Available online at www.sciencedirect.com
SCIENCE DIRECT®
Journal of Computational Physics 216 (2006) 454–493
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

An immersed interface method for simulating the interaction of a fluid with moving boundaries

Sheng Xu *, Z. Jane Wang

Department of Theoretical and Applied Mechanics, Cornell University, KL214 Kimball Hall, Ithaca, NY 14853, USA

Received 21 March 2005; received in revised form 1 December 2005; accepted 16 December 2005
Available online 14 February 2006

Abstract

In the immersed interface method, boundaries are represented as singular force in the Navier–Stokes equations, which enters a numerical scheme as jump conditions. Recently, we systematically derived all the necessary spatial and temporal jump conditions for simulating incompressible viscous flows subject to moving boundaries in 3D with second-order spatial and temporal accuracy near the boundaries [Sheng Xu, Z. Jane Wang, Systematic derivation of jump conditions for the immersed interface method in three-dimensional flow simulation, SIAM J. Sci. Comput., 2006, in press]. In this paper we implement the immersed interface method to incorporate these jump conditions in a 2D numerical scheme. We study the accuracy, efficiency and robustness of our method by simulating Taylor–Couette flow, flow induced by a relaxing balloon, flow past single and multiple cylinders, and flow around a flapping wing. Our results show that: (1) our code has second-order accuracy in the infinity norm for both the velocity and the pressure; (2) the addition of an object introduces relatively insignificant computational cost; (3) the method is equally effective in computing flow subject to boundaries with prescribed force or boundaries with prescribed motion.

Keywords: Immersed interface method; Immersed boundary method; Cartesian grid method; Moving deformable boundaries; Complex geometries; Flow around multiple objects; Singular force

1. Introduction

It is important to accurately and efficiently resolve moving boundaries and their effects on fluids in numerical simulations of fluid–structure interactions. Body-fitted grid methods, which employ structured or unstructured body-fitted grids, are designed to resolve boundaries and their effects with high-order accuracy, but it is computationally costly to update grids in moving boundary problems. Various Cartesian grid methods have been developed to avoid the grid regeneration. They allow for fast flow solvers and have the advantages of simplicity and efficiency. One class of Cartesian grid methods are suitable for solving flows with moving boundaries undergoing prescribed motion. Examples include the virtual/immersed boundary method

* Corresponding author. Tel.: +1 607 254 8593.
E-mail addresses: sx12@cornell.edu (S. Xu), jane.wang@cornell.edu (Z.J. Wang).

0021-9991/$ - see front matter © 2006 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2005.12.016

[9,28,6,14,29], the sharp interface method [34,39], the ghost fluid method [7,32,10], PHYSALIS [25,31], Calhoun's method [3] and Russell and Wang's method [27]. Another class of Cartesian grid methods are more suitable to solve flows with boundaries moved and deformed by driving force, most notably, Peskin's immersed boundary method [21,22] and the more recent immersed interface method [17–19,16]. Our goal of this paper is to develop the immersed interface method for simulating flow with both types of moving boundaries in an accurate and efficient way.

In his simulation of blood flow in the heart, Peskin introduced the immersed boundary method [21,22]. In this method, the boundary of an immersed object is treated as a set of Lagrangian fluid particles. The configuration of these Lagrangian fluid particles determines a force distribution in the fluid to represent the effect of the object. The force distribution is determined by a prescribed constitutive law of material, and it appears in the form of the Dirac delta function. In this sense, Peskin's immersed boundary method is a mathematical formulation, in which singular force is added to the Navier–Stokes equations for modeling immersed boundaries. This formulation naturally couples a fluid with multiple moving objects, and can be computed in an efficient way. A rigorous derivation of the formulation from the principle of least action can be found in [24].

The numerical implementation of the immersed boundary method employs a fixed Cartesian grid for a fluid and a Lagrangian grid for an immersed boundary. The communication between the fluid and the immersed boundary is achieved through the spreading of the singular force from the Lagrangian grid to the Cartesian grid and the interpolation of the velocity from the Cartesian grid to the Lagrangian grid. A discrete Dirac delta function is used to spread the singular force and interpolate the velocity. There are multiple choices of the discrete Dirac delta function, which is constructed to preserve various moments. The use of the discrete Dirac delta function removes the singularity in the governing equations and therefore the discontinuities of the flow field. Thus, standard discretization schemes can be adopted without modifications. To confine the thickness of the interface between a fluid and an object, the discrete Dirac delta function has a narrow support, and is dependent on the grid size.

Since the immersed boundary method was introduced in 1972 [21], it has been widely used in the simulation of fluid–structure interactions, especially in biological fluid dynamics. Examples can be found in [24]. Despite its efficiency and robustness, the initial implementations of the method had the following shortcomings. First, it was only first-order accurate in space; Second, there was a systematic tendency for a closed immersed boundary to slowly lose volume as though the fluid were leaking out; Third, a solution which is piecewise smooth across an immersed boundary was smeared out.

In the past 30 years, there have been various research efforts to analyze and improve the immersed boundary method. Beyer and LeVeque [1] examined the accuracy of the method for the one-dimensional diffusion equation, and found that additional terms for the discrete approximation of the Dirac delta function are sometimes necessary in order to achieve second-order accuracy, but it is unclear how to maintain the second-order accuracy for flow simulation in higher dimensions. A formally second-order immersed boundary method was proposed by Lai and Peskin [15], but the second-order accuracy is achieved only if the Dirac delta function is approximated by a grid-independent fixed smooth function. In practice, it is still first-order accurate. Realizing that the projection of a discrete Dirac delta function onto a divergence-free space may be computed analytically, Cortez and Minion [4] devised the blob projection immersed boundary method, which displayed formally fourth-order convergence rates of their background flow solver. However, the analytical form of the projection depends on the velocity boundary conditions imposed on the computational domain. Since the pressure was not reported, it is also unclear how accurately the pressure can be recovered.

Analyzing the leakage problem in the immersed boundary method, Peskin and Printz [23] found that the volume loss enclosed by an immersed boundary is caused by the violation of the divergence-free condition at Lagrangian fluid particles. They ruled out the obvious explanation that fluid escapes between discrete Lagrangian fluid particles which define the immersed boundary, and introduced a recipe for the construction of a finite-difference divergence operator to achieve better volume conservation. The blob projection immersed boundary method by Cortez and Minion [4] also achieved good volume conservation.

Other analysis and improvement of the immersed boundary method include the stability analysis by Tu and Peskin [33] and Stockie and Wetton [30], the adaptive version by Roma et al. [26], and the inclusion of boundary mass by Zhu and Peskin [40]. However, despite recent improvements, the method still suffers from some shortcomings, in particular first-order accuracy in general.

Motivated by the goal to eventually obtain second-order accuracy in Peskin's immersed boundary method, LeVeque and Li [17,18] developed the immersed interface method. The key idea of the immersed interface method, which is also the main difference from the immersed boundary method, is to incorporate the jump conditions caused by the Dirac delta function into finite difference schemes. The approximation of the Dirac delta function by a smooth function is therefore avoided. The immersed interface method was originally proposed for elliptic equations [17] and Stokes equations [18]. Later, it was extended to one-dimensional nonlinear parabolic equations [36], Poisson equations with Neumann boundary conditions [37,8], and the two-dimensional incompressible Navier–Stokes equations [19,16].

For fluid dynamics problems, the immersed interface method is based on the same mathematical formulation as Peskin's immersed boundary method, but the coupling between a fluid and an object is now handled by incorporating jump conditions. If necessary jump conditions are all known, the immersed interface method can achieve second-order or even higher-order accuracy. The sharpness of an interface computed by the method does not depend on grid resolutions. Furthermore, the method shows very good conservation of the mass enclosed by a no-penetration boundary. Thus, the immersed interface method shares the advantages of the immersed boundary method with the same mathematical formulation, while overcoming some of its shortcomings by eliminating the use of discrete Dirac delta functions.

The applicability of the immersed interface method depends on whether the necessary jump conditions are known. Recently, we systematically derived the jump conditions of all first-, second- and third-order spatial derivatives of the velocity and the pressure as well as first- and second-order temporal derivatives of the velocity for the 3D incompressible Navier–Stokes equations subject to singular force [38]. Using these jump conditions, the immersed interface method can be applied to the simulation of 3D incompressible viscous flows with local first- or second-order spatial and temporal discretization accuracy. In this paper, we obtain the jump conditions in 2D from our 3D results [38] by taking one direction in 3D as uniform, and implement the immersed interface method to simulate the interaction of a fluid with moving boundaries. The method we develop in this paper can simulate two classes of flow problems, one with boundaries moved and deformed by driving force and the other with boundaries prescribed with known motion.

Li and Lai [19] and Lee and LeVeque [16] have implemented the immersed interface method for some two-dimensional flows and achieved second-order spatial accuracy in their simulations. The current paper differs from their work in the following aspects: (1) we derive the jump conditions in this paper from our three-dimensional theoretical results [38], so the numerical tests in the current paper serve in part to verify our previous theoretical derivation; (2) in addition to more spatial jump conditions, we also derive temporal jump conditions and examine their effect on temporal integration; (3) we simulate flow problems with boundaries moved and deformed by driving force and with boundaries in prescribed motion, and investigate the spatial and temporal convergence and accuracy of our method against known analytical flow solutions and canonical flow examples; (4) we investigate the efficiency and robustness of the method to simulate flow with multiple moving boundaries.

This paper is written with sufficient details so that an interested reader can program and test the method. In Section 2, we present the mathematical formulation of governing equations in the immersed interface method. In Section 3, we describe the immersed interface method and give finite difference schemes with jump conditions incorporated. In Section 4, we present the necessary jump conditions for achieving first-order local spatial and temporal discretization accuracy in 2D simulation. Those jump conditions are functions of the singular force. In Section 5, we discuss the construction of the singular force. In Section 6, we implement the MAC scheme [11] as a basic flow solver and also give interpolation schemes and fluid force calculations. In Section 7, we provide numerical tests to examine the accuracy, efficiency and robustness of the immersed interface method. Section 8 is conclusions.

2. Governing equations subject to singular force

Both the immersed interface method and the immersed boundary method model objects as singular force in the Navier–Stokes equations. The nondimensional 2D Navier–Stokes equations subject to singular force are

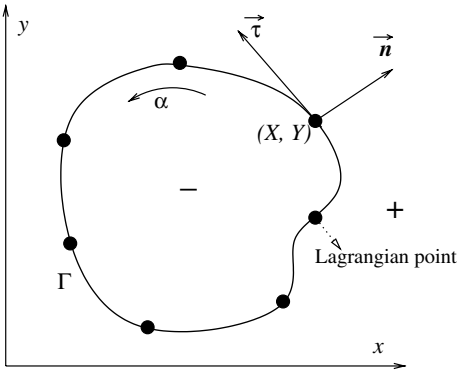


Fig. 1. Geometric description of an object surface.

$$\frac{\partial \vec{v}}{\partial t} + \nabla \cdot (\vec{v}\vec{v}) = -\nabla p + \frac{1}{Re}\Delta \vec{v} + \sum_{l=1}^{M} \vec{F}_l, \tag{1}$$

$$\nabla \cdot \vec{v} = 0, \tag{2}$$

where t is time, $\vec{v} := (u, v)$ is the velocity, p is the pressure, Re is the Reynolds number, M is the number of objects, and $\vec{F}_l$ is the singular force from object l. Taking the divergence of the momentum equation (1) gives the equation for pressure p as

$$\Delta p = -\frac{\partial D}{\partial t} - 2\nabla \cdot (\vec{v}D) + \frac{1}{Re}\Delta D + 2\left(\frac{\partial u}{\partial x}\frac{\partial v}{\partial y} - \frac{\partial u}{\partial y}\frac{\partial v}{\partial x}\right) + \sum_{l=1}^{M} \nabla \cdot \vec{F}_l, \tag{3}$$

where $D = \nabla \cdot \vec{v}$ is the divergence of the velocity and $\vec{x} := (x, y)$ is the Cartesian coordinates. We keep terms with divergence D in Eq. (3) to enforce the divergence-free condition in the MAC scheme, which we will use to solve Eqs. (1) and (3).

Since our numerical treatment of each object is the same, we omit subscript l associated with object l in later presentation. We consider the situation in which the singular force is applied to the closed smooth boundary of object l. Referring to Fig. 1, the closed boundary is a closed curve in 2D, and we denote it by Γ and its coordinates by $\vec{X} := (X, Y)$. We parametrize curve Γ at a reference time by nondimensionalized Lagrangian parameter α. The singular force $\vec{F}$ can be expressed by

$$\vec{F}(x, y, t) = \int_{\Gamma} \vec{f}(\alpha, t)\delta(x - X(\alpha, t))\delta(y - Y(\alpha, t))\mathrm{d}\alpha, \tag{4}$$

where $\vec{f} := (f_x, f_y)$ is the nondimensional singular force density, and $\delta(\cdot)$ is the nondimensional Dirac delta function. We assume f_x and f_y are smooth functions of α and t.

3. Finite differences with jump contributions

Because of the singular force applied on curve Γ, a flow field governed by Eqs. (1) and (2) is generally not smooth across the curve. The immersed interface method differs from the immersed boundary method in its handling of the force singularity. Instead of approximating the Dirac delta function by smooth functions, the immersed interface method modifies the standard discretization schemes to include the discontinuities of the flow field. The modification is based on the generalized Taylor expansion.

Lemma 1 (Generalized Taylor expansion). *Assume function $g(z)$ has discontinuity points of the first kind at $z_1, z_2, \ldots, z_m$ in (z_0, z_{m+1}), $z_0 < z_1 < z_2 < \ldots < z_m < z_{m+1}$, and $g(z) \in \mathscr{C}^{\infty}(z_0, z_1) \cup (z_1, z_2) \cup \cdots \cup (z_m, z_{m+1})$. $g(z)$ can be either continuous or discontinuous at z_0 and z_{m+1}. Let $[g^{(n)}(z_l)] = g^{(n)}(z_l^+) - g^{(n)}(z_l^-)(n = 1, 2, \ldots; l = 1, 2, \ldots, m)$. Then*

$$g(z_{m+1}^-) = \sum_{n=0}^{\infty} \frac{g^{(n)}(z_0^+)}{n!}(z_{m+1} - z_0)^n + \sum_{l=1}^{m}\sum_{n=0}^{\infty} \frac{[g^{(n)}(z_l)]}{n!}(z_{m+1} - z_l)^n. \tag{5}$$

The proof of the above lemma was presented in [38].

We apply the above lemma to construct central finite difference schemes where the discontinuity may occur at most once between two adjacent grid points.

Lemma 2 (Generalized central finite differences). *Let* $x_{i+1} - x_i = x_i - x_{i-1} = h > 0$ *and* $x_{i-1} < \xi < x_i \leqslant \eta < x_{i+1}$. *Suppose* $u(x)$ *is infinitely smooth except at discontinuity points of the first kind,* ξ *and* η. $u(x)$ *can be either continuous or discontinuous at* x_{i+1} *and* x_{i-1}. *Then*

$$\frac{\mathrm{d}u(x_i^-)}{\mathrm{d}x} = \frac{u(x_{i+1}^-) - u(x_{i-1}^+)}{2h} + \frac{1}{2h}\left(\sum_{n=0}^{2}\frac{-[u^{(n)}(\xi)]}{n!}(x_{i-1}-\xi)^n - \sum_{n=0}^{2}\frac{[u^{(n)}(\xi)]}{n!}(x_{i+1}-\eta)^n\right) + \mathrm{O}(h^2), \tag{6}$$

$$\frac{\mathrm{d}^2u(x_i^-)}{\mathrm{d}x^2} = \frac{u(x_{i+1}^-) - 2u(x_i) + u(x_{i-1}^+)}{h^2} - \frac{1}{h^2}\left(\sum_{n=0}^{3}\frac{-[u^{(n)}(\xi)]}{n!}(x_{i-1}-\xi)^n + \sum_{n=0}^{3}\frac{[u^{(n)}(\eta)]}{n!}(x_{i+1}-\eta)^n\right) + \mathrm{O}(h^2). \tag{7}$$

Other finite difference schemes with different orders can also be constructed based on Lemma 1, but the availability of jump conditions sets an upper-limit on the order of accuracy, as stated in the following proposition.

Proposition 1. *The highest order of a finite difference scheme for* $u^{(n)}(x)$ *with a stencil containing a discontinuity point* ζ *is* $m - n + 1$, *where* m *takes a maximum number such that jump conditions* $[u^{(l)}(\zeta)]$ $(l = 0, 1, 2, \ldots, m)$ *are all known.*

The highest order of spatial derivatives in momentum equation (1) and pressure equation (3) is 2. According to Proposition 1, to discretize these derivatives with first-order accuracy at grid points near an object boundary, the jump conditions of the velocity, the pressure and their first-order and second-order spatial derivatives are needed. To achieve second-order accuracy, the jump conditions of their third-order spatial derivatives are also needed.

If an object is moving, the temporal derivatives of the velocity at a grid point may have jumps whenever the object boundary crosses that grid point. Suppose a boundary passes a grid point at time $t_1, t_2, \ldots, t_m$ between time t_0 and t_{m+1}. Then the relation for velocity $\vec{v}$ at the grid point between time t_0 and t_{m+1} is given by

$$\vec{v}(t_{m+1}) = \sum_{n=0}^{\infty}\frac{\partial^n\vec{v}(t_0)}{\partial t^n}\frac{(t_{m+1}-t_0)^n}{n!} + \sum_{l=1}^{m}\sum_{n=0}^{\infty}\left[\left[\frac{\partial^n\vec{v}(t_l)}{\partial t}\right]\right]\frac{(t_{m+1}-t_l)^n}{n!}, \tag{8}$$

where $[[\cdot]]$ denotes a temporal jump at time t as $[[\cdot]] := (\cdot)_{t^+} - (\cdot)_{t^-}$. Eq. (8) follows directly from Lemma 1. The highest order of temporal derivatives in momentum equation (1) is 1. According to Proposition 1, to discretize these derivatives with first-order accuracy at grid points crossed by an object boundary, the jump conditions of the first-order temporal derivatives of the velocity are needed. To achieve second-order accuracy, the jump conditions of second-order temporal derivatives of the velocity are needed as well.

4. Spatial and temporal jump conditions

In this section, we give the necessary jump conditions for achieving first-order local spatial and temporal discretization accuracy in the 2D Navier–Stokes equations. They are derived from the 3D results in [38] by taking one direction in 3D as uniform.

For curve Γ in Fig. 1, unit tangential vector $\vec{\tau}$ and unit normal vector $\vec{n}$ are defined as

$$\vec{\tau} := (\tau_x, \tau_y) = \frac{1}{J}\left(\frac{\partial X}{\partial\alpha}, \frac{\partial Y}{\partial\alpha}\right), \quad \vec{n} := (n_x, n_y) = (\tau_y, -\tau_x),$$

where $J = \sqrt{(\frac{\partial X}{\partial\alpha})^2 + (\frac{\partial Y}{\partial\alpha})^2}$. As before, we use $[\cdot]$ to denote a spatial jump, i.e. $[\cdot] := (\cdot)_{\Gamma^+} - (\cdot)_{\Gamma^-}$, where Γ^+ is at the side of Γ in the direction of normal $\vec{n}$, and Γ^- is at the other side of Γ. In the Cartesian coordinate system, tangential force density f_τ and normal force density f_n are defined as

$$f_\tau = \frac{f_x\tau_x + f_y\tau_y}{J}, \quad f_n = \frac{f_x n_x + f_y n_y}{J}.$$

4.1. Spatial jump conditions

The spatial jump conditions used for 2D simulation are

$$[\vec{v}] = 0, \quad \left[\frac{\partial \vec{v}}{\partial x}\right] = -Re\tau_y\vec{\tau}f_\tau, \quad \left[\frac{\partial \vec{v}}{\partial y}\right] = Re\tau_x\vec{\tau}f_\tau,$$

$$\left[\frac{\partial^2 \vec{v}}{\partial x^2}\right] = \vec{r}_{u1}(\tau_x^2 - \tau_y^2) + \vec{r}_{u2}(2\tau_x\tau_y) + \vec{r}_{u3}(\tau_y^2),$$

$$\left[\frac{\partial^2 \vec{v}}{\partial y^2}\right] = \vec{r}_{u1}(\tau_y^2 - \tau_x^2) - \vec{r}_{u2}(2\tau_x\tau_y) + \vec{r}_{u3}(\tau_x^2),$$

$$\left[\frac{\partial^2 \vec{v}}{\partial x\partial y}\right] = \vec{r}_{u1}(2\tau_x\tau_y) + \vec{r}_{u2}(\tau_y^2 - \tau_x^2) - \vec{r}_{u3}(\tau_x\tau_y),$$

$$[p] = f_n, \quad \left[\frac{\partial p}{\partial x}\right] = \frac{\tau_x}{J}\frac{\partial f_n}{\partial \alpha} + \frac{\tau_y}{J}\frac{\partial f_\tau}{\partial \alpha}, \quad \left[\frac{\partial p}{\partial y}\right] = \frac{\tau_y}{J}\frac{\partial f_n}{\partial \alpha} - \frac{\tau_x}{J}\frac{\partial f_\tau}{\partial \alpha},$$

$$\left[\frac{\partial^2 p}{\partial x^2}\right] = r_{p1}(\tau_x^2 - \tau_y^2) + r_{p2}(2\tau_x\tau_y) + r_{p3}(\tau_y^2),$$

$$\left[\frac{\partial^2 p}{\partial y^2}\right] = r_{p1}(\tau_y^2 - \tau_x^2) - r_{p2}(2\tau_x\tau_y) + r_{p3}(\tau_x^2),$$

$$\left[\frac{\partial^2 p}{\partial x\partial y}\right] = r_{p1}(2\tau_x\tau_y) + r_{p2}(\tau_y^2 - \tau_x^2) - r_{p3}(\tau_x\tau_y),$$

where $\vec{r}_{u1}$, $\vec{r}_{u2}$, $\vec{r}_{u3}$, r_{p1}, r_{p2} and r_{p3} are

$$\vec{r}_{u1} = -\frac{1}{J^2}\left(\frac{\partial J\tau_x}{\partial \alpha}\left[\frac{\partial \vec{v}}{\partial x}\right] + \frac{\partial J\tau_y}{\partial \alpha}\left[\frac{\partial \vec{v}}{\partial y}\right]\right),$$

$$\vec{r}_{u2} = -\frac{1}{J}\left(Re\frac{\partial f_\tau\vec{\tau}}{\partial \alpha} + \frac{\partial n_x}{\partial \alpha}\left[\frac{\partial \vec{v}}{\partial x}\right] + \frac{\partial n_y}{\partial \alpha}\left[\frac{\partial \vec{v}}{\partial y}\right]\right),$$

$$\vec{r}_{u3} = Re[\nabla p],$$

$$r_{p1} = \frac{1}{J^2}\left(\frac{\partial^2 f_n}{\partial \alpha^2} - \frac{\partial J\tau_x}{\partial \alpha}\left[\frac{\partial p}{\partial x}\right] - \frac{\partial J\tau_y}{\partial \alpha}\left[\frac{\partial p}{\partial y}\right]\right),$$

$$r_{p2} = \frac{1}{J}\left(\frac{\partial}{\partial \alpha}\left(\frac{1}{J}\frac{\partial f_\tau}{\partial \alpha}\right) - \frac{\partial n_x}{\partial \alpha}\left[\frac{\partial p}{\partial x}\right] - \frac{\partial n_y}{\partial \alpha}\left[\frac{\partial p}{\partial y}\right]\right),$$

$$r_{p3} = 2\left[\frac{\partial u}{\partial x}\frac{\partial v}{\partial y}\right] - 2\left[\frac{\partial u}{\partial y}\frac{\partial v}{\partial x}\right].$$

Terms with the form of $[a \cdot b]$ appear here in r_{p3} and also later. Note that $[a \cdot b] \neq [a] \cdot [b]$. The correct calculation of $[a \cdot b]$ is

$$[a \cdot b] = a^- \cdot [b] + b^- \cdot [a] + [a] \cdot [b] = a^+ \cdot [b] + b^+ \cdot [a] - [a] \cdot [b], \tag{9}$$

where a^-, b^-, a^+ and b^+ can be obtained through interpolation. We give the interpolation scheme in Section 6.4.

4.2. Temporal jump conditions

When curve Γ passes a fixed point $\vec{x}$ in space at time t, using $\vec{X}$ to denote the point on Γ which coincides with the point $\vec{x}$, we have the following relation between $[[\psi(\vec{X}, t)]] = (\psi)_{t^+} - (\psi)_{t^-}$ and $[\psi(\vec{X}, t)] = (\psi)_{\Gamma^+} - (\psi)_{\Gamma^-}$ for quantity ψ,

$$[[\psi]] = \begin{cases} [\psi], & \vec{v}(\vec{x}) \cdot \vec{n}(\vec{X}) < 0, \\ -[\psi], & \vec{v}(\vec{x}) \cdot \vec{n}(\vec{X}) > 0. \end{cases} \tag{10}$$

If $\vec{v}(\vec{x}) \cdot \vec{n}(\vec{X}) = 0$, we can approximate temporal derivatives at $\vec{x}$ by those at $\vec{X}|_{\Gamma^+}$ or $\vec{X}|_{\Gamma^-}$ with $[[\cdot]] = 0$. According to Eq. (10), we need spatial jump conditions $[\vec{v}]$ and $\left[\frac{\partial \vec{v}}{\partial t}\right]$ in our 2D simulation. The latter is given by

$$\left[\frac{\partial \vec{v}}{\partial t}\right] = -\vec{v} \cdot [\nabla \vec{v}].$$

5. Construction of singular force

The jump conditions given in Section 4 are functions of the singular force density. The construction of the singular force depends on whether the motion of an immersed boundary is prescribed or driven by a force law based on its deformation.

We consider the general case in which the immersed boundary has Lagrangian mass density $\rho_s(\alpha)$. We can write the density $\rho(x, y, t)$ of the whole system as

$$\rho(x, y, t) = \rho_f(x, y, t) + \int_\Gamma \rho_s(\alpha)\delta(x - X(\alpha, t))\delta(y - Y(\alpha, t))\mathrm{d}\alpha, \tag{11}$$

where $\rho_f(x, y, t)$ is the fluid density. Note that the Dirac delta function is nondimensional. Taking an infinitesimal segment of curve Γ as shown in Fig. 2, we apply on it the Newton's second law nondimensionalized by the same scales as those used to nondimensionalize Eqs. (1) and (2) to obtain

$$\frac{\rho_s}{\rho_f} \frac{\mathrm{d}\vec{V}(\alpha, t)}{\mathrm{d}t} = \vec{F}_f + \vec{F}_o, \tag{12}$$

where $\vec{V}(\alpha, t)$ is the velocity of the segment, $\vec{F}_f$ is the resultant fluid force acting on the segment, and $\vec{F}_o$ is the force generated by the object itself, which can be muscle force or control force. We simply call $\vec{F}_o$ nonfluid force. The fluid force $\vec{F}_f$ is related to the singular force density by

$$\vec{F}_f = -\vec{f}. \tag{13}$$

If an object is deformable and if the force law relates the mechanic force and the deformation of the object is known, the singular force can be readily obtained. The relaxation of a stretched membrane in a fluid is a typical example. If the membrane is elastic, we have $\vec{F}_o = \vec{F}_{\text{solid}}$ with

$$\vec{F}_{\text{solid}} = \frac{\partial T\vec{\tau}}{\partial \alpha}, \tag{14}$$

where tension T is given by

$$T(\alpha, t) = E\left(\frac{J(\alpha, t)}{J_e(\alpha)} - 1\right), \tag{15}$$

in which E is a constant and J_e is J for the unstretched membrane.

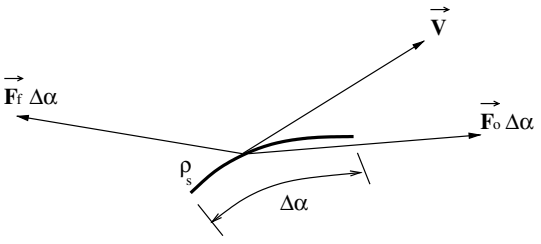


Fig. 2. Surface segment of velocity $\vec{V}$ subject to forces $\vec{F}_f$ and $\vec{F}_o$.

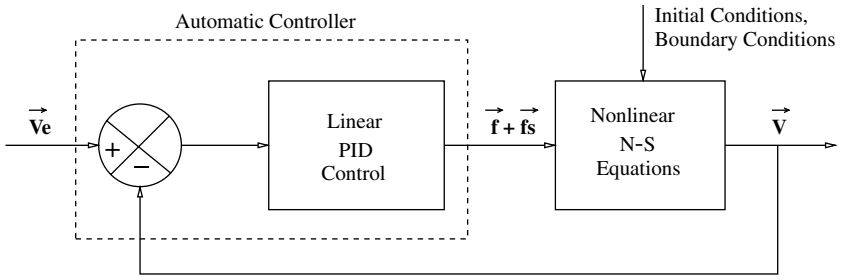


Fig. 3. Block diagram of the feedback control in the force construction for an object in prescribed motion.

If the motion of an object is prescribed, an inverse problem needs to be solved to obtain a singular force density distribution such that the resulting motion of the object matches the prescribed one. Here, we construct the singular force density distribution based on a feedback control. We let $\vec{F}_o$ be $\vec{F}_{\text{control}}$, given by

$$\vec{F}_{\text{control}} = \frac{\rho_m}{\rho_f}\frac{\mathrm{d}(\vec{V}_e - \vec{V})}{\mathrm{d}t} + K_d(\vec{V}_e - \vec{V}) + K_s(\vec{X}_e - \vec{X}), \tag{16}$$

where ρ_m, K_d and K_s are constants, $\vec{V}$ and $\vec{V}_e$ are the simulated and prescribed velocity of the boundary segment, and $\vec{X}$ and $\vec{X}_e$ its simulated and prescribed coordinates. We may also combine a solid model with feedback control, i.e. $\vec{F}_o = \vec{F}_{\text{solid}} + \vec{F}_{\text{control}}$.

When $\vec{F}_o$ is from $\vec{F}_{\text{control}}$ only, after plugging Eq. (13) and (16) into Eq. (12), we obtain

$$K_m\frac{\mathrm{d}\Delta\vec{V}}{\mathrm{d}t} + K_d\Delta\vec{V} + K_s\int\Delta\vec{V}\mathrm{d}t = \vec{f} + \vec{f}_s, \tag{17}$$

where $K_m = \frac{\rho_s+\rho_m}{\rho_f}$, $\Delta\vec{V} = \vec{V}_e - \vec{V}$ and $\vec{f}_s = \frac{\rho_s}{\rho_f}\frac{\mathrm{d}\vec{V}_e}{\mathrm{d}t}$. Thus we have a feedback control system as governed by Eq. (17) and sketched in Fig. 3. This feedback control system has a linear PID (Proportional Integral Derivative) controller and a nonlinear plant operated through the Navier–Stokes equations. If curve Γ is massless ($\rho_s = 0$), and we let $\rho_m = 0$, the linear controller becomes a PI (Proportional Integral) controller with $K_m = 0$ in Eq. (17).

The nonlinear plant makes the analytical design of the PID or PI controller very difficult. To tune the PID or PI controller for reducing the error, $\Delta\vec{V}$, and maintaining numerical stability, we currently resort to a trial and error approach. Our experience indicates that K_m and K_d have to take very small positive values or zero to ensure numerical stability. The main parameter to be tuned in the PID or PI controller is thus K_s. Since the pressure jump across a boundary is subject to an arbitrary constant, we tune K_s based on the order of magnitude analysis for the viscous force. The viscous terms in Eq. (1) have order of 1 in the boundary layer of thickness $\frac{1}{\sqrt{Re}}$ along the boundary. From Section 4.1, we estimate

$$-f_\tau = \left|\frac{1}{Re}\left[\frac{\partial\vec{v}}{\partial\vec{n}}\right]\right| \sim \frac{1}{\sqrt{Re}}.$$

Let the tolerances for $|\int\Delta V_\tau\mathrm{d}t|$ be Δ_s in Eq. (17), where subscript τ denote a tangential component of a vector. We have

$$K_s \sim \frac{1}{\Delta_s\sqrt{Re}}.$$

6. Implementation of the immersed interface method

We have described the three main components of the immersed interface method, which are the singular force density, the jump conditions in terms of the singular force density, and finite difference schemes incorporated with the jump conditions. Next we implement the method in a flow solver based on the MAC scheme [11].

6.1. MAC grid and boundary representation

A MAC grid is a staggered Cartesian grid. We use a uniform MAC grid as sketched in Fig. 4(a) for the finite-difference approximation to governing equations (1)–(3). The center of a cell for pressure p is (i, j), where $i \in \{1, 2, \ldots, N_x\}$ and $j \in \{1, 2, \ldots, N_y\}$. The center of a cell for velocity component u is (I, j) and the center of a cell for velocity component v is (i, J), where (I, J) corresponds to $(i+\frac{1}{2}, j+\frac{1}{2})$, $I \in \{0, 1, \ldots, N_x\}$ and $J \in \{0, 1, \ldots, N_y\}$. The dimensions of a cell are Δx and Δy.

We calculate geometric quantities and the singular force density at the Lagrangian points numbered by index m on curve Γ, as marked by circles in Fig. 4(b), where $m \in \{0, 1, \ldots, N_m\}$. We use periodic cubic splines to interpolate the values of the geometric quantities and the singular force density at the irregular points marked as X in Fig. 4(b), which are the intersections of curve Γ and grid lines. The cost of cubic–spline interpolation is of order $\mathcal{O}(N_m)$. We can then identify finite difference stencils that contain the irregular points, and compute the jump contributions for the finite difference schemes on those stencils.

To move Lagrangian points (X_m, Y_m), we first interpolate the velocity of all the irregular points with an interpolation scheme given in Section 6.4, and then use periodic cubic splines to interpolate velocity (u_m, v_m) at the Lagrangian points. When two irregular points are very close to each other, we use one of them in order to avoid failure of the cubic splines.

6.2. Spatial discretization

With the cell definitions for u, v and p shown in Fig. 5, we can write the second-order central finite difference approximations to Eqs. (1) and (3) in the following form:

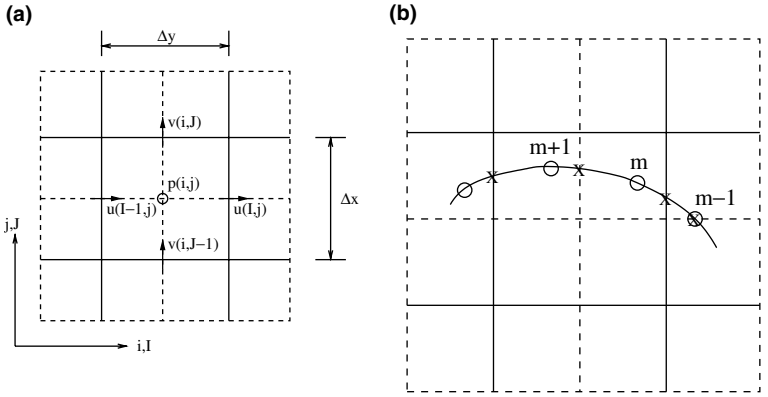


Fig. 4. Discrete representation of the computational domain and the object surface. (a) The MAC grid with staggered arrangement of flow variables; (b) Lagrangian points (open circle) and irregular points (x-mark) on the object surface.

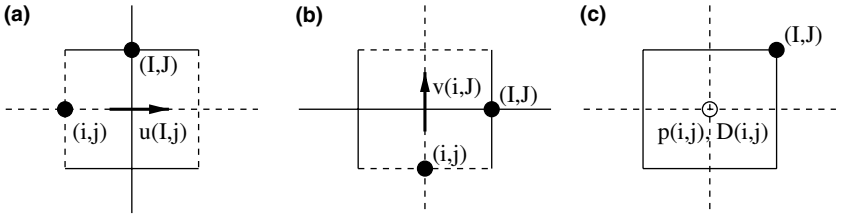


Fig. 5. Cell definitions for (a) u velocity component, (b) v velocity component, and (c) pressure p and velocity divergence D.

$$\left(\frac{\partial u}{\partial t}\right)_{I,j} = C^u_{I,j} - \left(\frac{u^2_{i+1,j} - u^2_{i,j}}{\Delta x} + \frac{u_{I,J}v_{I,J} - u_{I,J-1}v_{I,J-1}}{\Delta y}\right) - \frac{p_{i+1,j} - p_{i,j}}{\Delta x} + \frac{1}{Re}\left(\frac{u_{I+1,j} - 2u_{I,j} + u_{I-1,j}}{\Delta x^2} + \frac{u_{I,j+1} - 2u_{I,j} + u_{I,j-1}}{\Delta y^2}\right), \tag{18}$$

$$\left(\frac{\partial v}{\partial t}\right)_{i,J} = C^v_{i,J} - \left(\frac{u_{I,J}v_{I,J} - u_{I-1,J}v_{I-1,J}}{\Delta x} + \frac{v^2_{i,j+1} - v^2_{i,j}}{\Delta y}\right) - \frac{p_{i,j+1} - p_{i,j}}{\Delta y} + \frac{1}{Re}\left(\frac{v_{i+1,J} - 2v_{i,J} + v_{i-1,J}}{\Delta x^2} + \frac{v_{i,J+1} - 2v_{i,J} + v_{i,J-1}}{\Delta y^2}\right), \tag{19}$$

$$\begin{aligned}\left(\frac{\partial D}{\partial t}\right)_{i,j} = {} & C^p_{i,j} + \frac{p_{i+1,j} - 2p_{i,j} + p_{i-1,j}}{\Delta x^2} + \frac{p_{i,j+1} - 2p_{i,j} + p_{i,j-1}}{\Delta y^2} \\ & - 2\left(\frac{u_{I,j}D_{I,j} - u_{I-1,j}D_{I-1,j}}{\Delta x} + \frac{v_{i,J}D_{i,J} - v_{i,J-1}D_{i,J-1}}{\Delta y}\right) \\ & + \frac{1}{Re}\left(\frac{D_{i+1,j} - 2D_{i,j} + D_{i-1,j}}{\Delta x^2} + \frac{D_{i,j+1} - 2D_{i,j} + D_{i,j-1}}{\Delta y^2}\right) \\ & + 2\left(\frac{u_{I,j} - u_{I-1,j}}{\Delta x}\frac{v_{i,J} - v_{i,J-1}}{\Delta y} - \frac{u_{i,J} - u_{i,J-1}}{\Delta y}\frac{v_{I,j} - v_{I-1,j}}{\Delta x}\right),\end{aligned} \tag{20}$$

where $C^u_{I,j}$, $C^v_{i,J}$ and $C^p_{i,j}$ are the terms due to jump contributions in finite differences, and $D_{i,j}$ is computed as

$$D_{i,j} = \frac{u_{I,j} - u_{I-1,j}}{\Delta x} + \frac{v_{i,J} - v_{i,J-1}}{\Delta y} + C^D_{i,j}, \tag{21}$$

with the jump contribution term denoted by $C^D_{i,j}$. Some of the velocity and divergence values in Eqs. (18)–(20) are not centered in their cells shown in the cell diagram in Fig. 5. We interpolate them from adjacent values. Again, the interpolation schemes are presented later in Section 6.4.

$C^u_{I,j}$, $C^v_{i,J}$, $C^p_{i,j}$ or $C^D_{i,j}$ is nonzero only if the stencil of a finite difference in the corresponding equation contains irregular points. To give an example, we calculate $C^u_{I,j}$ for the case shown in Fig. 6(a). We denote the coordinates of the two irregular points A and B as (X, y_j) and (x_I, Y), respectively. A jump condition at point A is now defined as $[\cdot] = (\cdot)_{X^+} - (\cdot)_{X^-}$, and a jump condition at point B as $[\cdot] = (\cdot)_{Y^+} - (\cdot)_{Y^-}$. $C^u_{I,j}$ is the sum of the jump contribution from each finite difference in Eq. (18). In this case,

- the jump contribution associated with $-\frac{u^2_{i+1,j} - u^2_{i,j}}{\Delta x}$ is

$$\frac{1}{\Delta x}\left(\left[\frac{\partial u^2}{\partial x}\right](x_{i+1} - X) + \frac{1}{2}\left[\frac{\partial^2 u^2}{\partial x^2}\right](x_{i+1} - X)^2\right), \tag{22}$$

- the jump contribution associated with $-\frac{u_{I,J}v_{I,J} - u_{I,J-1}v_{I,J-1}}{\Delta y}$ is

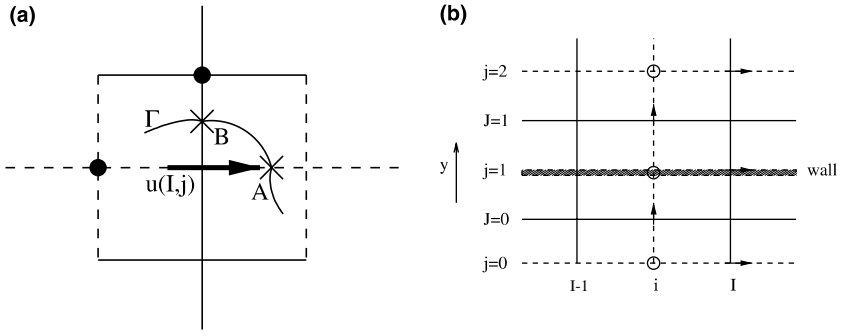


Fig. 6. Schematics of examples for (a) the calculation of jump contribution term C^u_{ij} and (b) the treatment of boundary conditions on a far-field rigid wall.

$$\frac{1}{\Delta y}\left(\left[\frac{\partial uv}{\partial y}\right](y_J - Y) + \frac{1}{2}\left[\frac{\partial^2 uv}{\partial y^2}\right](y_J - Y)^2\right), \tag{23}$$

- the jump contribution associated with $-\frac{p_{i+1,j}-p_{i,j}}{\Delta x}$ is

$$\frac{1}{\Delta x}\left([p] + \left[\frac{\partial p}{\partial x}\right](x_{i+1} - X) + \frac{1}{2}\left[\frac{\partial^2 p}{\partial x^2}\right](x_{i+1} - X)^2\right), \tag{24}$$

- the jump contribution associated with $\frac{1}{Re}\frac{u_{I+1,j}-2u_{I,j}+u_{I-1,j}}{\Delta x^2}$ is

$$-\frac{1}{Re\Delta x^2}\left(\left[\frac{\partial u}{\partial x}\right](x_{I+1} - X) + \frac{1}{2}\left[\frac{\partial^2 u}{\partial x^2}\right](x_{I+1} - X)^2\right), \tag{25}$$

- the jump contribution associated with $\frac{1}{Re}\frac{u_{I,j+1}-2u_{I,j}+u_{I,j-1}}{\Delta y^2}$ is

$$-\frac{1}{Re\Delta y^2}\left(\left[\frac{\partial u}{\partial y}\right](y_{j+1} - Y) + \frac{1}{2}\left[\frac{\partial^2 u}{\partial y^2}\right](y_{j+1} - Y)^2\right). \tag{26}$$

Eqs. (22) and (23) contain terms of the form of $[a \cdot b]$, for example $\left[\frac{\partial u^2}{\partial x}\right] = 2\left[u\frac{\partial u}{\partial x}\right]$, which can be calculated according to Eq. (9).

In our numerical tests in Section 7, we sometimes plot the streamfunction of a velocity field, which is obtained by solving

$$\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} = \omega, \tag{27}$$

where ψ is the streamfunction and ω is the vorticity. By definition, we have

$$u = -\frac{\partial \psi}{\partial y}, \quad v = \frac{\partial \psi}{\partial x}, \tag{28}$$

$$\omega = \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}. \tag{29}$$

Thus we can derive the following jump conditions:

$$[\psi] = 0, \quad \left[\frac{\partial \psi}{\partial x}\right] = 0, \quad \left[\frac{\partial \psi}{\partial y}\right] = 0, \quad \left[\frac{\partial^2 \psi}{\partial x^2}\right] = \left[\frac{\partial v}{\partial x}\right], \quad \left[\frac{\partial^2 \psi}{\partial y^2}\right] = -\left[\frac{\partial u}{\partial y}\right].$$

The central finite difference approximations to Eq. (27) and definition (29) are

$$\frac{\psi_{i+1,j} - 2\psi_{i,j} + \psi_{i-1,j}}{\Delta x^2} + \frac{\psi_{i,j+1} - 2\psi_{i,j} + \psi_{i,j-1}}{\Delta y^2} + C^{\psi}_{i,j} = \omega_{i,j},$$

$$\omega_{i,j} = \frac{v_{I,j} - v_{I-1,j}}{\Delta x} - \frac{u_{i,J} - u_{i,J-1}}{\Delta y} + C^{\omega}_{i,j},$$

where $C^{\psi}_{i,j}$ and $C^{\omega}_{i,j}$ are from the jump contributions in finite differences. They are computed in the similar way as the calculation of $C^{u}_{I,j}$ given in the previous example.

We now discuss our implementation of the no-slip, no-penetration and the pressure boundary conditions on a far-field rigid wall. Referring to Fig. 6(b), we enforce the no-slip and no-penetration boundary conditions as follows:

$$\frac{u(I,0) + u(I,2)}{2} = 0, \quad \frac{v(i,0) + v(i,1)}{2} = 0. \tag{30}$$

With the treatment of $\left(\frac{\partial D}{\partial t}\right)_{i,j}$ as presented in Section 6.3, Eq. (20) for the pressure is a discrete Poisson equation. We currently solve the discrete pressure Poisson equation and streamfunction Poisson equation using FFT, which has cost of order $\mathcal{O}(N_{ij}\ln(N_{ij}))$, where N_{ij} is the number of nodes in a Cartesian grid. Referring to Fig. 6(b), we derive from Eq. (1) the following Neumann boundary condition for the pressure at a rigid wall:

$$\left(\frac{\partial p}{\partial y}\right)_{i,j=1} = \frac{1}{Re}\frac{v_{i,j=2} + v_{i,j=0}}{\Delta y^2}, \tag{31}$$

where $v_{i,j=0}$ is computed based on the divergence free condition at grid node $(i, J = 0)$ as following:

$$\frac{u_{I,J=0} - u_{I-1,J=0}}{\Delta x} + \frac{0 - v_{i,j=0}}{\Delta y} = 0. \tag{32}$$

6.3. Temporal integration

We employ an explicit fourth-order Runge–Kutta scheme in the temporal integration. High-order explicit schemes are appropriate for the moderate Reynolds numbers that we consider here, as shown by Johnston and Liu [12,13] and E and Liu [5]. In the flow regime of moderate to high Reynolds numbers, viscous time step constraint is much less restrictive than the convective one. A high-order temporal integration scheme whose stability region includes a portion of the imaginary axis can ensure stability, and an implicit treatment of viscous terms is not necessary.

In this section, we focus on how to incorporate temporal jump conditions into the fourth-order Runge–Kutta scheme. Due to limited temporal jump conditions, the temporal accuracy is only first-order for a grid point at the instant when the grid point is crossed by an immersed boundary, but the overall temporal accuracy is not affected much since the number of such grid points is much less than the total number of grid points.

We define vectors $\mathbf{q}$ and $\mathbf{R}$ as

$$\mathbf{q} = \{u_{I,j}, v_{i,J}, X_m, Y_m\}, \tag{33}$$

$$\mathbf{R} = \{R^u_{I,j}, R^v_{i,J}, u_m, v_m\}, \tag{34}$$

where $R^u_{I,j}$ and $R^v_{i,J}$ are the right-hand sides of Eqs. (18) and (19), respectively. Then we have the ordinary differential equation as following:

$$\frac{d\mathbf{q}}{dt} = \mathbf{R}, \tag{35}$$

supplemented with Eq. (20), which is rewritten below with subscripts i, j omitted.

$$\frac{\partial D}{\partial t} = \Delta p + R^p. \tag{36}$$

With superscript n denoting a time level and Δt a time step, the sequence of temporal integration of Eq. (35) using the forth-order Runge–Kutta scheme is as follows:

(1) solve pressure $p_1^{n+\frac{1}{2}}$ and then compute $\mathbf{q_1}^{n+\frac{1}{2}}$:

$$\frac{0 - D^n}{\left(\frac{\Delta t}{2}\right)} = \Delta p_1^{n+\frac{1}{2}} + R^p(\mathbf{q}^n),$$

$$\mathbf{q_1}^{n+\frac{1}{2}} = \mathbf{q}^n + \frac{\Delta t}{2}\left(\mathbf{R}(\mathbf{q}^n, p_1^{n+\frac{1}{2}}) + \mathbf{c_1}^n\right);$$

(2) solve pressure $p_2^{n+\frac{1}{2}}$ and then compute $\mathbf{q_2}^{n+\frac{1}{2}}$:

$$\frac{0 - D^n}{\left(\frac{\Delta t}{2}\right)} = \Delta p_2^{n+\frac{1}{2}} + R^p(\mathbf{q_1}^{n+\frac{1}{2}}),$$

$$\mathbf{q_2}^{n+\frac{1}{2}} = \mathbf{q}^n + \frac{\Delta t}{2}\left(\mathbf{R}(\mathbf{q_1}^{n+\frac{1}{2}}, p_2^{n+\frac{1}{2}}) + \mathbf{c_2}^n\right);$$

(3) solve pressure p_3^{n+1} and then compute $\mathbf{q_3}^{n+1}$:

$$\frac{0 - D^n}{\Delta t} = \Delta p_3^{n+1} + R^p(\mathbf{q_2}^{n+\frac{1}{2}}),$$

$$\mathbf{q_3}^{n+1} = \mathbf{q}^n + \Delta t\left(\mathbf{R}(\mathbf{q_2}^{n+\frac{1}{2}}, p_3^{n+1}) + \mathbf{c_3}^n\right);$$

(4) solve pressure p_4^{n+1} and then compute $\mathbf{q}^{n+1}$:

$$\frac{0-D^n}{\Delta t} = \Delta p_4^{n+1} + R^p(\mathbf{q}_3^n),$$

$$\mathbf{q}^{n+1} = \mathbf{q}^n + \Delta t\left(\frac{1}{6}\left(\mathbf{R}(\mathbf{q}^n, p_1^{n+\frac{1}{2}}) + 2\mathbf{R}(\mathbf{q}_1^{n+\frac{1}{2}}, p_2^{n+\frac{1}{2}}) + 2\mathbf{R}(\mathbf{q}_2^{n+\frac{1}{2}}, p_3^{n+1}) + \mathbf{R}(\mathbf{q}_3^{n+1}, p_4^{n+1})\right) + \mathbf{c}_4^n\right);$$

where $\mathbf{c}_1^n$, $\mathbf{c}_2^n$, $\mathbf{c}_3^n$ and $\mathbf{c}_4^n$ are jump contributions for the temporal discretization of $\frac{d\mathbf{q}}{dt}$ in Runge–Kutta sub-steps. They are nonzero only at those grid nodes which are passed by a boundary. They are computed as follows:

- treat Runge–Kutta sub-step (1) as a forward finite difference in interval $\frac{\Delta t}{2}$, and calculate $\mathbf{c}_1^n$ for a grid node if the curve Γ passes the grid node at time t^* between time t^n and $t^{n+\frac{1}{2}}$:

$$\mathbf{c}_1^n = \frac{2}{\Delta t}\left([[\mathbf{q}]] + \left[\left[\frac{\partial \mathbf{q}}{\partial t}\right]\right]\left(t^{n+\frac{1}{2}} - t^*\right)\right);$$

- treat Runge–Kutta sub-step (2) as a backward finite difference in interval $\frac{\Delta t}{2}$, and calculate $\mathbf{c}_2^n$ for a grid node if the curve Γ passes the grid node at time t^* between time t^n and $t^{n+\frac{1}{2}}$:

$$\mathbf{c}_2^n = \frac{2}{\Delta t}\left([[\mathbf{q}]] + \left[\left[\frac{\partial \mathbf{q}}{\partial t}\right]\right](t^n - t^*)\right);$$

- treat Runge–Kutta sub-step (3) as a central finite difference in interval Δt, and calculate $\mathbf{c}_3^n$ for a grid node if the curve Γ passes the grid node at time t^* between time t^n and $t^{n+\frac{1}{2}}$:

$$\mathbf{c}_3^n = \frac{1}{\Delta t}\left([[\mathbf{q}]] + \left[\left[\frac{\partial \mathbf{q}}{\partial t}\right]\right](t^{n+1} - t^*)\right),$$

or if the curve Γ passes the grid node at time t^* between time $t^{n+\frac{1}{2}}$ and t^{n+1}:

$$\mathbf{c}_3^n = \frac{1}{\Delta t}\left([[\mathbf{q}]] + \left[\left[\frac{\partial \mathbf{q}}{\partial t}\right]\right](t^n - t^*)\right);$$

- treat Runge–Kutta sub-step (4) as a combination of one forward, one backward, and two central finite differences in interval Δt, and calculate $\mathbf{c}_4^n$ for a grid node if the curve Γ passes the grid node at time t^* between time t^n and $t^{n+\frac{1}{2}}$:

$$\mathbf{c}_4^n = \frac{1}{6\Delta t}\left([[\mathbf{q}]] + \left[\left[\frac{\partial \mathbf{q}}{\partial t}\right]\right](t^{n+1} - t^*)\right) + \frac{2}{3\Delta t}\left([[\mathbf{q}]] + \left[\left[\frac{\partial \mathbf{q}}{\partial t}\right]\right](t^{n+1} - t^*)\right) + \frac{1}{6\Delta t}\left([[\mathbf{q}]] + \left[\left[\frac{\partial \mathbf{q}}{\partial t}\right]\right](t^n - t^*)\right),$$

or if the curve Γ passes the grid node at time t^* between time $t^{n+\frac{1}{2}}$ and t^{n+1}:

$$\mathbf{c}_4^n = \frac{1}{6\Delta t}\left([[\mathbf{q}]] + \left[\left[\frac{\partial \mathbf{q}}{\partial t}\right]\right](t^{n+1} - t^*)\right) + \frac{2}{3\Delta t}\left([[\mathbf{q}]] + \left[\left[\frac{\partial \mathbf{q}}{\partial t}\right]\right](t^n - t^*)\right) + \frac{1}{6\Delta t}\left([[\mathbf{q}]] + \left[\left[\frac{\partial \mathbf{q}}{\partial t}\right]\right](t^n - t^*)\right).$$

Time t^* and a jump condition $[[\cdot]]$ at time t^*, which is define as $[[\cdot]] = (\cdot)_{t^{*+}} - (\cdot)_{t^{*-}}$, are obtained by interpolation using known information at time level n, $n+\frac{1}{2}$ or $n+1$. We present the interpolation schemes in Section 6.4.

6.4. Filtering and interpolation

For a moving object problem, Lagrangian points on the object boundary are moved using the velocity interpolated from the surrounding fluid velocity on Cartesian grid points. The boundary shape thus contains the irregular truncation errors of the interpolation. Since the jump conditions involve the differentiation of geometric quantities and the singular force density along a boundary, it is important to maintain the shape smoothness of the object. We employ Fourier filtering to smooth the shape.

It is necessary to interpolate quantities such as $a^+(b^+)$, $a^-(b^-)$, $u_{i,j}(v_{i,j})$, $u_{I,J}(v_{I,J})$, $u_{i,J}(v_{I,j})$, t^*, $[[\cdot]](t^*)$ and the velocity at irregular points. We categorize the interpolations into three scenarios, as shown in Fig. 7.

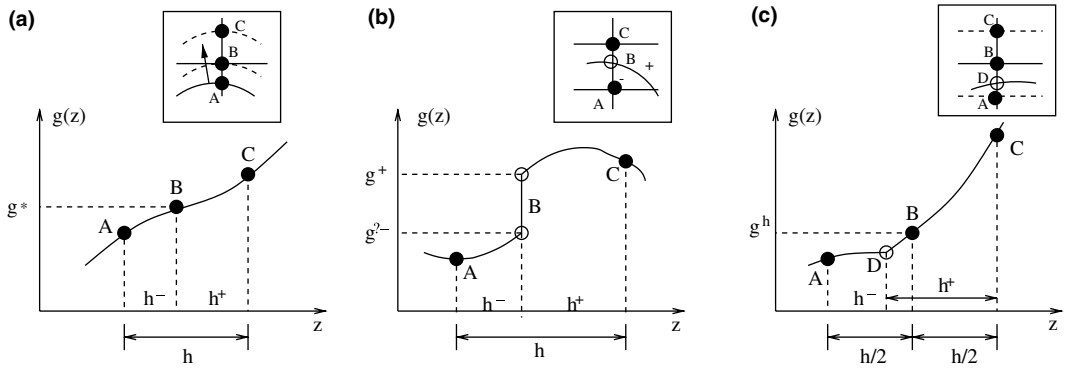


Fig. 7. Interpolation scenarios associated with (a) a smooth function, (b) a discontinuous function at its discontinuity point, and (c) a piecewise smooth function.

To determine time t^* when an interface crosses a grid point, we monitor the position of a irregular point along a grid line, i.e. $A \to B \to C$ in the inset of Fig. 7(a). Since the time is a smooth function of the changing coordinate of the irregular point, the interpolate scheme is given by

$$g^* = \frac{h^- g_C + h^+ g_A}{h} + \mathrm{O}(h^2), \tag{37}$$

where g is time and z is the changing coordinate in Fig. 7(a) for the current case. It can also be shown that jump condition $[[\cdot]](t)$ is a smooth function of time t along the grid line, and therefore the above interpolation scheme applies to $[[\cdot]](t^*)$.

The spatial interpolation for a^+ (b^+), a^- (b^-) and the velocity at irregular points is described in Fig. 7(b). The interpolation scheme is

$$g^+ = \frac{h^- g_C + h^+ g_A}{h} - \frac{h^-[g_B]}{h} - \frac{h^+ h^-}{h^2}\left[\frac{\partial g_B}{\partial z}\right] + \mathrm{O}(h^2), \tag{38}$$

$$g^- = \frac{h^- g_C + h^+ g_A}{h} + \frac{h^+[g_B]}{h} - \frac{h^+ h^-}{h}\left[\frac{\partial g_B}{\partial z}\right] + \mathrm{O}(h^2), \tag{39}$$

where $[g_B] = 0$ and $g^+ = g^-$ for the velocity interpolation since the velocity is continuous.

The interpolation for $u_{i,j}$ $(v_{i,j})$, $u_{I,J}$ $(v_{I,J})$, $u_{i,J}$ and $v_{I,j}$ is described in Fig. 7(c). When point D falls between points A and B as in Fig. 7(c), we have

$$g_B = \frac{g_A + g_C}{2} + \frac{1}{2}[g_D] - \frac{1}{2}\left[\frac{\partial g_D}{\partial z}\right]h^- + \frac{1}{4}\left[\frac{\partial^2 g_D}{\partial z^2}\right](h^-)^2 + \mathrm{O}(h^2). \tag{40}$$

When point D falls between points B and C, we have

$$g_B = \frac{g_A + g_C}{2} - \frac{1}{2}[g_D] - \frac{1}{2}\left[\frac{\partial g_D}{\partial z}\right]h^+ - \frac{1}{4}\left[\frac{\partial^2 g_D}{\partial z^2}\right](h^+)^2 + \mathrm{O}(h^2). \tag{41}$$

6.5. Force calculation

We here use C_x and C_y respectively to denote the x and y components of the nondimensional force applied by the fluid to the object. They are nondimensionalized by half of the pressure scale in Eq. (1). Referring to Fig. 1, they can be calculated as follows:

$$C_x = 2\oint_\Gamma \left(-p^+ n_x + \frac{1}{Re}\left(\frac{\partial u}{\partial \mathbf{n}}\right)^+\right)\mathrm{d}\Gamma \tag{42}$$

$$= -2\oint_\Gamma f^x \mathrm{d}\alpha + 2\oint_\Gamma \left(-p^- n_x + \frac{1}{Re}\left(\frac{\partial u}{\partial \mathbf{n}}\right)^-\right)\mathrm{d}\Gamma, \tag{43}$$

$$C_y = 2\oint_\Gamma \left(-p^+ n_y + \frac{1}{Re}\left(\frac{\partial v}{\partial \mathbf{n}}\right)^+\right)\mathrm{d}\Gamma \tag{44}$$

$$= -2\oint_\Gamma f^y \mathrm{d}\alpha + 2\oint_\Gamma \left(-p^- n_y + \frac{1}{Re}\left(\frac{\partial v}{\partial \mathbf{n}}\right)^-\right)\mathrm{d}\Gamma. \tag{45}$$

For a centrally symmetric boundary, decomposing its prescribed motion to the superposition of a translation and a rotation around the geometric center, we find

$$\oint_\Gamma (-p^- n_x)\mathrm{d}\Gamma = S\frac{\mathrm{d}u_T}{\mathrm{d}t}, \quad \oint_\Gamma (-p^- n_y)\mathrm{d}\Gamma = S\frac{\mathrm{d}v_T}{\mathrm{d}t},$$

$$\oint_\Gamma \left(\frac{\partial u}{\partial \mathbf{n}}\right)^- \mathrm{d}\Gamma = 0, \quad \oint_\Gamma \left(\frac{\partial v}{\partial \mathbf{n}}\right)^- \mathrm{d}\Gamma = 0,$$

where (u_T, v_T) are the translational velocity and S is the area of the object. So C_x and C_y in Eqs. (43) and (45) can be simplified to

$$C_x = -2\oint_\Gamma f^x \mathrm{d}\alpha + 2S\frac{\mathrm{d}u_T}{\mathrm{d}t}, \quad C_y = -2\oint_\Gamma f^y \mathrm{d}\alpha + 2S\frac{\mathrm{d}v_T}{\mathrm{d}t}. \tag{46}$$

7. Results

In this section, we test the method in several distinct fluid–structure interactions, including flows of know analytical solutions, flow induced by a relaxing balloon, flow past a cylinder, a flapper, and multiple cylinders. In particular, we investigate the spatial and temporal convergence rates of the method, and demonstrate the robustness and efficiency of the method in handling single or multiple boundaries prescribed with known motion or driven by a force law. The Reynolds number, Re, of the flows ranges from 1 to 200.

In these tests, object boundaries are massless, i.e. $\rho_s = 0$ in Eqs. (11) and (12), and we have $\mathbf{f} = \mathbf{F}_o$. We let $\rho_m = 0$ in Eq. (16). So $K_m = 0$ and $\mathbf{f}_s = 0$ in Eq. (17) when the feedback control is used to construct force $\mathbf{F}_o$.

7.1. Flow without immersed boundaries

As a first test, we simulate a flow without any immersed boundaries to check the flow solver. We consider the following flow which satisfies the Navier–Stokes equations exactly:

$$u = -\frac{\sqrt[3]{Re}}{4}\exp\left(-\frac{17}{256\sqrt[3]{Re}}t\right)\cos\left(\frac{\sqrt[3]{Re}}{16}x\right)\cos\left(\frac{\sqrt[3]{Re}}{4}y\right),$$

$$v = \frac{\sqrt[3]{Re}}{16}\exp\left(-\frac{17}{256\sqrt[3]{Re}}t\right)\sin\left(\frac{\sqrt[3]{Re}}{16}x\right)\sin\left(\frac{\sqrt[3]{Re}}{4}y\right),$$

$$p = \frac{\sqrt[3]{Re^2}}{1024}\exp\left(-\frac{34}{256\sqrt[3]{Re}}t\right)\left(17\sin\left(\frac{\sqrt[3]{Re}}{16}x + \frac{\sqrt[3]{Re}}{4}y\right)\sin\left(\frac{\sqrt[3]{Re}}{16}x - \frac{\sqrt[3]{Re}}{4}y\right)\right.$$
$$\left. -15\cos\left(\frac{\sqrt[3]{Re}}{16}x + \frac{\sqrt[3]{Re}}{4}y\right)\cos\left(\frac{\sqrt[3]{Re}}{16}x - \frac{\sqrt[3]{Re}}{4}y\right)\right).$$

We simulated the flow at $Re = 1$ in the domain defined by $-\frac{16\pi}{\sqrt[3]{Re}} \leqslant x \leqslant \frac{16\pi}{\sqrt[3]{Re}}$ and $-\frac{4\pi}{\sqrt[3]{Re}} \leqslant y \leqslant \frac{4\pi}{\sqrt[3]{Re}}$. Periodic boundary conditions were used in this test.

We run the simulation up to time $t = 10$ with time step $\Delta t = 0.01$ and different spatial resolutions to check the spatial convergence rate. The infinity norm of numerical error based on the analytical solution is given in Table 1, where the order of accuracy is defined as

Table 1
Spatial convergence analysis for the flow solver without immersed boundaries

$N_x \times N_y$	$\|u\|_\infty$	Order	$\|v\|_\infty$	Order	$\|p\|_\infty$	Order
16×16	1.40×10^{-3}		3.49×10^{-4}		4.04×10^{-5}	
32×32	3.47×10^{-4}	2.01	8.66×10^{-5}	2.01	1.03×10^{-5}	1.97
64×64	8.65×10^{-5}	2.00	2.16×10^{-5}	2.00	2.57×10^{-6}	2.00
128×128	2.16×10^{-5}	2.00	5.40×10^{-6}	2.00	6.43×10^{-7}	2.00

The infinity norm is based on the analytical solution.

$$\text{order} = \log_2\left(\frac{\|\cdot\|_\infty}{\|\cdot\cdot\|_\infty}\right),$$

where $\|\cdot\cdot\|_\infty$ is the infinity norm of a field with a resolution twice the resolution of a field associated with infinity norm $\|\cdot\|_\infty$. Clearly, second-order spatial convergence rate is achieved, as expected from the central finite difference schemes in space.

We fix $t = 10$ and $N_x \times N_y = 32 \times 32$, but use different time steps to check the temporal convergence rate. We compute a reference solution using a small time step, $\Delta t = 0.001$, and subtract the reference solution from the other numerical solutions to cancel out the error due to spatial discretization. The infinity norm of numerical error based on the reference solution is given in Table 2. Only first-order temporal convergence rate is achieved, which is lower than the expected fourth-order convergence rate of the Runge–Kutta scheme. This is due to the numerical treatment of the temporal derivative of the divergence, $\frac{\partial D}{\partial t}$, in pressure equation (20), where we set divergence D at time levels $n + \frac{1}{2}$ and $n + 1$ to be zero, which does not follow the error cancellation mechanism in the Runge–Kutta scheme.

If we discard term $\frac{\partial D}{\partial t}$ in Eq. (20), we can achieve nearly fourth-order temporal convergence rate, as seen from Table 3. The simulation is run to $t = 25.6$, and the reference solution is computed with $\Delta t = 0.0005$ and $N_x \times N_y = 32 \times 32$. However, keeping $\frac{\partial D}{\partial t}$ in pressure equation (20) improves the divergence-free condition, as indicated by the comparisons in Table 4. In Table 4, the simulation was run to $t = 10$ with $\Delta t = 0.01$ and

Table 2
Temporal convergence analysis for the flow solver without immersed boundaries

Δt	$\|u\|_\infty$	Order	$\|v\|_\infty$	Order	$\|p\|_\infty$	Order
0.01	3.28×10^{-8}		1.70×10^{-8}		2.92×10^{-8}	
0.02	6.92×10^{-8}	1.08	3.59×10^{-8}	1.08	6.17×10^{-8}	1.08
0.04	1.43×10^{-7}	1.05	7.39×10^{-8}	1.04	1.27×10^{-7}	1.04
0.08	2.90×10^{-7}	1.02	1.51×10^{-7}	1.03	2.59×10^{-7}	1.03

Temporal divergence term $\frac{\partial D}{\partial t}$ in pressure equation (3) is included. The infinity norm is based on a reference solution computed with $\Delta t = 0.001$.

Table 3
Temporal convergence analysis for the flow solver without immersed boundaries

Δt	$\|u\|_\infty$	Order	$\|v\|_\infty$	Order	$\|p\|_\infty$	Order
0.04	5.84×10^{-14}		1.21×10^{-14}		5.10×10^{-14}	
0.08	9.71×10^{-13}	4.06	1.19×10^{-13}	3.30	1.52×10^{-13}	1.58
0.16	1.75×10^{-11}	4.17	1.91×10^{-12}	4.00	2.55×10^{-12}	4.07
0.32	4.08×10^{-10}	4.54	3.08×10^{-11}	4.01	4.27×10^{-11}	4.07

Temporal divergence term $\frac{\partial D}{\partial t}$ in pressure equation (3) is not included. The infinity norm is based on a reference solution.

Table 4
Effect of temporal divergence term $\frac{\partial D}{\partial t}$ in pressure equation (3) on numerical accuracy

	$\|u\|_\infty$	$\|v\|_\infty$	$\|p\|_\infty$	$\|D\|_\infty$
Keep $\frac{\partial D}{\partial t}$	3.46×10^{-4}	8.66×10^{-5}	1.03×10^{-5}	1.25×10^{-8}
Discard $\frac{\partial D}{\partial t}$	4.71×10^{-4}	2.22×10^{-4}	7.46×10^{-4}	4.13×10^{-4}

$N_x \times N_y = 32 \times 32$. From Table 4, we note that the accuracy based on the analytical solution is about the same with or without $\frac{\partial D}{\partial t}$, which indicates that the error is dominated by the spatial error instead of the temporal one even at this large $\Delta t = 0.01$. In our later simulations, we have about the same spatial resolution as the current case but smaller Δt. So we keep $\frac{\partial D}{\partial t}$ to better enforce the divergence-free condition without losing the accuracy.

7.2. Taylor–Couette flow

We next consider Taylor–Couette flow between two rotating and translating concentric cylinders. The geometry of the flow is shown in Fig. 8, with $r_1 = 0.5$, $r_2 = 1$, $\Omega_1 = 1$ and $\Omega_2 = -1$. The Reynolds number based on the radius and the angular velocity of the outer cylinder is $Re = \frac{|\Omega_2| r_2^2}{\nu} = 10$. To test moving boundaries crossing the Cartesian grid, we allow the center of the cylinders to oscillate according to

$$x_c = d\sin(t), \quad y_c = d\sin(t),$$

where d is a constant. The analytical solution to the flow between the two cylinders is given by

$$u = d \cdot \cos(t) - \left(A + \frac{B}{r^2}\right)(y - y_c),$$
$$v = d \cdot \cos(t) + \left(A + \frac{B}{r^2}\right)(x - x_c),$$
$$p = d \cdot \sin(t) \cdot (x + y) + \frac{A^2 r^2}{2} - \frac{B^2}{2r^2} + AB \cdot \ln(r^2),$$

where $A = \frac{\Omega_2 r_2^2 - \Omega_1 r_1^2}{r_2^2 - r_1^2}$, $B = \frac{(\Omega_1 - \Omega_2) r_1^2 r_2^2}{r_2^2 - r_1^2}$, $r = \sqrt{(x - x_c)^2 + (y - y_c)^2}$. The analytical solution to the flow inside the inner cylinder is

$$u = d \cdot \cos(t) - \Omega_1 (y - y_c),$$
$$v = d \cdot \cos(t) + \Omega_1 (x - x_c),$$
$$p = d \cdot \sin(t) \cdot (x + y) + \frac{\Omega_1^2 r^2}{2}.$$

We use periodic boundary conditions at the far-field boundaries. The nonfluid force model for both cylinders is given by

$$\mathbf{F}_o = K_s(\mathbf{X}_e - \mathbf{X}), \tag{47}$$

with $K_s = 1000$.

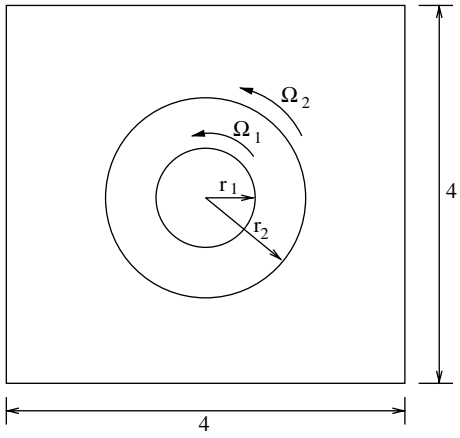


Fig. 8. Geometry of flow between two rotating concentric cylinders.

7.2.1. d = 0

In the steady state, the normal force density, f_n, is zero and the tangential force density, f_τ, is a constant at the inner cylinder. Referring to the jump conditions in Section 4, we know the jump conditions across the inner cylinder for all the first-order and second-order velocity derivatives and all the second-order pressure derivatives are nonzero. Therefore the simulation of this case is a good test to check the spatial convergence rate of the immersed interface method. In particular, we look at the infinity error norm to check the simulation accuracy locally. We use a very small time step, $\Delta t = 0.0001$, to ensure the temporal discretization error is negligible compared with the spatial one. The results are given in Table 5, which indicate second-order spatial convergence rate for both the velocity and the pressure.

As indicated by Lemma 2, the discretization of the Laplace operator near the cylinders in Eqs. (1) and (3) is only first-order accurate because only limited jump conditions are used and the jump conditions for velocity and pressure derivatives of higher orders are nonzero. Interestingly, we still achieve second-order accuracy of the flow field even near the cylinders. The same phenomenon was noted by Li and Lai in their simulation [19].

We calculate jump conditions across the inner cylinder surface based on the analytical solutions, and compare them with those computed from the formula given in Section 4. Fig. 9 shows the comparison for $[\frac{\partial^2 u}{\partial x^2}]$ with $[\cdot] := (\cdot)_{X^+} - (\cdot)_{X^-}$, and $[\frac{\partial^2 p}{\partial y^2}]$ with $[\cdot] := (\cdot)_{Y^+} - (\cdot)_{Y^-}$, indicating very good agreement. For these comparisons, the spatial resolution of the simulation is $N_x \times N_y \times N_m = 64 \times 64 \times 128$.

7.2.2. d = 0.5

In this case, the velocity across the two cylinders are not smooth, which causes jumps of temporal velocity derivatives and therefore jump contributions in temporal velocity discretizations for a grid point when it is crossed by the cylinders. Thus this case provides a test of the effect of the temporal jump contributions on the temporal accuracy.

We first run simulations with the temporal jump contributions up to $t = 10$ with a fixed spatial resolution, $N_x \times N_y \times N_m = 128 \times 128 \times 256$. We compute a reference solution using a very small time step, $\Delta t = 5 \times 10^{-5}$, and subtract the reference solution from the other numerical solutions to cancel out spatial discretization error

Table 5
Spatial convergence analysis for steady Taylor–Couette flow

$N_x \times N_y$, N_m	$\|u\|_\infty$	Order	$\|v\|_\infty$	Order	$\|p\|_\infty$	Order
32×32, 64	2.09×10^{-2}		1.65×10^{-2}		4.26×10^{-2}	
64×64, 128	6.03×10^{-3}	1.79	5.33×10^{-3}	1.63	9.18×10^{-3}	2.21
128×128, 256	1.56×10^{-3}	1.95	1.35×10^{-3}	1.98	2.34×10^{-3}	1.97
256×256, 512	4.12×10^{-4}	1.92	4.23×10^{-4}	1.67	4.68×10^{-4}	2.32

The infinity norm is based on the analytical flow solution.

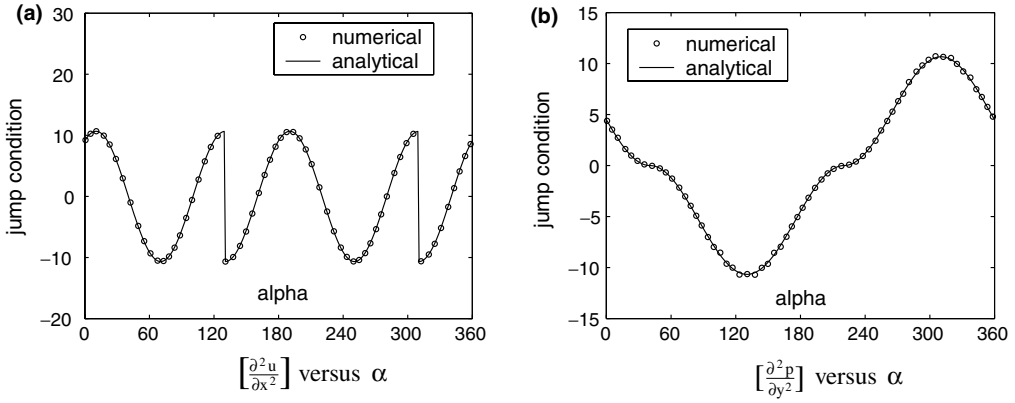


Fig. 9. Comparisons between analytical and numerical results for jump conditions.

Table 6
Temporal convergence analysis for oscillating Taylor–Couette flow

Δt	$\|u\|_\infty$	Order	$\|v\|_\infty$	Order	$\|p\|_\infty$	Order
8×10^{-4}	1.01×10^{-4}		1.32×10^{-4}		9.92×10^{-3}	
4×10^{-4}	4.83×10^{-5}	1.06	5.94×10^{-5}	1.15	8.70×10^{-3}	0.19
2×10^{-4}	3.06×10^{-5}	0.67	2.17×10^{-5}	1.45	4.76×10^{-3}	0.87
1×10^{-4}	6.96×10^{-6}	2.14	7.67×10^{-6}	1.50	1.00×10^{-3}	2.25

Temporal jump contributions are included. The infinity norm is based on a reference solution.

Table 7
Temporal convergence analysis for oscillating Taylor–Couette flow

Δt	$\|u\|_\infty$	Order	$\|v\|_\infty$	Order	$\|p\|_\infty$	Order
8×10^{-4}	9.68×10^{-5}		1.32×10^{-4}		9.71×10^{-3}	
4×10^{-4}	5.20×10^{-5}	0.90	6.09×10^{-5}	1.12	8.69×10^{-3}	0.16
2×10^{-4}	2.91×10^{-5}	0.84	1.58×10^{-5}	1.94	4.76×10^{-3}	0.87
1×10^{-4}	7.26×10^{-6}	2.00	6.00×10^{-6}	1.40	1.01×10^{-3}	2.24

Temporal jump contributions are not included. The infinity norm is based on a reference solution.

and obtain temporal error. The infinity norm of the temporal error is shown in Table 6. We then run another set of simulations without the temporal jump contributions. The results are provided in Table 7.

Tables 6 and 7 indicate that the temporal convergence rates with and without the temporal contributions are about the same. Both are nearly first-order instead of fourth-order because of the numerical treatment to the temporal divergence term in the pressure equation (see Section 7.1). In addition, the inclusion of the temporal jump contributions has negligible effect on the local and overall accuracy of the numerical solutions based on the analytical solution, as indicated by Tables 8 and 9.

7.3. Flow induced by a relaxing balloon

In this test, we compute the moving boundary problem considered by Li and Lai [19], where a 2D distorted pressurized balloon immersed in an incompressible fluid relaxes to its circular equilibrium shape. The initial velocity and the pressure are set zero, and the only driving force is the balloon tension. The fluid flow and the balloon motion are fully coupled. At equilibrium, the velocity is zero and the pressure is piecewise constant inside and outside the balloon with a jump across the balloon.

The initial shape of the distorted balloon expressed in the cylindrical coordinates (r, θ) is

$$r(\theta) = r_0(1 + \epsilon\sin(\kappa\theta)), \quad 0 \leqslant \theta \leqslant 2\pi,$$

where r_0, ϵ and κ are constants, and κ is an integer. The normal and the tangential force densities of the driving force are

$$f_n = E \cdot k_c, \quad f_\tau = 0,$$

where E is a constant and k_c the curvature. At equilibrium, the radius of the balloon, r_e, is

$$r_e = r_0\sqrt{1 + 0.5\epsilon^2},$$

and the pressure jump across the balloon is E/r_e. The area of the balloon is conserved and it is equal to πr_e^2.

We first run the test with the same parameter values as used by Li and Lai (after correcting some typo errors in their paper [20][1]). They are $r_0 = 0.5$, $\epsilon = 0.4$, $\kappa = 5$, $E = 0.05$ and $Re = 1$. Here Reynolds number $Re = 1$ corresponds to viscosity $\mu = 1$. The initial ($t = 0$) and the equilibrium ($t = \infty$) balloon shapes in the computational domain are given in Fig. 10. The boundaries of the computational domain are rigid walls. We simulate the case up to $t = 98$ with $N_x \times N_y \times N_m = 64 \times 64 \times 128$ and $\Delta t = 7 \times 10^{-5}$.

[1] The initial interface should be $r(\theta) = r_0 + \varepsilon\sin(k\theta)$ with $\varepsilon = 0.2$. The value of μ should be 1 instead of 0.1 [20].

Table 8
Numerical accuracy with and without temporal jump contributions

	$\|u\|_\infty$	$\|v\|_\infty$	$\|p\|_\infty$
With temporal jump contributions	2.04×10^{-2}	2.01×10^{-2}	3.48×10^{-2}
Without temporal jump contributions	2.04×10^{-2}	2.01×10^{-2}	3.48×10^{-2}

The infinity norm is based on the analytical solution.

Table 9
Numerical accuracy with and without temporal jump contributions

	$\|u\|_2$	$\|v\|_2$	$\|p\|_2$
With temporal jump contributions	6.94×10^{-2}	1.33×10^{-1}	9.26×10^{-1}
Without temporal jump contributions	6.94×10^{-2}	1.33×10^{-1}	9.26×10^{-1}

The 2-norm is based on the analytical solution.

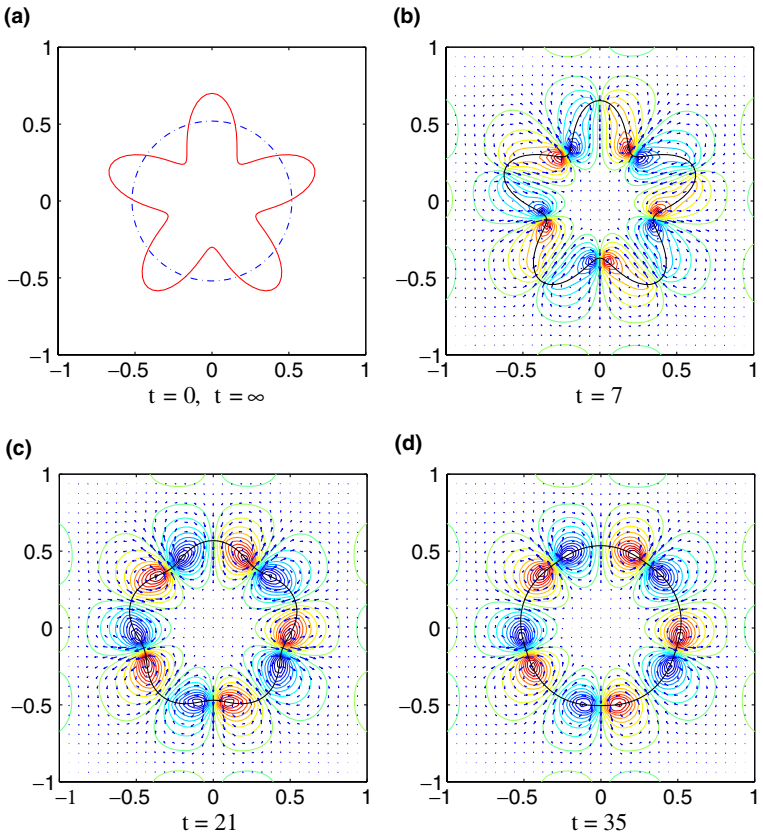


Fig. 10. Numerical evolution of the balloon shape and the vorticity and the velocity fields for flow induced by the relaxation of a distorted balloon.

Fig. 10 plots the balloon shapes, the vorticity contours and the velocity vectors at time $t = 7$, 21 and 35. The balloon shapes at these time instants are very close to those obtained by Li and Lai [19]. We regard the balloon is near equilibrium at time $t = 98$. The pressure at time $t = 98$ is plotted in Fig. 11. The conservation of the area of the balloon is checked in Fig. 12 for $Re = 1$ and $Re = 100$, which indicates very little leakage, about 0.1%. The simulation for $Re = 100$ is run with $\Delta t = 0.0001$ and the same spatial resolution as $Re = 1$. Fig. 13 plots the temporal variation of the balloon radius at $\theta = \pi/10$ at $Re = 1$ and $Re = 100$. We can observe at $Re = 100$ the fast relaxation of the balloon through damped oscillations, which are absent at $Re = 1$.

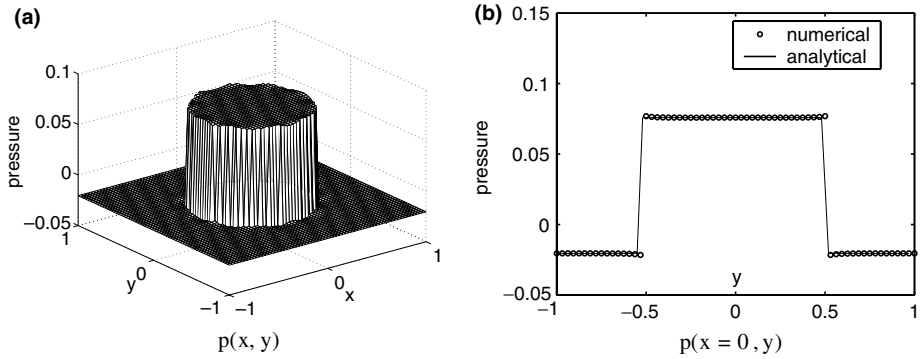


Fig. 11. Equilibrium pressure: (a) pressure field, (b) comparison of pressure distribution along y axis at $x = 0$ between theoretical and numerical results.

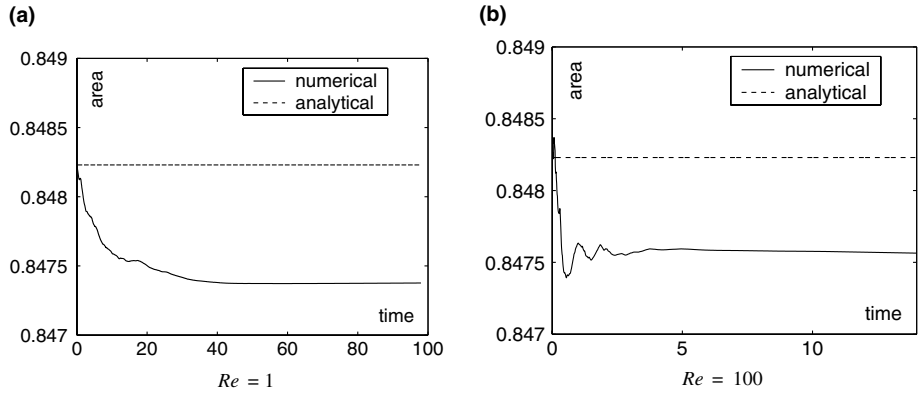


Fig. 12. The temporal evolution of the area enclosed by the balloon.

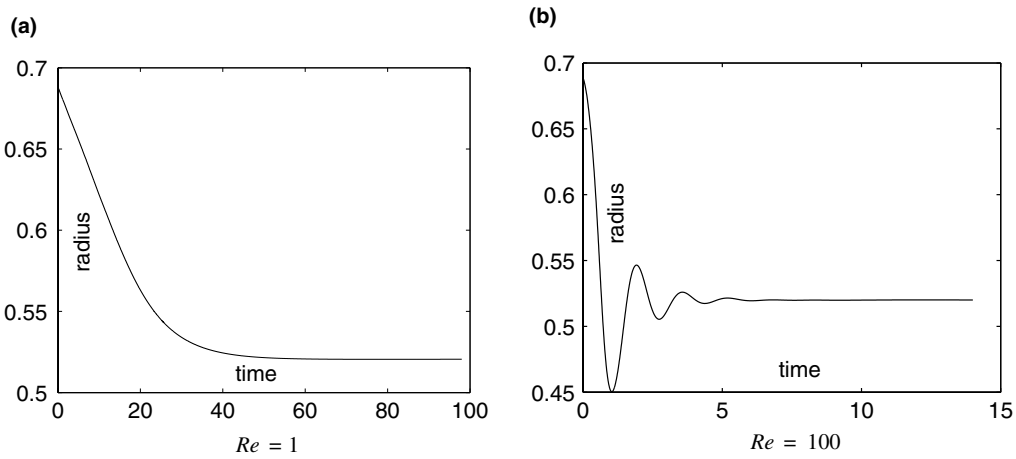


Fig. 13. The temporal evolution of the radius of a Lagrangian point on the balloon.

We perform spatial convergence analysis for the flow field of $Re = 10$ at time $t = 2$. In this analysis, the time step is $\Delta t = 0.0001$. We calculate the error between a simulated flow field and the one obtained by the fine grid $N_x \times N_y \times N_m = 256 \times 256 \times 512$. Table 10 give the infinity norm of the error against the spatial resolution. Near second-order convergence rate for both velocity components is observed. When the spatial convergence

Table 10
Spatial convergence analysis for the flow induced by a relaxing balloon

$N_x \times N_y$, N_m	$\|u\|_\infty$	Order	$\|v\|_\infty$	Order
16×16, 32	6.14×10^{-2}		8.16×10^{-2}	
32×32, 64	1.49×10^{-2}	2.04	2.91×10^{-2}	1.49
64×64, 128	3.56×10^{-3}	2.07	7.25×10^{-3}	2.00
128×128, 256	7.33×10^{-4}	2.28	7.80×10^{-4}	3.22

The infinity norm is based on a reference solution.

rate for the pressure is calculated, special care is needed near the interface. At different spatial resolutions, a grid point can be inside the balloon at one grid level and outside at another, such as the middle grid point sketched in Fig. 14. We thus discard this type of points to exclude the pressure jump in the calculation of the infinity norm. The same treatment is not necessary for the velocity since the velocity is continuous across the interface. We denote the modified infinity norms for u, v and p by $\|u_0\|_\infty$, $\|v_0\|_\infty$ and $\|p_0\|_\infty$ instead of $\|u\|_\infty$, $\|v\|_\infty$ and $\|p\|_\infty$, respectively. The results of the convergence analysis based on the modified infinity norm are given in Table 11, which indicate the convergence rate for the pressure is about the same as the velocity.

7.4. Flow passing a moving cylinder

In simulations using the immersed interface method and the immersed boundary method, flexible objects are often constructed as a network of springs. Springs are also used to tether the objects. The spring constant is given by the material property and it defines a characteristic time scale associated with the vibration mode of an object. To investigate the effect of this time scale on a time-dependent flow, we simulate flow passing a cylinder which is accelerated from rest to a uniform velocity.

The geometry is given in Fig. 15, with rigid walls as far-field boundaries. The velocity of the cylinder, u_c, is given by

$$u_c = \frac{1}{1 + \tanh(2)} \left(\tanh\left(\frac{4t}{t_c} - 2 \right) + \tanh(2) \right),$$

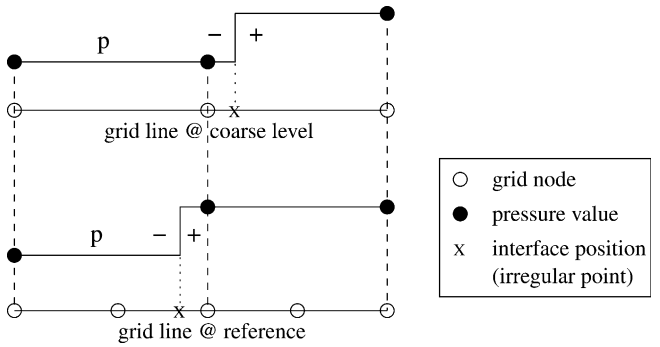


Fig. 14. Schematic to show that a grid point is at one side of a pressure discontinuity point at one grid resolution level and at the other side at another level.

Table 11
Spatial convergence analysis for the flow induced by a relaxing balloon

$N_x \times N_y$, N_m	$\|u_0\|_\infty$	Order	$\|v_0\|_\infty$	Order	$\|p_0\|_\infty$	Order
16×16, 32	2.33×10^{-2}		4.97×10^{-2}		8.59×10^{-2}	
32×32, 64	9.66×10^{-3}	1.27	2.22×10^{-2}	1.16	2.62×10^{-2}	1.71
64×64, 128	2.56×10^{-3}	1.91	6.21×10^{-3}	1.84	1.10×10^{-2}	1.25
128×128, 256	5.68×10^{-4}	2.17	6.06×10^{-4}	3.36	2.15×10^{-3}	2.35

The infinity norm is based on a reference solution and is modified to exclude the grid points near the interface.

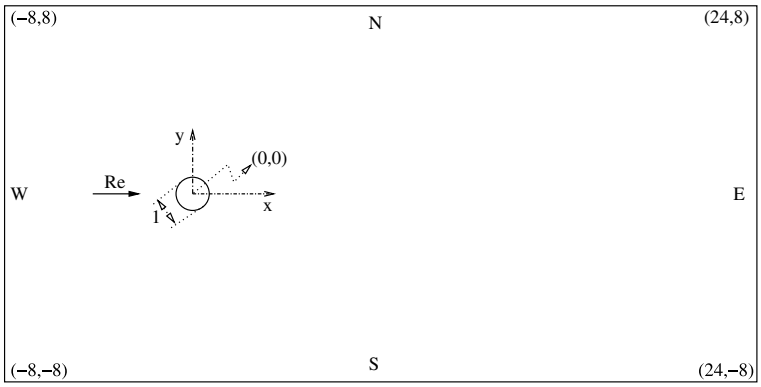


Fig. 15. Geometry for flow passing a stationary cylinder.

where t_c is the characteristic acceleration time. Fig. 16 plots u_c versus t. We define Reynolds numbers based on the uniform velocity. The spring force model is given by Eq. (47), and sketched on the left in Fig. 17. The temporal resolution of the simulation is set by the CFL numbers with $\mathrm{CFL}_v = \frac{\Delta t}{Re}(\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2}) = 0.1$ and $\mathrm{CFL}_c = \Delta t(\frac{u_{max}}{\Delta x} + \frac{v_{max}}{\Delta y}) = 0.1$.

We vary t_c to be 0.1, 0.2, 0.4, 0.8, 1.6, 3.2 and 6.4 with fixed $Re = 20$ and $K_s = 1000$. As seen in Fig. 18, the Lagrangian points can follow the prescribed motion when $t_c \geqslant 1.6$. When $t_c = 0.1$, large oscillations are seen in drag history, as shown in Fig. 19(a). Its power spectrum in Fig. 19(b) has a dominant frequency, $f_s = 7$. When $t_c = 1.6$, oscillations in the drag history has the same frequency but smaller amplitude, as seen in Fig. 20. The same frequency is observed for all values of t_c. The amplitudes of the oscillations decrease as t_c increases.

This cylinder and spring system can be effectively approximated as a spring-mass–damper system, as described on the right of Fig. 17. The frequency of the damped oscillations is given by

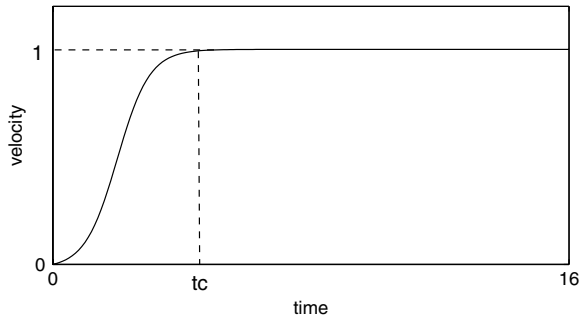


Fig. 16. Schematic showing cylinder velocity versus time.

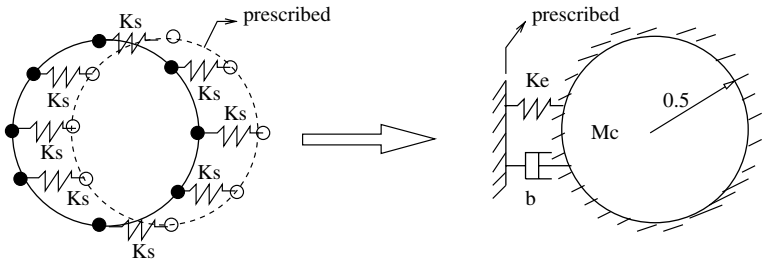
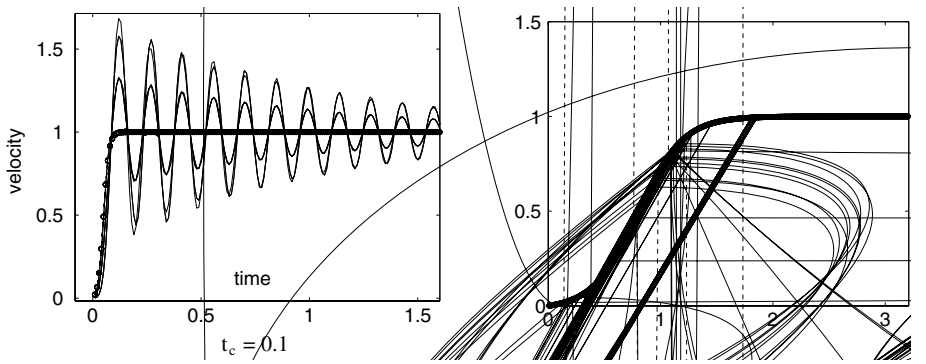


Fig. 17. Schematic showing the physical interpretation of the nonfluid force model.



$$f_s = \frac{1}{2\pi}\sqrt{\frac{K_e}{M_c}(1-\zeta^2)}, \tag{48}$$

where $K_e = 2\pi K_s$ (2π comes from the upper limit of nondimensional Lagrangian parameter α) is the effective spring stiffness, $M_c = \frac{\pi}{4}$ is the fluid mass, and $\zeta = \frac{b}{2\sqrt{M_c K_e}}$ is the damping ratio, where b is the damping coefficient of the damper. As shown in Fig. 21(a), the oscillation frequency is proportional to $\sqrt{K_s}$, as expected for a linear spring.

Define $M_s = K_s/f_s^2$. From Eq. (48), we have $M_s = \frac{2\pi M_c}{1-\zeta^2} = \frac{\pi^2}{2(1-\zeta^2)}$, which is only a function of the damping ratio, ζ. We now look at the relation between M_s and Reynolds number Re, as plotted in Fig. 21(b), where

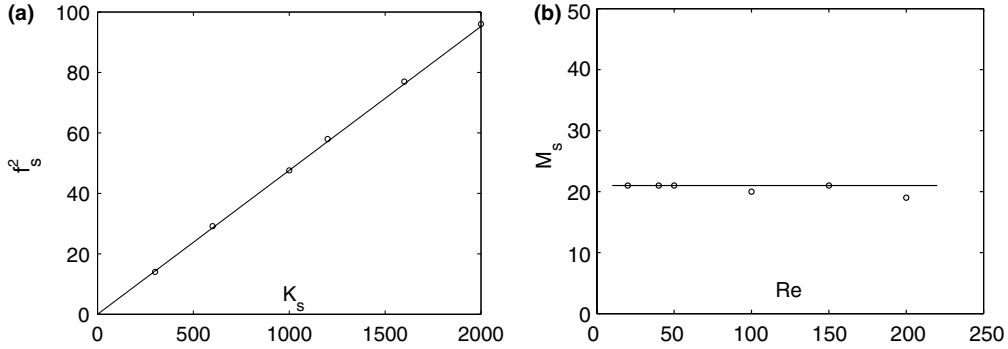


Fig. 21. (a) The frequency, f_s, versus the spring stiffness, K_s; (b) the constant (equivalent to the damping ratio), M_s, versus the Reynolds number, Re.

$Re = 20, 40, 50, 100, 150,200$ with fixed $t_c = 0.2$ and the corresponding $K_s = 1000, 500, 400, 200, 150, 100$. We observe little dependence of M_s on the Reynolds number, Re, in the range.

In summary, we find that as long as the typical time scale of the flow is about 10 times larger than the characteristic time scale of the spring system, the Lagrangian points can follow the prescribed motion. We will use this as our rule of thumb for simulating flow past a cylinder and a flapping wing in the following sections.

7.5. Flow passing a stationary cylinder

Flow passing a stationary cylinder is a canonical example to test numerical methods. We use the geometry shown in Fig. 15 for the test of our immersed interface method. Initially, the flow is set to be uniform with $u = 1$ and the cylinder moves with the same velocity as the flow. The far-field boundary conditions used for the test are

- $u = 1$, $v = 0$ and $\frac{\partial p}{\partial x} = \frac{1}{Re}\frac{\partial^2 u}{\partial x^2}$ at $x = -8$ (boundary W);
- $\frac{\partial u}{\partial x} = 0$, $\frac{\partial v}{\partial x} = 0$ and $\frac{\partial p}{\partial x} = \frac{1}{Re}\frac{\partial^2 u}{\partial x^2}$ at $x = 24$ (boundary E);
- $\frac{\partial u}{\partial y} = 0$, $v = 0$ and $\frac{\partial p}{\partial y} = \frac{1}{Re}\frac{\partial^2 v}{\partial y^2}$ at $y = -8$ (boundary S);
- $\frac{\partial u}{\partial y} = 0$, $v = 0$ and $\frac{\partial p}{\partial y} = \frac{1}{Re}\frac{\partial^2 v}{\partial y^2}$ at $y = 8$ (boundary N).

The discrete forms of the above boundary conditions are similar to those described in Eqs. (30) and (31). In particular, the divergence-free condition is incorporated in the discrete pressure boundary conditions. We take the far-upstream velocity and the cylinder diameter as the velocity and the length scales. We compute the flow at five Reynolds numbers, $Re = 20$, 40, 50, 100 and 200. The spatial resolution for these computations is $N_x \times N_y \times N_m = 640 \times 320 \times 128$. The temporal resolution is set by $\mathrm{CFL}_v = \mathrm{CFL}_c = 0.2$.

Non-fluid force $\mathbf{F}_o$ for the cylinder is a combination of a feedback control and a spring-supported membrane, as shown schematically in Fig. 22. Thus, we have $\mathbf{F}_o = \mathbf{F}_{\text{solid}} + \mathbf{F}_{\text{control}}$, with $\mathbf{F}_{\text{solid}}$ and $\mathbf{F}_{\text{control}}$ given by

$$\mathbf{F}_{\text{solid}} = E_m \frac{\partial}{\partial \alpha}\left(\left(\frac{J}{J_e} - 1\right)\tau\right) + E_s\left(1 - \frac{r}{r_e}\right)\mathbf{X}, \tag{49}$$

$$\mathbf{F}_{\text{control}} = K_d(\mathbf{V}_e - \mathbf{V}) + K_s(\mathbf{X}_e - \mathbf{X}), \tag{50}$$

where E_m and E_s are constants, r is the radius, and r_e is the desired radius, i.e. $r_e = 0.5$. Table 12 lists the values of force parameters E_m, E_s, K_d and K_s for the considered Reynolds numbers. The dominant parameter in these tests is K_s. For the Reynolds numbers considered here, we can determine K_s from the following empirical formula:

$$K_s \approx \frac{20000}{Re}.$$

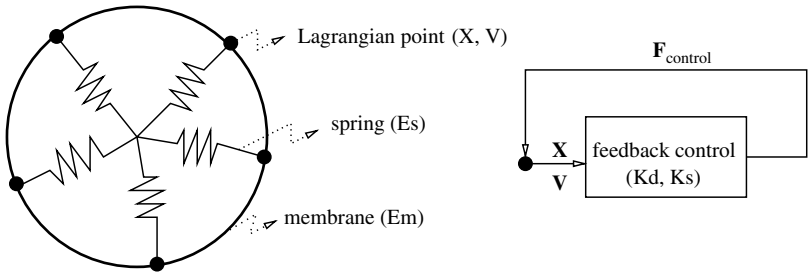


Fig. 22. Schematic of the force construction for a stationary cylinder in flow.

Table 12
Parameter values in the force construction for a stationary cylinder in flow at different Reynolds numbers

	$Re = 20$	$Re = 40$	$Re = 50$	$Re = 100$	$Re = 200$
E_m	0	40	0	10	0
E_s	0	40	0	20	0
K_d	0.1	0.1	0	0	0
K_s	1000	400	500	160	100

We remark that the combination of the solid model and the feedback control in these tests is to demonstrate the flexibility of the force construction. In cases of $Re = 20$, 50 and 200, we only use the feedback control.

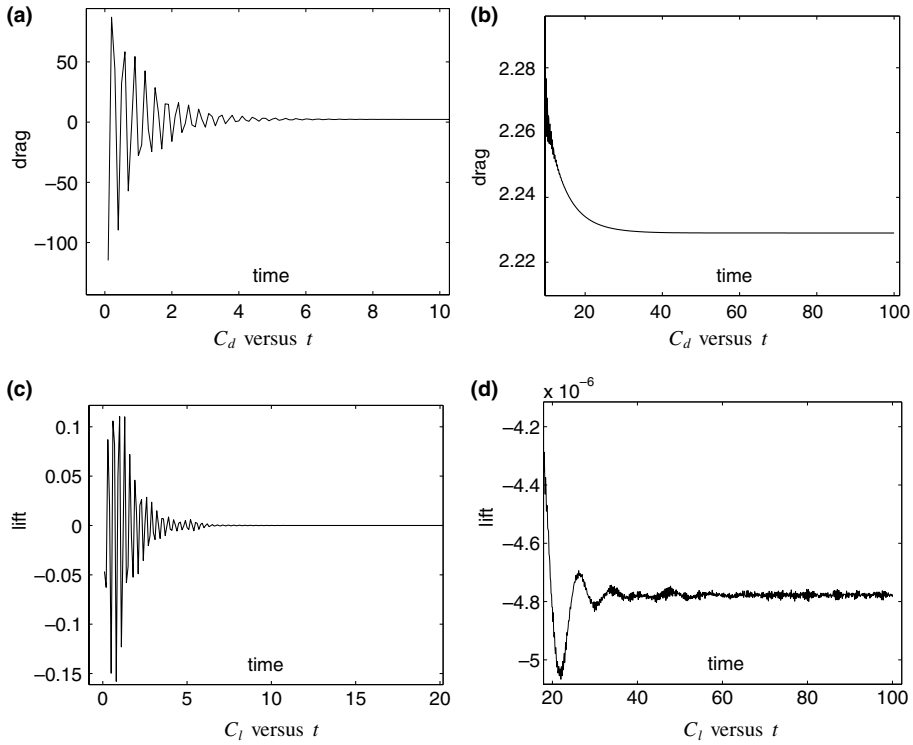


Fig. 23. Drag and lift coefficients versus time for flow passing a stationary cylinder at $Re = 20$: (a) drag history in transient; (b) drag history from transient to steady state; (c) lift history in transient; (d) lift history from transient to steady state.

7.5.1. Re = 20, 40 and 50

Fig. 23 shows the drag and the lift history for $Re = 20$. The large oscillations at the start are due to the impulsive stop of the cylinder, which are analyzed in the previous section. The flow can be considered steady at $t = 100$, long after the transient. The nonzero lift of order 10^{-6} is ascribed to numerical error.

Between $Re = 40$ and $Re = 50$, the flow becomes unstable and develops a von Karman wake. Figs. 24 and 25 show the drag and the lift history for $Re = 40$ and $Re = 50$. At $Re = 40$ the oscillations in lift damp out, and at $Re = 50$ they amplify.

Figs. 26–28 show the flow details around the cylinder for $Re = 20$ at $t = 100$ and $Re = 40$ at $t = 120$.

Table 13 compares length of the trailing bubble L_{TB}, angle of separation θ_s, and drag coefficient C_d with previous experimental and computational results summarized in [27]. Geometric quantities T_{LB} and θ_s compare favorably with others, though the value of drag coefficient C_d is slightly higher than previous studies. Russell and Wang [27] suggested that their simplification of the far-field boundary conditions is a source of increased drag. We believe the reason for the drag increase in our case is the same. To test this, we have changed the domain size for $Re = 20$ from 32×16 to 48×24 with the corresponding change of $N_x \times N_y$ from 640×320 to 960×480. The new drag and lift history is shown in Fig. 29. At the steady state, the drag coefficient settles down to 2.14, which is in the range of previously reported values.

The flow inside the cylinder is static in our case. Thus the vorticity and the pressure distributions over the cylinder surface, denoted as ω_s and p_s respectively, can be calculated as follows:

$$\omega_s = \left(\frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}\right)_s = \left[\frac{\partial v}{\partial x}\right] - \left[\frac{\partial u}{\partial y}\right] = -Re \cdot f_\tau, \quad p_s = [p] = f_n,$$

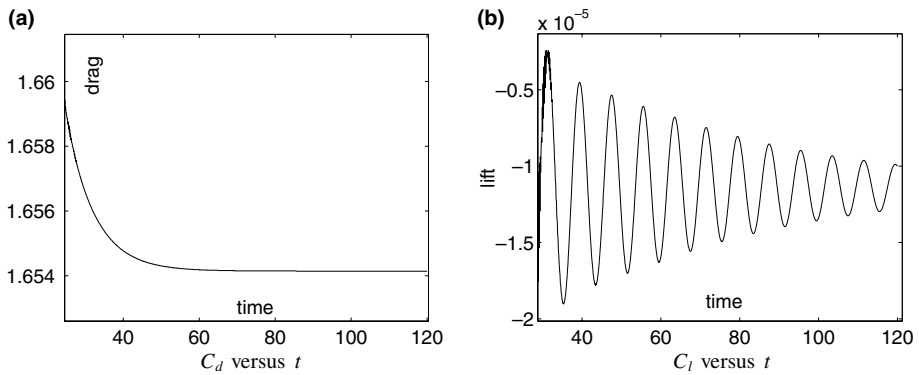


Fig. 24. (a) Drag and (b) lift coefficients versus time for flow passing a stationary cylinder at $Re = 40$.

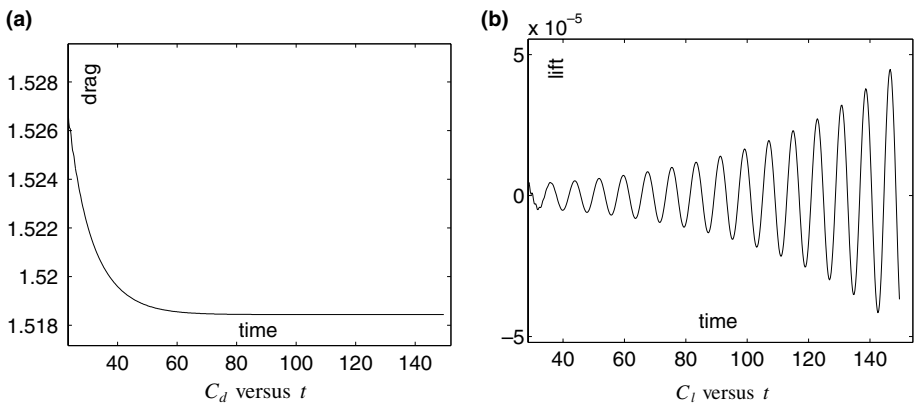


Fig. 25. (a) Drag and (b) lift coefficients versus time for flow passing a stationary cylinder at $Re = 50$.

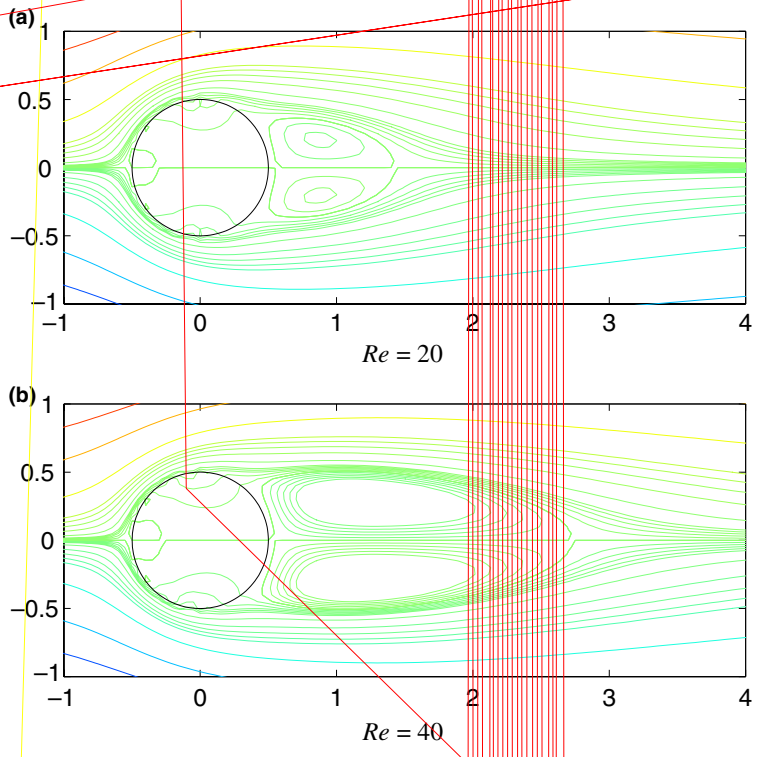


Fig. 26. Streamlines of flow passing a stationary cylinder at the steady state.

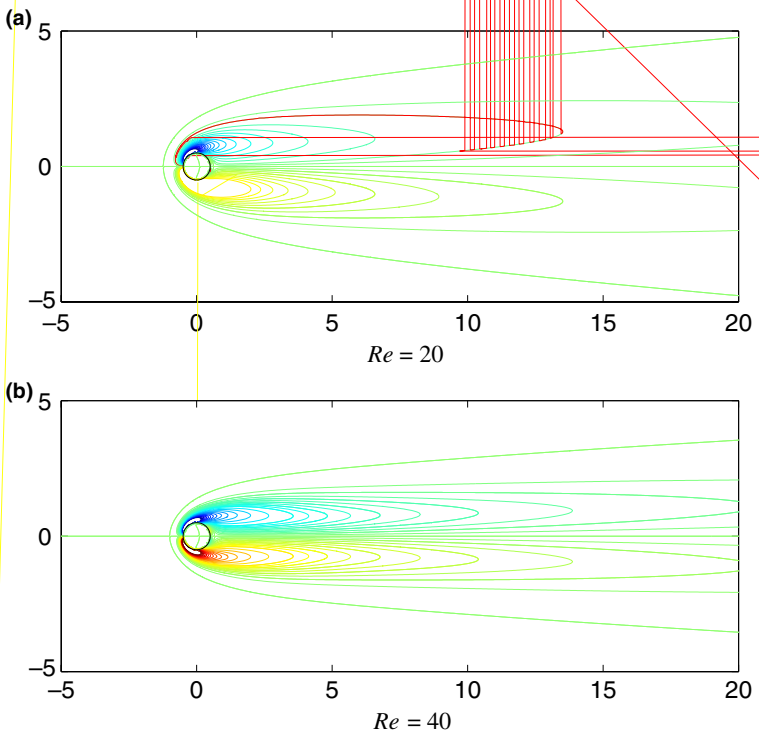
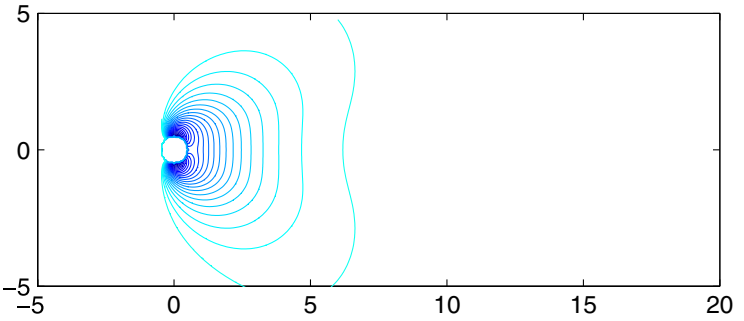


Fig. 27. Vorticity contours of flow passing a stationary cylinder at the steady state.


5
0
−5
−5
0
5
10
15
20

where $[\cdot] := (\cdot)_{\Gamma^+} - (\cdot)_{\Gamma^-}$. Figs. 30 and 31 compare the vorticity and the pressure distributions over the cylinder surface with the previous computational results [2] for $Re = 20$ and $Re = 40$, indicating a good agreement.

We also monitor the surface position and the surface velocity distribution for an object to see whether the force model enforces the desired motion while maintaining the shape. Fig. 32(a) shows the temporal variation of the relative error e_r for the surface position, where e_r is defined as

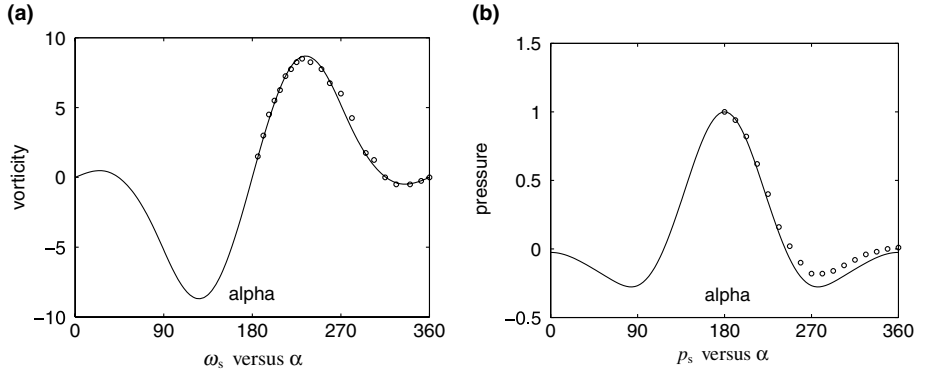


Fig. 30. (a) Vorticity and (b) pressure distributions on the cylinder surface in flow at $Re = 20$ – lines: current results, open circles: numerical results from [2].

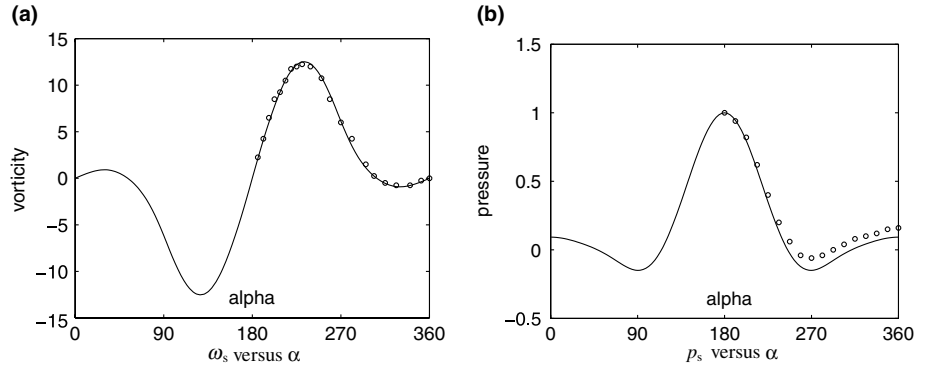


Fig. 31. Vorticity (a) and pressure (b) distributions on cylinder surface in flow at $Re = 40$ – lines: current results, open circles: numerical results from [2].

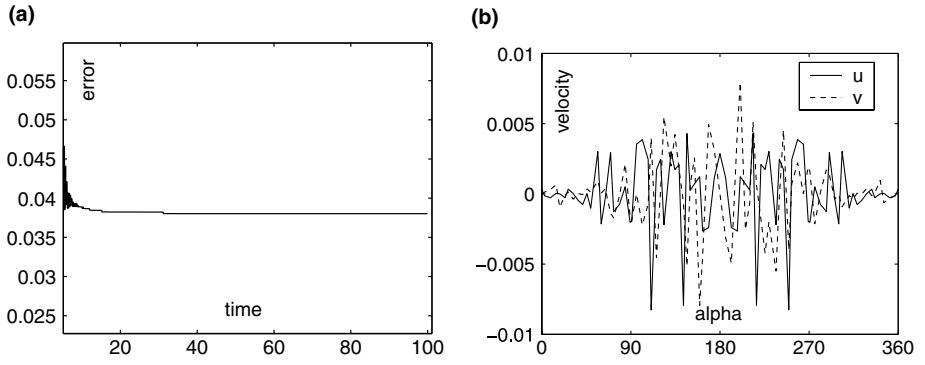


Fig. 32. Flow past a stationary cylinder at $Re = 20$: (a) relative shape error e_r versus time; (b) the velocity distribution on the cylinder surface at time $t = 100$.

$$e_r = 100\left|\frac{\max_m(\sqrt{X_m^2 + Y_m^2}) - 0.5}{0.5}\right|.$$

The relative error of the surface position at steady state is less than 0.04%. Fig. 32(b) shows the velocity distribution around the cylinder surface at time $t = 100$.

7.5.2. Re = 100 and 200

Figs. 33 and 34 are the drag and the lift history for $Re = 100$ and $Re = 200$.

Figs. 35 and 36 show the flows around the cylinder in terms of vorticity and pressure. The expected trailing Von Karman vortex street develops in the wake.

Table 14 provides a summary of results for $Re = 100$ and $Re = 200$, where S_t is the Strouhal number, i.e. the nondimensional vortex shedding frequency. Our results are within the ranges of the previously reported

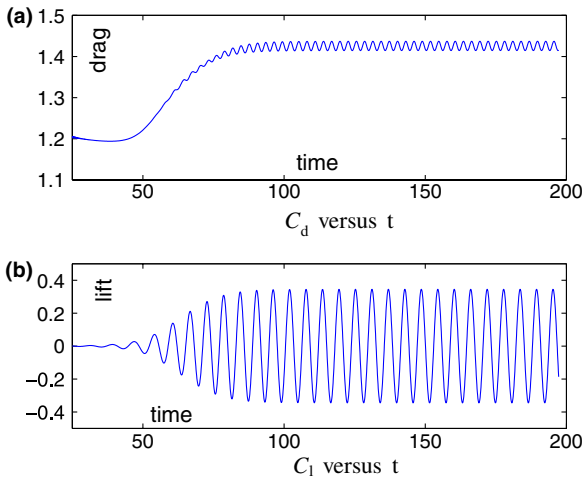


Fig. 33. (a) Drag and (b) lift coefficients versus time for flow passing a stationary cylinder at $Re = 100$.

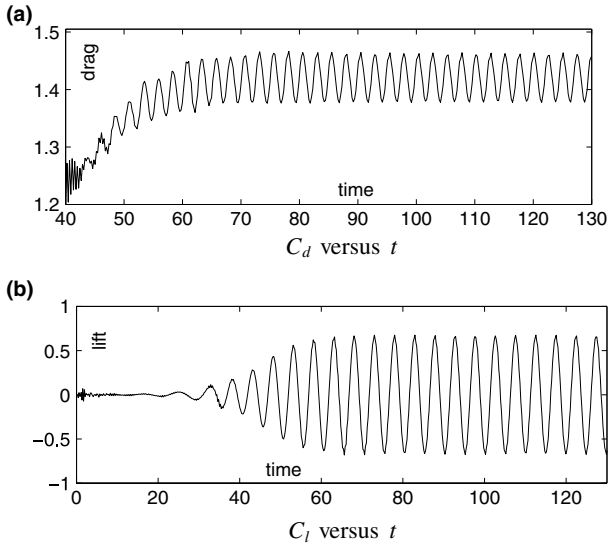


Fig. 34. (a) Drag and (b) lift coefficients versus time for flow passing a stationary cylinder at $Re = 200$.

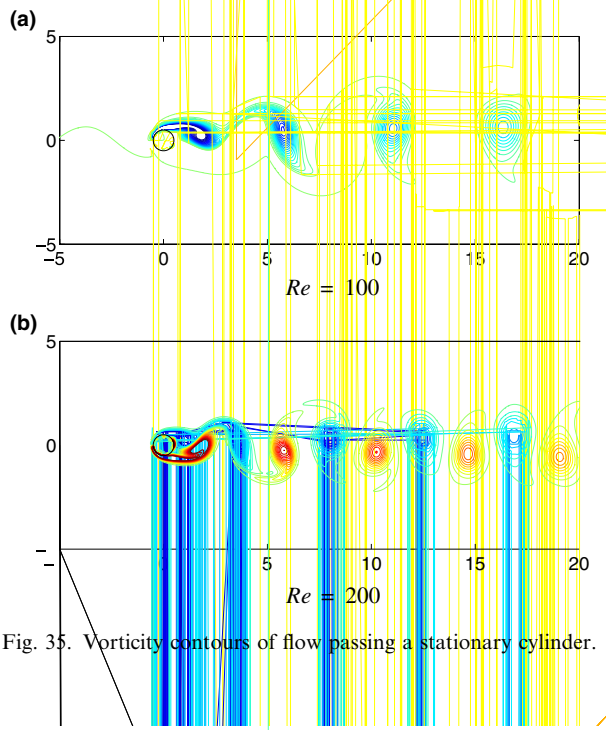
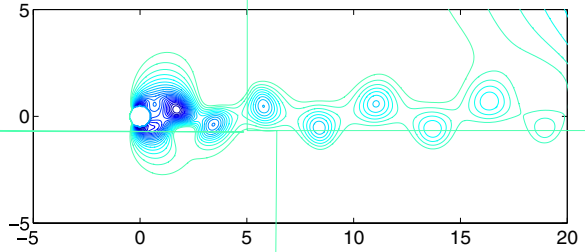
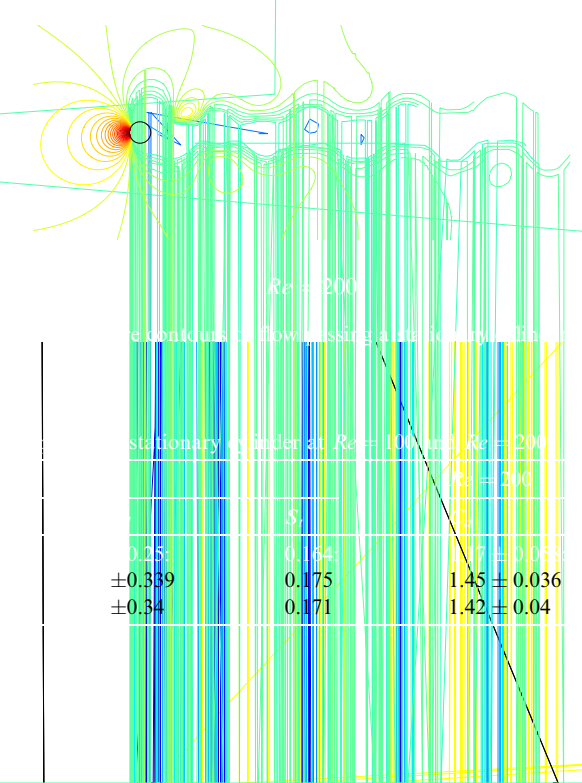


Fig. 35. Vorticity contours of flow passing a stationary cylinder.





[27]	1.43 ± 0.009	± 0.339	0.175	1.45 ± 0.036	± 0.75	0.202
Present	1.423 ± 0.013	± 0.34	0.171	1.42 ± 0.04	± 0.66	0.202

values. We have also changed the domain size for $Re = 100$ from 32×16 to 48×24 with the corresponding change of $N_x \times N_y$ from 640×320 to 960×480. The new drag and lift history is shown in Fig. 37, which shows that $C_d = 1.379 \pm 0.009$, $C_l = \pm 0.31$, and $S_t = 0.167$.

7.6. Flow around a flapping wing

In this example, we simulate the flow around a hovering wing inside a rigid box, as described in Fig. 38, with $N_x \times N_y \times N_m = 512 \times 512 \times 512$ and $\Delta t = 0.001$. The nonfluid force is given by Eqs. (49) and (50) with $E_m = 2$, $E_s = 2$, $K_d = 0.1$ and $K_s = 160$. The wing is an ellipse with chord length c and aspect ratio e. We take c as the length scale, $\frac{\pi A_0}{T_f}$ the velocity scale, and $\frac{cT_f}{\pi A_0}$ the time scale, where A_0 is the amplitude of the wing center translation and T_f is the wing flapping period. The nondimensional flapping period is $t_f = \frac{\pi A_0}{c}$. The nondimensionalized wing motion is governed by

$$a(t) = 0.5a_0\left(\cos\left(\frac{2t}{a_0}\right) + 1\right),$$
$$\theta(t) = \theta_0\left(1 - \sin\left(\frac{2t}{a_0} + \phi\right)\right),$$

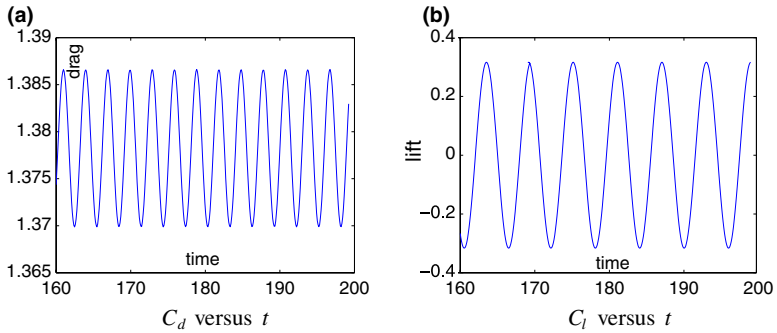


Fig. 37. (a) Drag and (b) lift coefficients versus time for flow passing a stationary cylinder at $Re = 100$ in a domain of size 48×24.

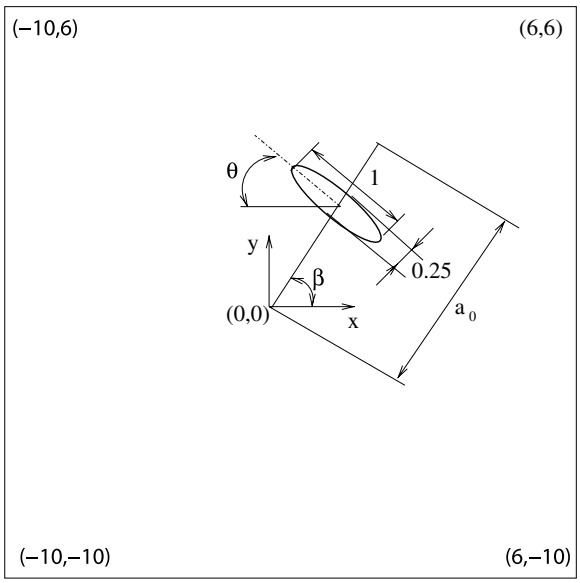


Fig. 38. Geometry for flow around a hovering flapper.

where $a(t)$ is the displacement of the wing with an amplitude $a_0 = A_0/c$, $\theta(t)$ is the angle of attack (we use θ instead of α, since α is used as the Lagrangian parameter of a surface) with amplitude $2\theta_0$, and ϕ the phase difference. The Reynolds number is defined as $Re = \frac{\pi A_0 c}{T_f \nu}$. We run the simulation for $e = 4$, $a_0 = 2.5$, $\theta_0 = \frac{\pi}{4}$, $\phi = 0$ and $Re = 157$, which are the same as used by Wang [35].

Fig. 39 shows four snapshots of the computed vorticity fields near the wing during one flapping period. They are very similar to those obtained by Wang [35], where the physical interpretation was given.

We plot instantaneous drag and lift in Fig. 40 and compare them with the results of Wang [35]. Considering the difference in the far-field boundary conditions and the inevitable difference of the wing motion due to the oscillations of Lagrangian points, the agreement is quite good.

We define the shape distortion to be

$$e_d = \max_m \left((X_m - X_{em})^2 + (Y_m - Y_{em})^2 \right),$$

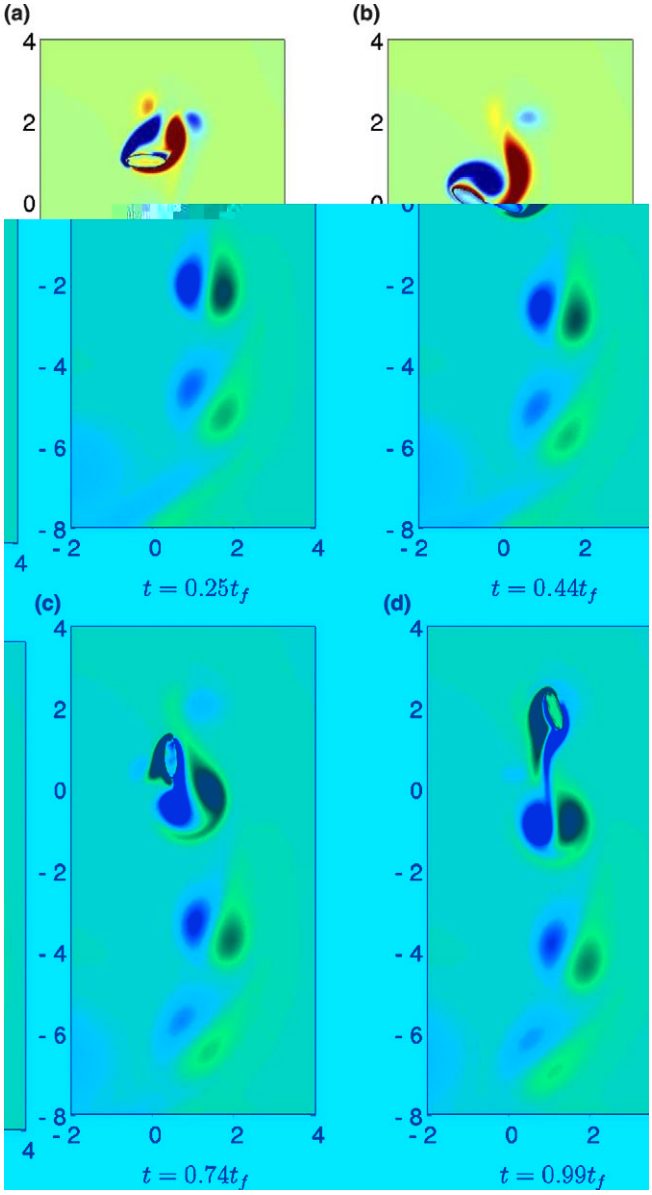


Fig. 39. Vorticity fields around a hovering wing of $Re = 157$ at four different instants in a flapping period.

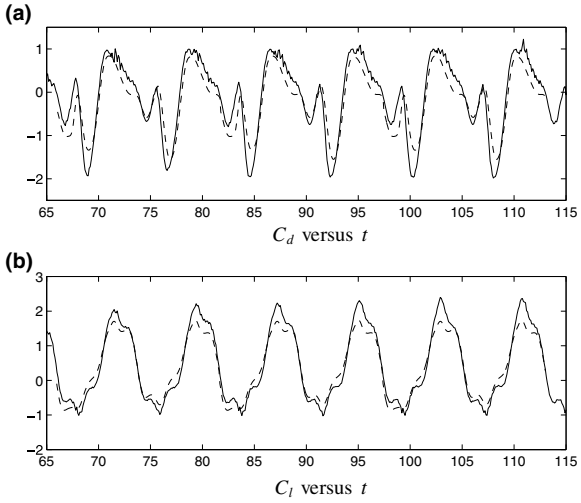


Fig. 40. (a) Drag and (b) lift coefficients versus time for flow around a hovering flapper of $Re = 157$—solid lines: current results, dashed lines: previous results [35].

where (X_{em}, Y_{em}) is the prescribed coordinates of Lagrangian point m. The shape distortion of the wing is very small, as seen in Fig. 41.

Fig. 42(a) and (b) show the velocity evolution of Lagrangian points at the leading edge and in the middle of the wing, which indicates that it is more difficult to control the Lagrangian points at the leading and trailing edges to follow the prescribed velocity.

7.7. Flow passing multiple cylinders

We provide two examples below to demonstrate the capability and efficiency of our method to simulate flows with multiple moving objects.

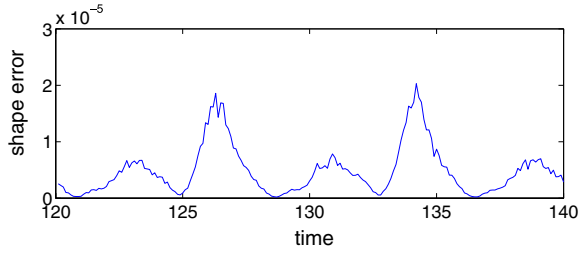


Fig. 41. The shape error of a hovering wing at $Re = 157$.

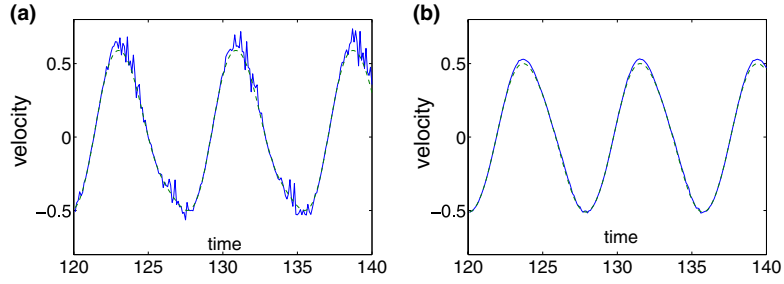


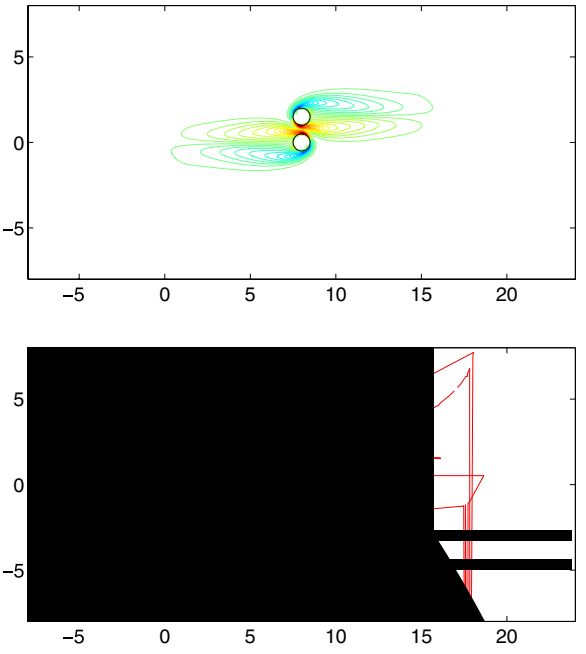
Fig. 42. Velocity on the wing surface: (a) the velocity versus time of a Lagrangian point at $\alpha = 0$; (b) the velocity versus time of a Lagrangian point at $\alpha = \frac{\pi}{2}$.

7.7.1. Two cylinders moving with respect to each other

This example was previously studied by Russell and Wang [27]. The initial geometry, shown in Fig. 43, is the same as theirs. In this simulation, $N_x \times N_y \times N_m = 640 \times 320 \times 128$, and $\Delta t = 0.0005$. The far-field boundaries are rigid walls. The nonfluid force for both cylinders is given by Eq. (47) with $K_s = 800$. To avoid the impulsive start of the cylinders, we let each cylinder oscillate about its initial position for two periods and then move toward the other at $Re = 40$. The motion of the lower cylinder is given by

$$x_{lc} = \begin{cases} \frac{4}{\pi} \sin\left(\frac{\pi t}{4}\right), & 0 \leqslant t \leqslant 16, \\ t - 16, & 16 \leqslant t \leqslant 32, \end{cases}$$

and the motion of the upper cylinder is given by



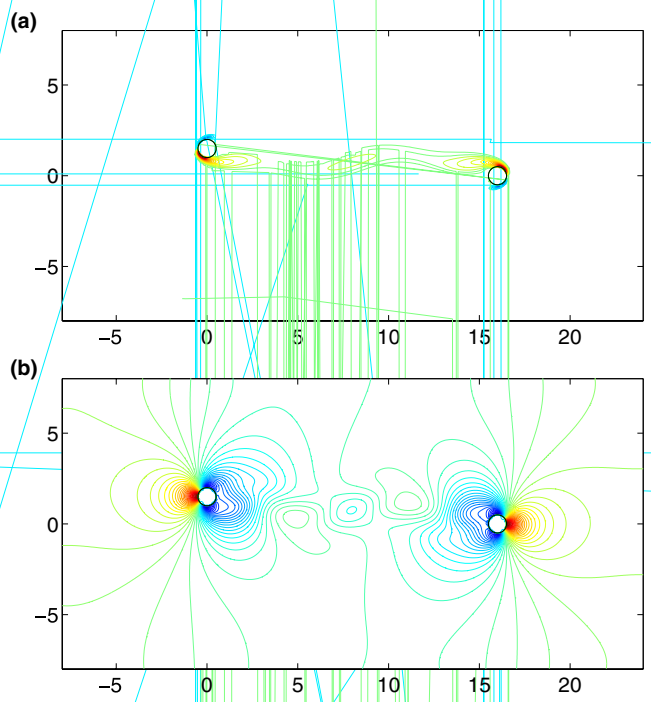


Fig. 45. Flow fields around two cylinders moving with respect to each other at $Re = 40$ when two cylinders are separated by a distance of 16. Contours of (a) vorticity and (b) pressure.

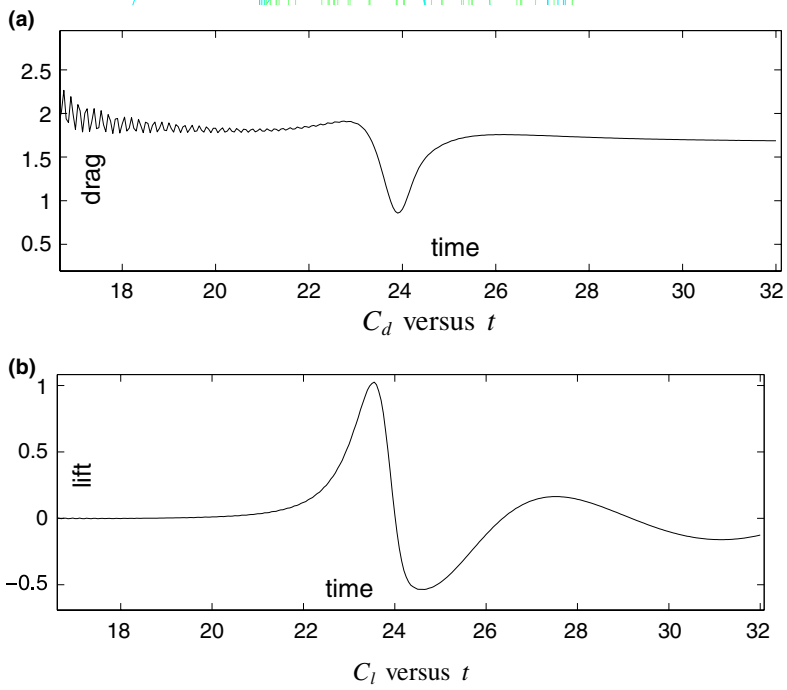


Fig. 46. (a) Drag and (b) lift coefficients versus time for the upper cylinder in flow around two cylinders moving with respect to each other at $Re = 40$.

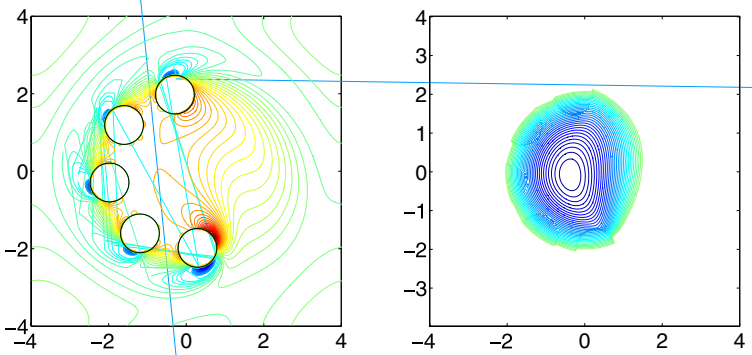
$$x_{uc} = \begin{cases} 16 - \frac{4}{\pi}\sin\left(\frac{\pi t}{4}\right), & 0 \leqslant t \leqslant 16, \\ 32 - t, & 16 \leqslant t \leqslant 32. \end{cases}$$

Fig. 44 shows the vorticity and pressure fields at time $t = 24$, when the two cylinders are closest to each other, and Fig. 45 shows the vorticity and pressure fields at time $t = 32$.

Fig. 46 shows the temporal evolution of the drag and lift coefficients for the upper cylinder. We see the same drag behavior as obtained by Russell and Wang [27], that is an increase in drag as the two cylinders approach each other and a decrease in drag as they pass in close proximity. The two cylinders are repulsive when approaching each other and become attractive after passed each other.

7.7.2. Cylinders translating along a circle

In this last example, we simulate cylinders translating with the same speed along a circle to examine the efficiency of our method for simulating multiple moving objects.



The geometry of the simulation is given in Fig. 47. The Reynolds number based on the diameter and the translational speed of a cylinder is $Re = 20$. The nonfluid force for each cylinder is modeled again by Eqs. (49) and (50) with $E_m = 20$, $E_s = 20$, $K_d = 0$ and $K_s = 800$. Simulations are carried out with $N_x \times N_y \times N_m = 256 \times 256 \times 256$, and $\Delta t = 0.0002$.

Table 15 lists the computational time versus the number of cylinders in 40,000 time integration steps. Since the computational time associated with a cylinder is of order $\mathcal{O}(N_m)$, there is no significant increase of the computational time when one or two more cylinders are added. Most computational time is spent on the FFT Poisson solver for the pressure, which has cost of order $\mathcal{O}(N_{ij} \ln(N_{ij}))$.

Fig. 48 shows the vorticity and the pressure contours for the flow containing five cylinders.

8. Conclusions

The immersed interface method shares the same mathematical formulation and therefore the advantage of Peskin's immersed boundary method. With the appropriate inclusion of jump conditions in finite difference schemes, the immersed interface method as described here can achieve higher order accuracy, maintain a sharp interface and preserve volumes. Specifically, our numerical tests show that second-order spatial accuracy in terms of the infinity norm for both the velocity and the pressure can be achieved. Because of the use of a fixed Cartesian grid, the method is simple and efficient for flows with moving objects. The cost to treat an object is of order $\mathcal{O}(N_m)$, where N_m is the number of Lagrangian points in the object representation. The method can be applied equally well to objects with prescribed force or objects with prescribed motion. In the current implementation of the method, we choose to employ uniform grids and FFT Poisson solvers, but the method is amenable to nonuniform grids and other Poisson solvers.

We are now implementing the method in 3D with the jump conditions that have been presented in [38].

Acknowledgements

We thank Professor Randall J. LeVeque and Professor Zhilin Li for helpful discussions during the course of this work. The work is supported by AFOSR and Packard Foundation.

References

[1] R.P. Beyer, R.J. Leveque, Analysis of a one-dimensional model for the immersed boundary method, SIAM J. Numer. Anal. 29 (2) (1992) 332–364.
[2] M. Braza, P. Chassaing, H. Ha Minh, Numerical study and physical analysis of the pressure and velocity fields in the near wake of a circular cylinder, J. Fluid Mech. 165 (1986) 79–130.
[3] Donna Calhoun, A Cartesian grid method for solving the two-dimensional streamfunction-vorticity equations in irregular regions, J. Comput. Phys. 176 (2002) 231–275.
[4] R. Cortez, M. Minion, The Blob projection method for immersed boundary problems, J. Comput. Phys. 161 (2000) 428–453.
[5] W. E, Jian-Guo Liu, Vorticity boundary condition and related issues for finite difference schemes, J. Comput. Phys. 124 (1996) 368–382.
[6] E.A. Fadlun, R. Verzicco, P. Orlandi, J. Mohd-Yusof, Combined immersed-boundary finite-difference methods for three-dimensional complex flow simulations, J. Comput. Phys. 161 (2000) 35–60.
[7] Ronald P. Fedkiw, Tariq Aslam, Barry Merriman, Stanley Osher, A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the Ghost Fluid Method), J. Comput. Phys. 152 (1999) 457–492.
[8] Aaron L. Fogelson, James P. Keener, Immersed interface method for Neumann and related problems in two and three dimensions, SIAM J. Sci. Comput. 22 (5) (2000) 1630–1654.
[9] D. Goldstein, R. Handler, L. Sirovich, Modeling a no-slip flow boundary with an external force field, J. Comput. Phys. 105 (1993) 354–366.
[10] Yue Hao, Andrea Prosperetti, A numerical method for three-dimensional gas–liquid flow computations, J. Comput. Phys. 196 (2004) 126–144.
[11] Francis H. Harlow, J. Eddie Welch, Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface, Phy. Fluids 8 (12) (1965) 2182–2189.
[12] Hans Johnston, Jian-Guo Liu, Finite difference schemes for incompressible flow based on local pressure boundary conditions, J. Comput. Phys. 180 (2002) 120–154.
[13] Hans Johnston, Jian-Guo Liu, Accurate, stable and efficient Navier–Stokes solvers based on explicit treatment of the pressure term, J. Comput. Phys. 199 (2004) 221–259.

[14] Jungwoo Kim, Dongjoo Kim, Haecheon Choi, An immersed-boundary finite-volume method for simulations of flow in complex geometries, J. Comput. Phys. 171 (2001) 132–150.
[15] Ming-Chih Lai, Charles S. Peskin, An immersed boundary method with formal second-order accuracy and reduced numerical viscosity, J. Comput. Phys. 160 (2000) 705–719.
[16] Long Lee, Randall J. LeVeque, An immersed interface method for incompressible Navier–Stokes equations, SIAM J. Sci. Comput. 25 (3) (2003) 832–856.
[17] Randall J. LeVeque, Zhilin Li, The immersed interface method for elliptic equations with discontinuous coefficients and singular sources, SIAM J. Numer. Anal. 31 (4) (1994) 1019–1044.
[18] Randall J. LeVeque, Zhilin Li, Immersed interface methods for stokes flow with elastic boundaries or surface tension, SIAM J. Sic. Comput. 18 (3) (1997) 709–735.
[19] Zhilin Li, Ming-Chih Lai, The immersed interface method for the Navier–Stokes equations with singular forces, J. Comput. Phys. 171 (2001) 822–842.
[20] Zhilin Li, Private Communication.
[21] Charles S. Peskin, Flow patterns around heart valves: a numerical method, J. Comput. Phys. 10 (1972) 252–271.
[22] Charles S. Peskin, Numerical analysis of blood flow in the heart, J. Comput. Phys. 25 (1977) 220–252.
[23] Charles S. Peskin, Beth Feller Printz, Improved volume conservation in the computation of flows with immersed elastic boundaries, J. Comput. Phys. 105 (1993) 33–46.
[24] Charles S. Peskin, The immersed boundary method, Acta Numer. (2002) 1–39.
[25] A. Prosperetti, H.N. Oğuz, PHYSALIS: a new o(N) method for the numerical simulation of disperse systems: potential flow of spheres, J. Comput. Phys. 167 (2001) 196–216.
[26] A.M. Roma, C.S. Peskin, M.J. Berger, An adaptive version of the immersed boundary method, J. Comput. Phys. 153 (1999) 509–534.
[27] David Russell, Z. Jane Wang, A Cartesian grid method for modeling multiple moving objects in 2D incompressible viscous flow, J. Comput. Phys. 191 (2003) 177–205.
[28] E.M. Saiki, S. Biringen, Numerical simulation of a cylinder in uniform flow: application of a virtual boundary method, J. Comput. Phys. 123 (1996) 450–465.
[29] A.L.F. Lima, E. Silva, A. Silveira-Neto, J.J.R. Damasceno, Numerical simulation of two-dimensional flows over a circular cylinder using the immersed boundary method, J. Comput. Phys. 189 (2003) 351–370.
[30] John M. Stockie, Brian R. Wetton, Analysis of stiffness in the immersed boundary method and implications for time-stepping schemes, J. Comput. Phys. 154 (1999) 41–64.
[31] S. Takagi, H.N. Oğuz, Z. Zhang, A. Prosperetti, PHYSALIS: a new method for particle simulation Part II: two-dimensional Navier–Stokes flow around cylinders, J. Comput. Phys. 187 (2003) 371–390.
[32] Yu-Heng Tseng, Joel H. Ferziger, A ghost-cell immersed boundary method for flow in complex geometry, J. Comput. Phys. 192 (2003) 593–623.
[33] C. Tu, C.S. Peskin, Stability and instability in the computation of flows with moving immersed boundaries, SIAM J. Statist. Comput. 13 (1992) 70–83.
[34] H.S. Udaykumar, R. Mittal, P. Rampunggoon, A. Khanna, A sharp interface Cartesian grid method for simulating flows with complex moving boundaries, J. Comput. Phys. 174 (2001) 345–380.
[35] Z. Jane Wang, Two dimensional mechanism for insect hovering, Phys. Rev. Lett. 85 (10) (2000).
[36] Andreas Wiegmann, Kenneth P. Bube, The immersed interface method for nonlinear differential equations with discontinuous coefficients and singular sources, SIAM J. Numer. Anal. 35 (1) (1998) 177–200.
[37] Andreas Wiegmann, Kenneth P. Bube, The explicit-jump immersed interface method: finite difference methods for PDEs with piecewise smooth solutions, SIAM J. Numer. Anal. 37 (3) (2000) 827–862.
[38] Sheng Xu, Z. Jane Wang, Systematic derivation of jump conditions for the immersed interface method in three-dimensional flow simulation, SIAM J. Sci. Comput., 2006, in press.
[39] T. Ye, R. Mittal, H.S. Udaykumar, W. Shyy, An accurate Cartesian grid method for viscous incompressible flows with complex immersed boundaries, J. Comput. Phys. 156 (1999) 209–240.
[40] Luoding Zhu, Charles S. Peskin, Simulation of a flapping flexible filament in a flowing soap film by the immersed boundary method, J. Comput. Phys. 179 (2002) 452–468.