above and below the interface are same, $\rho = 1, \mu = 2 \times 10^{-4}$. The Reynolds number is equal to 10,000, which is very large. The mesh size we use is 1024×2048. The Navier–Stokes equations are solved by the projection method [2].

A "frozen coefficient" analysis reveals that an explicit method needs to satisfy the following time step stability constraint:

$$\Delta t < C\ We^{1/2}\ (\bar{s}_\alpha h)^{3/2}, \quad \text{where } \bar{s}_\alpha = \min_\alpha s_\alpha, \tag{138}$$

where We is the Weber number [12,13], $h = 1/N_b$ is the grid spacing, N_b is the number of grid points along the interface Γ. Since the arclength spacing, $\Delta s \approx s_\alpha h$, Eq. (138) implies that the stability constraint is in fact determined by the minimum spacing in the arclength between two adjacent points on the grid.

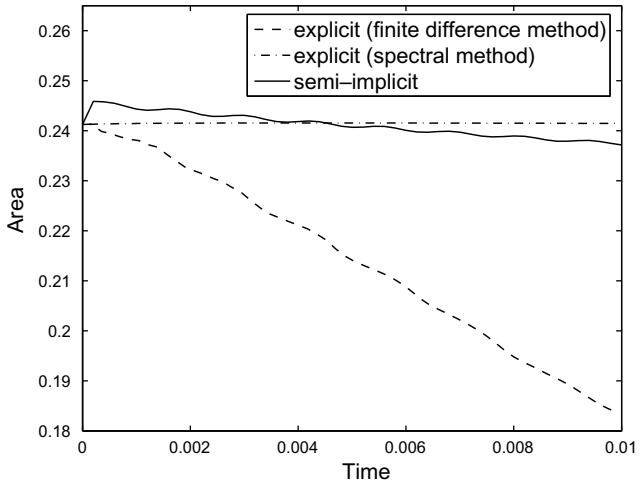

Fig. 2. Area for our SSD semi-implicit and explicit schemes using a finite difference or a spectral method. For the explicit scheme, we use $\Delta t = 3 \times 10^{-6}$. For the SSD semi-implicit scheme, we use $\Delta t = 1 \times 10^{-4}$. $S_b = 10^5$, $\mu = 0.01$.

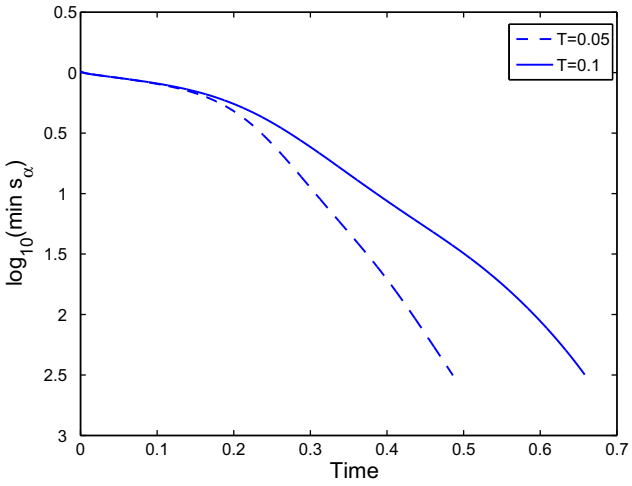

Fig. 3. The evolution of $\log_{10}(\bar{s}_\alpha)$ for different values of surface tension.

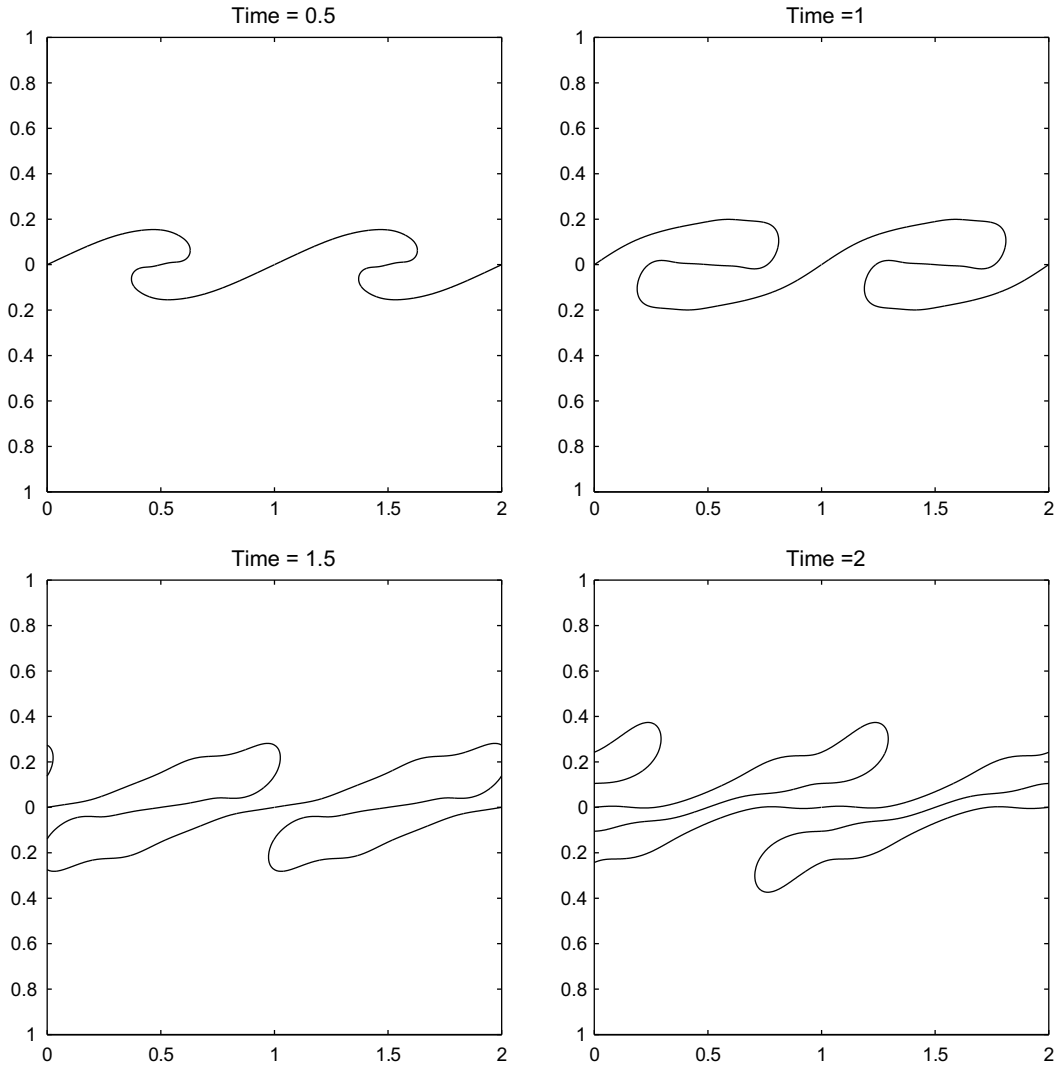

Fig. 4. Configurations of the interface at different times, 0.5 s, 1 s, 1.5 s, 2 s, surface tension coefficient is $T = 0.05$.

We plot $\bar{s}_\alpha$ as a function of time for two different values of surface tension coefficients in Fig. 3 on a base 10 logarithmic scale. We can see that $\bar{s}_\alpha$ decreases by a factor of more than 10^2 as the time increases. Thus the time step constraint decreases

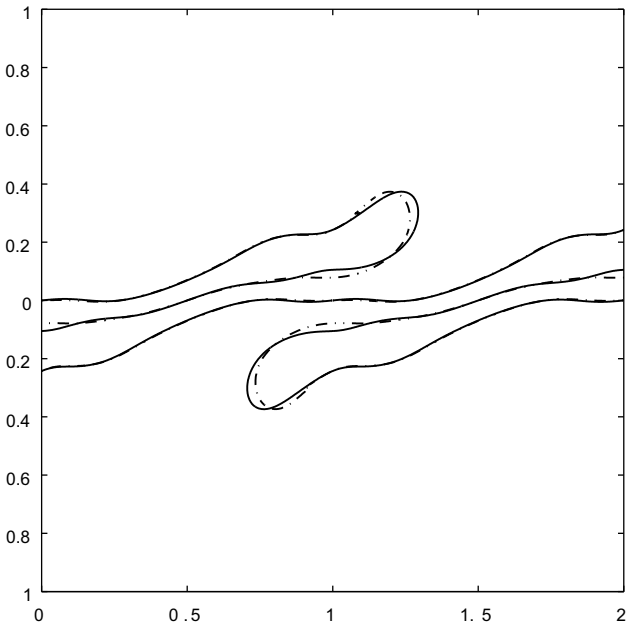

Fig. 5. Resolution comparison for $N = 1024$ (solid) and $N = 1536$ (dashed-dotted) at $t = 2$ s with surface tension coefficient $T = 0.05$.

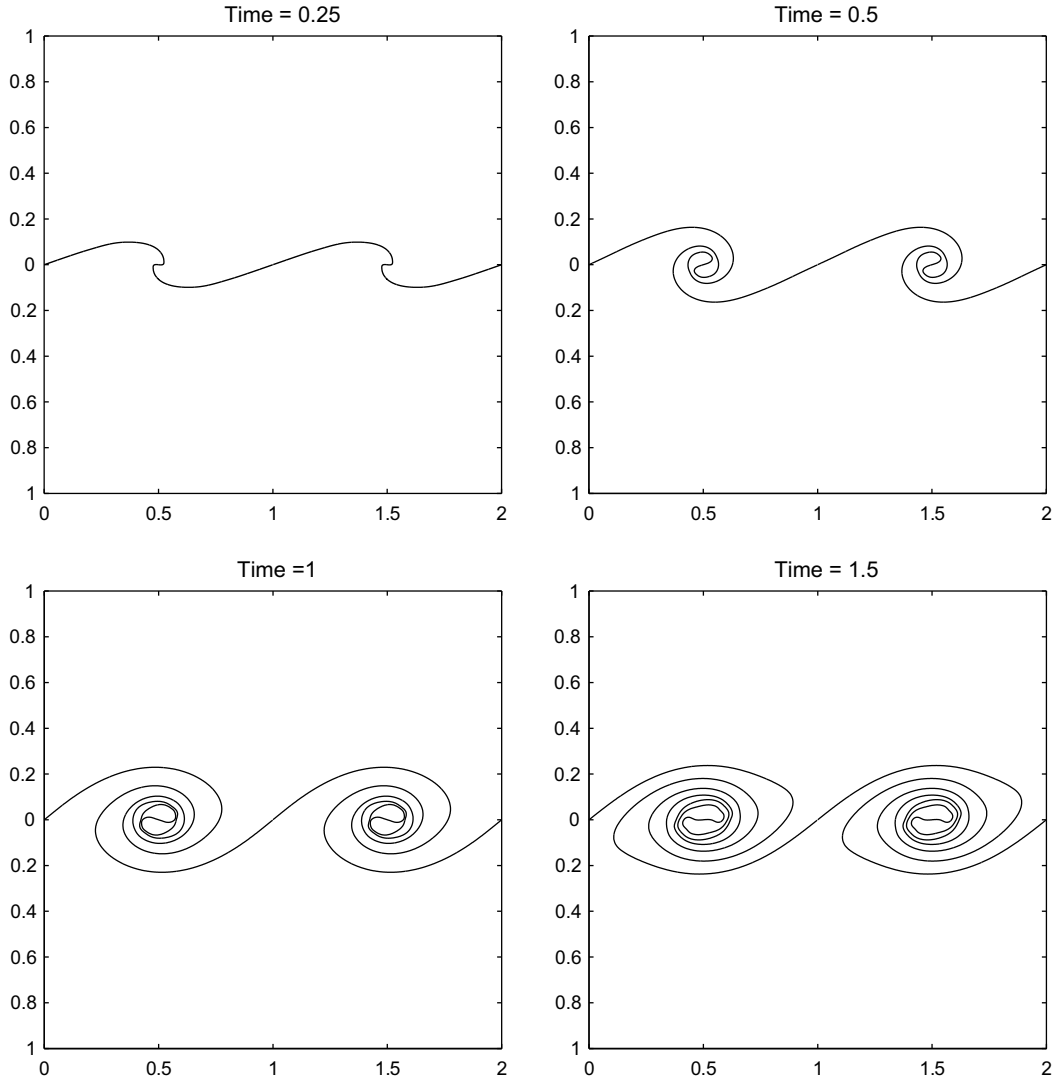

Fig. 6. Configurations of the interface at different times, 0.25 s, 0.5 s, 1 s, 1.5 s, surface tension coefficient is $T = 0.005$.

by at least a factor of 10^3 for an explicit method. This makes the computations using an explicit immersed boundary method very expensive.

The evolution of the interface with surface tension coefficient $T = 0.05$ is shown in Fig. 4. The Weber number in this case is $We = 40/\pi$. For this value of Weber number, the interface is unstable but not completely dominated by roll-up induced through the Kelvin–Helmholtz instability. At the early time, the interface tends to roll up, but the surface tension force prevents it from rolling up further. The interface deforms into two elongated fingers that penetrate each fluid into the other. As the time increases, the neck of these fingers becomes thinner and the length of the fingers becomes longer. In the process, the fluid accumulates near the tip of each figure. Our result is qualitatively similar to the result obtained by Hou et al. [13], Ceniceros and Roma [6] and by Tryggvason et al. [36] for an intermediate Weber number. In Fig. 5, we compare the interface that is computed using $N = 1024$ with that using $N = 1536$ at $t = 2s$. We note that the interface computed by the coarser mesh moves a bit faster than the interface computed by the finer mesh. Our result is in a qualitative agreement with the corresponding result obtained by Ceniceros and Roma (see Fig. 34 of [6]). As pointed by Tryggvason et al. [36], the surface tension forces are spread out more on a coarser mesh. As a result, the effect of surface tension is weaker on a coarser mesh and the interface computed by a coarser mesh moves a bit faster than the interface computed by a finer mesh. We also observe that the accurate computation of the interface at this late stage requires a very high spatial resolution. This is consistent with the result obtained by Ceniceros and Roma (see Fig. 34 of [6]).

When the surface tension coefficient is 0.005, the interface rolls up in time as it is shown in Fig. 6. The Weber number in this case is $We = 400/\pi$. At $t = 0.25$, the interface produces two fingers. These fingers grow in length in the sheet-wise direction. The tips of the fingers broaden. This can be clearly seen at $t = 0.5$. By $t = 1$, the vortex sheet produces another turn and the fingers have become broader and larger. As time increases, the neck of the fingers becomes thinner. It is possible that the sides of the fingers might also collide at some finite time, and so abbreviate their smooth evolution. In the case of an inviscid vortex sheet with surface tension, a finite time pinching singularity has been observed near the thin neck of the finger, forming a trapped bubble. In the case of the viscous vortex sheet with surface tension, this does not appear to the case (at least for this initial data). In Fig. 7, we plot the close-up of the tip region and its pocket of fluid. The neck below the tip is becoming thinner in time. So far as can be discerned, it seems that the thickness of the neck of the finger does not appear to approach to zero in a finite time. If this is the case, viscosity would regularize the pinching singularity of the inviscid vortex sheet.

To obtain better evidence that the viscous vortex sheet does not develop a pinching singularity, we perform a resolution study. In Fig. 8, we compare the interfacial profiles that are obtained by using two large meshsizes: $N = 1024$ and $N = 1536$, respectively. The interfaces computed by the two resolutions agree very well except near the tip of the interface. As we observed before, the interface computed by a coarser mesh moves a bit faster than the interface computed by a finer mesh near the tip. Note that, at this time, there are still more than 10 grid points to resolve the smallest gap, this seems to indicate that our computation is still reasonably resolved.

We remark that Ceniceros and Roma had previously performed a very careful study of the viscous vortex sheet problem for various Weber numbers using an acrlength parameterization of the interface and a locally adaptive mesh to solve for the velocity field [6]. They observed that viscosity tends to stabilize the vortex sheet against pinch-off and presented a detailed analysis to explain why this is the case. This behavior is qualitatively different from the inviscid vortex sheet with surface tension [13]. Our computational results are in a qualitative agreement with the corresponding results obtained in [6].

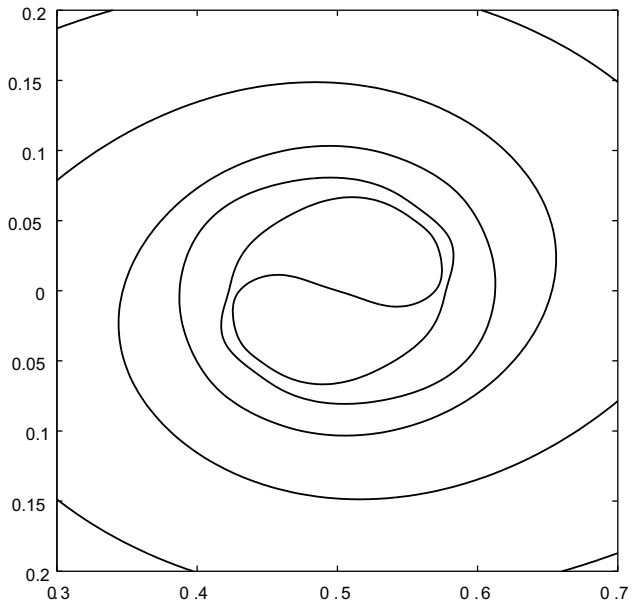

Fig. 7. The closeup view of the vortex sheet roll-up at $t = 1$ s with surface tension coefficient $T = 0.005$.

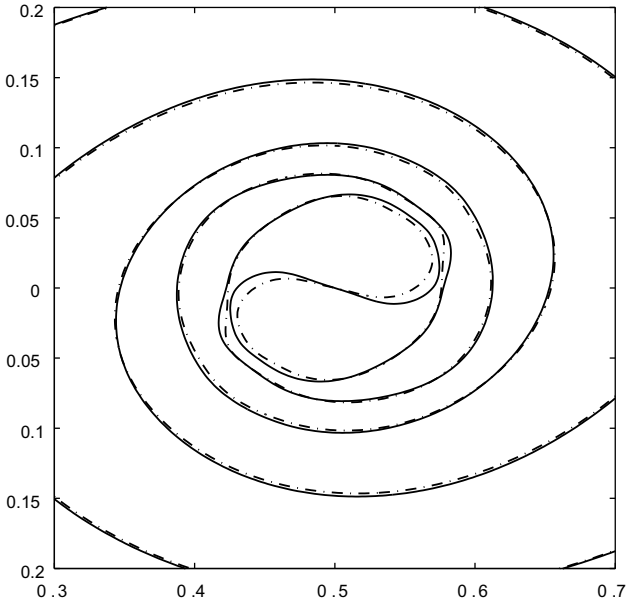

Fig. 8. Resolution comparison for $N = 1024$ (solid) and $N = 1536$ (dashed-dotted) at t = 1 s with surface tension coefficient $T = 0.005$.

It requires very high space resolutions to resolve the nearly singular interfacial dynamics of the viscous vortex sheet problem. Since the immersed boundary method is only first order in space, it is not the best method to use to resolve the nearly singular solution of the viscous vortex sheet problem. Local mesh refinement would be helpful in providing the higher local resolution near the region where the two interfaces are nearly touching as was done in [6]. We do not intend to use the immersed boundary method to perform a careful study of the nearly singular dynamics of the viscous vortex sheet problem. This is not the purpose of our paper. The main purpose of including the viscous vortex sheet problem in our study is to illustrate that the immersed boundary method can be applied to study interfacial dynamics that is more complicated than the prototype elastic problem that we considered earlier.

We should also point out that for the viscous vortex sheet problem with small surface tension coefficient, the stiffness of the problem is much less severe than the elastic problem with very large elastic coefficients that we consider in this paper. For the viscous vortex sheet problem, the use of an arclength parameterization can already reduce significantly the stiffness associated with surface tension, as observed by Ceniceros and Roma [6]. The combination of the acrlength parameterization of the interface and the use of locally adaptive mesh to solve for the velocity field seems to offer a computationally effective method to study the nearly singular behavior of the viscous vortex sheet problem [6].

8. Concluding remarks

In this paper, we have developed a very efficient semi-implicit immersed boundary scheme for solving the immersed boundary problem for the Navier–Stokes equations. The immersed boundary method has emerged as one of the most useful numerical methods in computing fluid structure interaction, and has found numerous applications. But it also suffers from the severe time step stability restriction due to the stiffness of the elastic force. Guided by our stability analysis, we have developed an efficient semi-implicit scheme which removes the stiffness of the immersed boundary method. We have demonstrated that our semi-implicit scheme has much better stability property than the explicit scheme. More importantly, unlike most existing implicit or semi-implicit schemes, our semi-implicit scheme can be implemented very efficiently. In fact, our semi-implicit scheme has a computational complexity that is essentially the same as that of an explicit scheme in each time step, but with a much better stability property. The saving in the computational cost is very substantial. We have demonstrated this improved stability for a range of parameters and numerical resolutions. Our computational results have shown that the more severe the stiffness of the immersed boundary problem becomes, the bigger the computational saving our semi-implicit schemes can offer.

One of the essential steps in developing our semi-implicit scheme is to obtain an unconditionally stable semi-implicit discretization of the immersed boundary problem. This provides us with a building block to construct our efficient semi-implicit schemes. By applying the small scale decomposition to the unconditionally stable semi-implicit time discretization and further simplifying the leading order singular kernel, we obtain our SSD semi-implicit scheme. The advantage of this semi-implicit scheme is that the leading order term can be expressed as a convolution operator, which can be evaluated explicitly using the Fourier transformation. This allows us to solve for the implicit solution explicitly in the spectral space. Another contribution of this paper is to introduce a variant of the semi-implicit ADI discretization for the convection term to eliminate the usual CFL stability restriction. Both the ADI discretization of the convection term and the leading order semi-im-

plicit scheme using the Small Scale Decomposition can be inverted very efficiently. This enables us to obtain a semi-implicit scheme which has the same order of complexity as the explicit scheme per time step but with a much better stability property.

As an application, we have applied our semi-implicit scheme to study the large time behavior of the viscous vortex sheet with surface tension. The Reynolds number we consider is equal to 10,000, which is very challenging computationally. For large Weber number, we observe that the vortex sheet rolls up and forms a thin neck near the tip of the finger. Our computations seem to suggest that the thickness of the neck of the finger does not approach to zero in a finite time. Our computational result is in a qualitative agreement with the previous result obtained by Ceniceros and Roma [6] who observed that viscosity tends to regularize the finite time pinching singularity of the inviscid vortex sheet and performed a detailed study to explain why this is the case. This behavior is qualitatively different from the inviscid vortex sheet with surface tension.

The methodology that we present here can be generalized to the three dimensional case. We are currently developing an efficient semi-implicit scheme for the 3D immersed boundary problem. This will be reported elsewhere in the future.

Acknowledgement

We would like to thank Profs. Charles Peskin and Hector Ceniceros for a number of stimulating discussions on the Immersed Boundary method. The research was in part supported by DOE under the DOE Grant DE-FG02-06ER25727 and by NSF under the NSF FRG Grant DMS-0353838, ITR Grant ACI-0204932, and DMS-0713670.

References

[1] M. Abramowitz, I.A. Stegun, Modified Bessel Functions I and K, Dover, New York, 9th printing, 1972.
[2] D.L. Brown, R. Cortex, M. Minion, Accurate projection methods for the incompressible Navier–Stokes equations, J. Comput. Phys. 168 (2001) 464–499.
[3] R.P. Beyer, A computational model of the cochlea using the immersed boundary method, J. Comput. Phys. 98 (1992) 145–162.
[4] R. Cortez, N. Cowen, R. Dillon, L.J. Fauci, Simulation of swimming organisms: coupling internal mechanics with external fluid dynamics, Comput. Sci. Eng. 6 (2004) 38–45.
[5] Y.C. Chang, T.Y. Hou, B. Merriman, S. Osher, A level set formulation of Eulerian interface capturing methods for incompressible fluid flows, J. Comput. Phys. 124 (1996) 449–464.
[6] H.D. Ceniceros, A.M. Roma, Study of the long-time dynamics of viscous vortex sheet with a fully adaptive nonstiff method, Phys. Fluid 16 (2004) 4285–4318.
[7] L. Fauci, C.S. Peskin, A computational model of aquatic animal locomotion, J. Comput. Phys. 77 (1988) 85–108.
[8] L. Fauci, C.S. Peskin, A fast numerical method for solving the three-dimensional Stokes equations in the presence of suspended particles, J. Comput. Phys. 79 (1988) 50–69.
[9] L.J. Fauci, Interaction of oscillating silaments – a computational study, J. Comput. Phys 86 (1990) 294–313.
[10] A.L. Fogelson, A mathematical model and numerical methods for studying platelet adhesion and aggregation during blood clotting, J. Comput. Phys. 56 (1984) 111–134.
[11] E. Givelberg, Modeling elastic shells immersed in fluid, PhD Thesis, Courant Institute of Mathematical Sciences, New York University, 1997.
[12] T.Y. Hou, J. Lowengrub, M. Shelley, Removing the stiffness from interfacial flows with surface tension, J. Comput. Phys. 114 (1994) 312–338.
[13] T.Y. Hou, J. Lowengrub, M. Shelley, The long-time motion of vortex sheets with surface tension, Phys. Fluid A 9 (1997) 1933–1954.
[14] M. Hopkins, L.J. Fauci, A computational model of the collective fluid dynamics of motile microorganisms, J. Fluid Mech. 455 (2002) 149–174.
[15] T.Y. Hou, Z. Shi, Removing the stiffness of elastic force from the immersed boundary method for the 2D stokes equations, J. Comput Phys., in press, doi:10.1016/j.jcp.2008.03.002.
[16] 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.
[17] A.A. Mayo, C.S. Peskin, An implicit numerical method for fluid dynamics problems with immersed elastic boundaries, Contemp. Math. 141 (1993) 261–277.
[18] D.M. McQueen, C.S. Peskin, E.L. Yellin, Fluid dynamics of the mitral valve: physiological aspects of a mathematical model, Am. J. Physiol. 242 (1982) H1095–H1110.
[19] D.M. McQueen, C.S. Peskin, Computer assisted design of pivoting-disc prosthetic mitral valves, J. Thorac. Cardiovasc. Surg. 86 (1983) 126–135.
[20] D.M. McQueen, C.S. Peskin, Computer assisted design of butterfly bileaflet valves for mitral position, Scand. J. Thorac. Cardiovasc. Surg. 19 (1985) 139–148.
[21] D.M. McQueen, C.S. Peskin, A three-dimensional computational method for blood flow in the heart: (II) Contractile fibers, J. Comput. Phys. 82 (1989) 289–297.
[22] D.M. McQueen, C.S. Peskin, Heart simulation by an immersed boundary method with formal second order accuracy and reduced viscosity, in: Mechanics for a New Millennium, Proceedings of the International Conference on Theoretical and Applied mechanics (ICTAM), Kluwer Academic Publishers, 2000.
[23] L.A. Miller, C.S. Peskin, When vortices stick: and aerodynamic transition in tiny insects, J. Exp. Biol. 207 (2004) 3073–3088.
[24] L.A. Miller, C.S. Peskin, A computational fluid dynamics of 'clap and fling' in small insects, J. Exp. Biol. 208 (2005) 195–212.
[25] Y. Mori, C.S. Peskin, Implicit second-order immersed boundary methods with boundary mass, Comput. Methods Appl. Mech. Engrg. 197 (2008) 2049–2067.
[26] E.P. Newren, A.L. Fogelson, R.D. Guy, R.M. Kirby, Unconditionally stable discretizations of the immersed boundary equations, J. Comput. Phys. 222 (2007) 702–719.
[27] C.S. Peskin, Numerical study of blood flow in the heart, J. Comput. Phys. 25 (1977) 220–252.
[28] C.S. Peskin, D.M. McQueen, A three-dimensional computational method for blood flow in the heart: (I) Immersed elastic fibers in a viscous incompressible fluid, J. Comput. Phys. 81 (1989) 372–405.
[29] C.S. Peskin, The immersed boundary method, Acta Numer. 11 (2002) 479–517.
[30] R. Peyret, T.D. Taylor, Computational Methods for Fluid Flow, Springer-Verlag, New York, 1983.
[31] M.E. Rosar, C.S. Peskin, Fluid flow in collapsible elastic tubes: a three-dimensional numerical model, New York J. Math. 7 (2001) 281–302.
[32] J.M. Stockie, B.R. Wetton, Stability analysis for the immersed fiber problem, SIAM J. Appl. Math. 55 (1995) 1577–1591.
[33] J.M. Stockie, S.I. Green, Simulating the motion of flexible pulp fibers using the immersed boundary method, J. Comput. Phys. 147 (1998) 147–165.
[34] 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.

[35] 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. Stat. Comput. 13 (1992) 1361–1376.
[36] W. Tauber, S.O. Unverdi, G. Tryggvason, The nonlinear behavior of a sheared immiscible fluid interface, Phys. Fluid 14 (2002) 2871–2885.
[37] N.T. Wang, A.L. Fogelson, Computational methods or continuum models of platelet aggregation, J. Comput. Phys. 151 (1999) 649–675.
[38] L. Zhu, C.S. Peskin, Simulation of a flexible flapping filament in a flowing soap film by the immersed boundary method, J. Comput. Phys. 179 (2002) 452–468.