INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2010; **62**:267–290
Published online 26 February 2009 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/fld.2018

A level set-based immersed interface method for solving incompressible viscous flows with the prescribed velocity at the boundary

Zhijun Tan[1,*,†], K. M. Lim[1,2] and B. C. Khoo[1,2]

[1]*Singapore-MIT Alliance, 4 Engineering Drive 3, National University of Singapore, Singapore 117576, Singapore*
[2]*Department of Mechanical Engineering, National University of Singapore, 10 Kent Ridge Crescent, Singapore 119260, Singapore*

SUMMARY

A second-order accurate immersed interface method (IIM) is presented for solving the incompressible Navier–Stokes equations with the prescribed velocity at the boundary, which is an extension of the IIM of Le *et al.* (*J. Comput. Phys.* 2006; **220**:109–138) to a level set representation of the boundary in place of the Lagrangian representation of the boundary using control points on a uniform Cartesian grid. In order to enforce the prescribed velocity boundary condition, the singular forces at the immersed boundary are applied on the fluid. These forces are related to the jump in pressure and the jumps in the derivatives of both the pressure and velocity, and are approximated via using the local Hermite cubic spline interpolation. The strength of singular forces is determined by solving a small system of equations at each time step. The Navier–Stokes equations are discretized via using finite difference method with the incorporation of jump conditions on a staggered Cartesian grid and solved by a second-order accurate projection method. Numerical results demonstrate the accuracy and ability of the proposed method to simulate the viscous flows in irregular domains. Copyright © 2009 John Wiley & Sons, Ltd.

Received 31 March 2008; Revised 12 January 2009; Accepted 16 January 2009

KEY WORDS: incompressible Navier–Stokes equations; singular force; immersed interface method; projection method; level set method; the prescribed velocity

1. INTRODUCTION

This paper concerns the viscous incompressible flows with the prescribed velocity condition at the boundary. In a two-dimensional bounded domain D with the irregular boundary ∂D_1 and

*Correspondence to: Zhijun Tan, Singapore-MIT Alliance, 4 Engineering Drive 3, National University of Singapore, Singapore 117576, Singapore.
†E-mail: smatz@nus.edu.sg

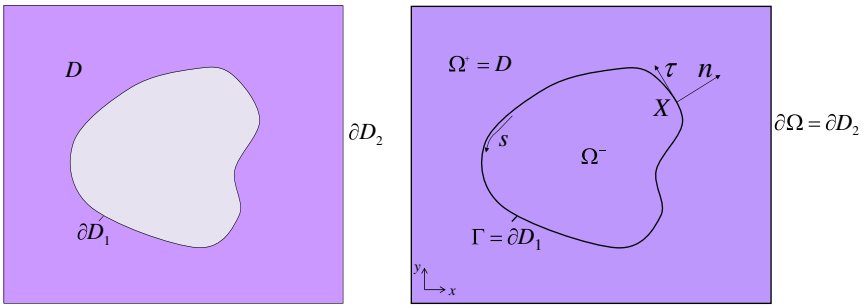

Figure 1. A typical irregular domain (left) and the extended regular rectangular domain with an embedded boundary (right) for the Navier–Stokes equations.

regular boundary ∂D_2, the incompressible Navier–Stokes equations formulated in the primitive velocity–pressure variables are considered and written as

$$\rho(\mathbf{u}_t + (\mathbf{u}\cdot\nabla)\mathbf{u}) + \nabla p = \mu\Delta\mathbf{u} + \mathbf{G}^{\text{ext}}(\mathbf{x}, t), \quad \mathbf{x} \in D \tag{1}$$

$$\nabla\cdot\mathbf{u} = 0, \quad \mathbf{x} \in D \tag{2}$$

with initial and boundary conditions

$$\mathbf{u}(\mathbf{x}, 0) = \mathbf{u}_0, \quad \mathbf{u}|_{\partial D_1} = \mathbf{u}_p, \quad \mathbf{u}|_{\partial D_2} = \mathbf{u}_b \tag{3}$$

where $\mathbf{u} = (u, v)^{\text{T}}$ is the fluid velocity, p is the fluid pressure, ρ is the fluid density, μ is the fluid viscosity, $\mathbf{x} = (x, y)$ is the Cartesian coordinate variable, $\mathbf{G}^{\text{ext}}(\mathbf{x}, t) = (G_1^{\text{ext}}, G_2^{\text{ext}})^{\text{T}}$ is an external forcing term. Throughout this paper, the fluid density ρ and fluid viscosity μ are assumed to be constants over the whole domain. The readers are referred to Figure 1 (left) for an illustration of the problem.

The irregular domain D can be extended to a larger rectangular domain Ω by an embedding technique, see Figure 1 (right), where $\Omega^+ = D$ and $\partial\Omega = \partial D_2$. In order to impose the prescribed velocity condition at the irregular boundary, i.e. $\mathbf{u}|_{\partial D_1} = \mathbf{u}_p$, the boundary ∂D_1 can be treated as an immersed boundary Γ which exerts force to the fluid [1, 2]. These singular forces at the immersed boundary Γ can be introduced as the augmented variables so that the irregular boundary condition is satisfied, i.e. $\mathbf{u}|_\Gamma = \mathbf{u}_p$. The boundary Γ separates the extended fluid region into two parts Ω^+ and Ω^- with $\Omega = \Omega^+ \cup \Gamma \cup \Omega^-$. Therefore, finding the solutions of Equations (1)–(3) is equivalent to solving the following equations:

$$\rho(\mathbf{u}_t + (\mathbf{u}\cdot\nabla)\mathbf{u}) + \nabla p = \mu\Delta\mathbf{u} + \mathbf{G}^{\text{ext}}(\mathbf{x}, t) + \mathbf{F}(\mathbf{x}, t), \quad \mathbf{x} \in \Omega \tag{4}$$

$$\nabla\cdot\mathbf{u} = 0, \quad \mathbf{x} \in \Omega \tag{5}$$

$$\mathbf{u}|_\Gamma = \mathbf{u}_p \tag{6}$$

with the boundary condition $\mathbf{u}|_{\partial\Omega} = \mathbf{u}_b$. Here, Equation (6) is the corresponding augmented equation and the singular force $\mathbf{F}$ has the form of

$$\mathbf{F}(\mathbf{x}, t) = \int_\Gamma \mathbf{f}(s, t)\delta(\mathbf{x} - \mathbf{X}(s, t))\,\mathrm{d}s \tag{7}$$

DOI: 10.1002/fld

where $\mathbf{X}(s,t)=(X(s,t),Y(s,t))$ is the arc-length parametrization of the boundary Γ, s is the arc-length, $\mathbf{f}=(f_1,f_2)^{\mathrm{T}}$ is the force density, and $\delta(\cdot)$ is the Dirac delta function defined in the distribution sense. With the above embedding forcing approach, the original problem for (1)–(3) becomes an interface problem for (4)–(6) with a complete system on the regular domain. The solution in Equations (4)–(6) is a functional of the singular force $\mathbf{f}$. For the current problems, the immersed boundary is rigid (i.e. the irregular boundary is fixed) and the velocity at the rigid boundary is specified, then the singular force at the rigid boundary is determined by the requirement that the fluid velocity should satisfy the prescribed velocity at the rigid boundary, which is Equation (6).

Conventional methods for solving the Navier–Stokes equations with rigid immersed boundaries include the body-fitted or structured grid approach. In this approach, the Navier–Stokes equations are discretized on a curvilinear grid that conforms to the immersed boundary and hence the boundary conditions can be imposed easily. The disadvantage of this method is that robust grid generation is required to account for the complexity of the immersed boundaries.

An alternative approach for solving complex viscous flows is the Cartesian grid method that solves the governing equations on a Cartesian grid and has the advantages of retaining the simplicity of the Navier–Stokes equations on the Cartesian coordinates and enabling the use of fast solvers. One of the most successful Cartesian grid methods is Peskin's immersed boundary method [3]. This method was originally developed to study the fluid dynamics of blood flow in the human heart [4]. The method was further developed and has been applied to many biological problems involving flexible boundaries [5, 6]. The immersed boundary method has also been applied to handle problems with immersed boundaries [1, 7]. In order to deal with rigid immersed boundaries, Lai and Peskin [1] proposed to evaluate the force density using an expression of the form

$$\mathbf{f}(s,t)=K_r(\mathbf{X}^e(s)-\mathbf{X}(s,t)) \tag{8}$$

where K_r is a constant, $K_r\gg 1$, $\mathbf{X}$ and $\mathbf{X}^e$ are the arc-parametrization of the computed and the required positions of the boundaries, respectively. The forcing term in Equation (8) is a particular case of the feedback forcing formulation proposed by Goldstein *et al.* [8] with $\beta=0$. In [8], the force is expressed as

$$\mathbf{f}(s,t)=\alpha\int_0^t \mathbf{U}(s,t')\,\mathrm{d}t'+\beta\mathbf{U}(s,t) \tag{9}$$

where $\mathbf{U}$ is the velocity of the boundary, and α and β are chosen to be negative and large enough so that $\mathbf{U}$ will stay close to zero. In order to avoid using very small time step, Mohd-Yusof [9] and Fadlun *et al.* [10] proposed a direct forcing formulation. This forcing is direct in the sense that the exact velocity is imposed directly on the rigid boundary through an interpolation procedure. Lima E Sliva *et al.* [7] proposed a different approach to compute the forcing term $\mathbf{f}$ based on the evaluation of the various terms in the Navier–Stokes equations at the rigid boundary. Another similar approach that combines the original immersed boundary method with the direct and explicit forcing was introduced by Uhlmann [11] for the simulation of particulate flows. The forcing term at the boundary is evaluated based on the desired velocity at the boundary, which is simply given by the rigid-body motion and a preliminary velocity obtained explicitly without the application of a forcing term.

Once the forcing term is obtained at the boundary, the immersed boundary method uses a discrete delta function to spread the force density to the nearby Cartesian grid points. Since the

DOI: 10.1002/fld

immersed boundary method smears out sharp interface to a thickness of order of the mesh width and it is only first-order accurate for problems with non-smooth but continuous solutions. In contrast, the immersed interface method (IIM) can avoid smearing out sharp interfaces and maintains second-order accuracy by incorporating the known jumps into the finite difference scheme near the interface. The IIM was originally proposed by LeVeque and Li [12] for solving elliptic equations, and later extended to Stokes flow with elastic boundaries or surface tension [13]. The interested readers are referred to the newly published book by Li and Ito [14]. The method was further developed for the Navier–Stokes equations in [15–17] for problems with flexible boundaries. Recently, the IIM has been employed to solve for viscous flows with static rigid immersed boundaries [16, 18–20]. In [18, 19], the no-slip boundary conditions are imposed directly by determining the correct jump conditions for stream function and vorticity. In [20], a Cartesian grid method for modeling multiple moving objects in incompressible viscous flow is considered. Le *et al.* [16] have presented an immerse interface method for viscous flows involving rigid and flexible boundaries. In [16], the immersed boundaries are presented by a set of Lagrangian controls points. The strength of singular forces is determined to impose the no-slip condition at the boundary by solving a small system of equations at each time step. Another Cartesian grid approach has been presented by Ye *et al.* [21] and Udaykumar *et al.* [22] via using a finite volume technique. They reshaped the immersed boundary cells and used a polynomial interpolating function to approximate the fluxes and gradients on the faces of the boundary cells while preserving second-order accuracy.

In this work, a level set-based IIM with second-order accuracy is developed for solving the incompressible viscous flows with the prescribed velocity at the boundary, which is based on the approach of Le *et al.* [16] in terms of the evaluation of the forcing term. The method combines the IIM with a level set representation of the interface on a uniform Cartesian grid, which is a further extension of the work reported in [16] where the immersed boundary is represented by a cubic spline interpolation. However, the spline approach is difficult to use for multi-connected domains and for three-dimensional problems. In addition, the level set method usually has better stability. For a spline approach, re-parameterization, filtering, and re-griding may be needed at every or every other time steps. All these reasons favor a level set approach over a spline approach. The numerical implementation of the different interface representation avails the potential reader a choice of the associated numerical techniques. In the proposed method, the singular force at the immersed boundary is determined to enforce the prescribed velocity condition at the boundary. At each time step, the singular force is computed implicitly by solving a small, dense linear system of equations using singular value decomposition (SVD) iterative method. Once the force is computed, next the jump in pressure and jumps in the derivatives of both the pressure and velocity are computed. The Navier–Stokes equations are discretized on a staggered Cartesian grid by a second-order accurate projection method for the pressure and velocity. The jumps in the solution and its derivatives are incorporated into the finite difference discretization to obtain a sharp interface resolution. Fast solvers from the FISHPACK software library [23] are used to solve the resulting discrete systems of equations. The numerical results show that the overall scheme is second-order accurate for the velocity and nearly second-order accurate for the pressure.

The rest of the paper is organized as follows. In Section 2, the jump conditions for the velocity and pressure and their derivatives along the immersed boundary via the singular force $\mathbf{f}$ are presented. The numerical algorithm and numerical implementation are presented in Sections 3 and 4, respectively. In Section 5, some numerical examples are presented. Some concluding remarks will be made in Section 6.

DOI: 10.1002/fld

2. JUMP CONDITIONS ACROSS THE BOUNDARY

With the singular force, the jump conditions for the solutions of the Navier–Stokes equations and their derivatives can be applied and determined. Let $\mathbf{n}=(n_1, n_2)$ and $\boldsymbol{\tau}=(\tau_1, \tau_2)$ be the unit outward normal and tangential vectors to the boundary Γ, respectively. The jump of an arbitrary function $q(\mathbf{X})$ across Γ at $\mathbf{X}$ is denoted by

$$[q]=\lim_{\varepsilon\to 0^+} q(\mathbf{X}+\varepsilon\mathbf{n})-\lim_{\varepsilon\to 0^+} q(\mathbf{X}-\varepsilon\mathbf{n}) \tag{10}$$

Denoting (ξ, η) the local coordinates associated with the directions of $\mathbf{n}$ and $\boldsymbol{\tau}$, respectively, the jump conditions for the velocity and pressure (see [15, 16] for details) are as follows:

$$[\mathbf{u}]=\mathbf{0}, \quad [\mathbf{u}_\eta]=\mathbf{0}, \quad [\mathbf{u}_\xi]=-\frac{1}{\mu}\hat{f}_2\boldsymbol{\tau} \tag{11}$$

$$[\mathbf{u}_{\eta\eta}]=\frac{1}{\mu}\kappa\hat{f}_2\boldsymbol{\tau}, \quad [\mathbf{u}_{\xi\eta}]=-\frac{1}{\mu}\frac{\partial\hat{f}_2}{\partial\eta}\boldsymbol{\tau}-\frac{1}{\mu}\kappa\hat{f}_2\mathbf{n} \tag{12}$$

$$[\mathbf{u}_{\xi\xi}]=-[\mathbf{u}_{\eta\eta}]+\frac{1}{\mu}[p_\xi]\mathbf{n}+\frac{1}{\mu}[p_\eta]\boldsymbol{\tau}-\frac{1}{\mu}[\mathbf{G}^{\text{ext}}] \tag{13}$$

$$[p]=\hat{f}_1, \quad [p_\xi]=[\mathbf{G}^{\text{ext}}]\cdot\mathbf{n}+\frac{\partial\hat{f}_2}{\partial\eta}, \quad [p_\eta]=\frac{\partial\hat{f}_1}{\partial\eta} \tag{14}$$

$$[p_{\eta\eta}]=\frac{\partial^2\hat{f}_1}{\partial\eta^2}-\kappa[p_\xi], \quad [p_{\xi\eta}]=\frac{\partial([\mathbf{G}^{\text{ext}}]\cdot\mathbf{n})}{\partial\eta}+\frac{\partial^2\hat{f}_2}{\partial\eta^2}+\kappa[p_\eta] \tag{15}$$

$$[p_{\xi\xi}]=[\nabla\cdot\mathbf{G}^{\text{ext}}]+v_\eta[u_\xi]-u_\eta[v_\xi]-[p_{\eta\eta}] \tag{16}$$

Here, $\hat{f}_1$ and $\hat{f}_2$ are the components of the force density in the normal and tangential directions of the embedded boundary, denoting $\hat{\mathbf{f}}=(\hat{f}_1, \hat{f}_2)$, and κ is the curvature of the embedded boundary. In this work, the jump conditions of second-order spatial derivative for the pressure are incorporated into the finite difference scheme. It is noted from expressions (11)–(16) that the values of the jumps of the first and second-order derivatives of the velocity and pressure taken with respect to the (x, y) coordinates are easily obtained by a simple coordinate transformation:

$$[q_x]=[q_\xi]n_1+[q_\eta]\tau_1, \quad [q_y]=[q_\xi]n_2+[q_\eta]\tau_2 \tag{17}$$

$$[q_{xx}]=[q_{\xi\xi}]n_1^2+2[q_{\xi\eta}]n_1\tau_1+[q_{\eta\eta}]\tau_1^2 \tag{18}$$

$$[q_{yy}]=[q_{\xi\xi}]n_2^2+2[q_{\xi\eta}]n_2\tau_2+[q_{\eta\eta}]\tau_2^2, \quad q=\mathbf{u}, p \tag{19}$$

3. NUMERICAL ALGORITHM

The numerical algorithm to be employed is based on the pressure-increment projection algorithm for the discretization of the Navier–Stokes equations with special treatment at the grid points near the embedded boundary [16]. The spatial discretization is carried out on a standard marker-and-cell

DOI: 10.1002/fld

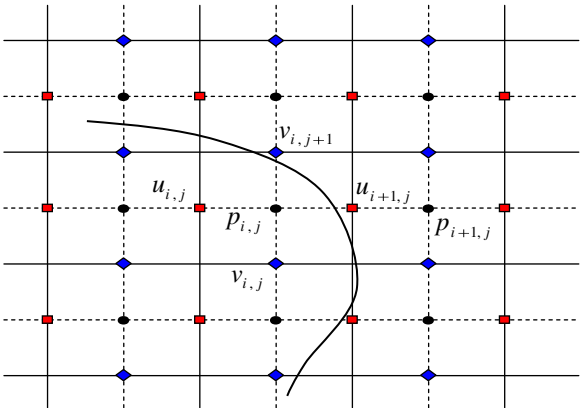

Figure 2. A diagram of the embedded boundary cutting through a staggered grid with a uniform mesh width h, where the velocity component u is at the left–right face of the cell and v is at the top–bottom face, and the pressure is at the cell center.

(MAC) staggered grid similar to that found in Harlow and Welch [24]. A uniform MAC gird with mesh width $h=\Delta x=\Delta y$ is used in the computation. With the MAC mesh, the pressure field is defined at the cell center (i, j), where $i\in\{1,2,\ldots,N_x\}$ and $j\in\{1,2,\ldots,N_y\}$. The velocity fields u and v are defined at the vertical edges and horizontal edges of a cell, respectively. The pressure and the velocity components u and v are arranged as in Figure 2.

3.1. Projection method

Assuming that the force $\mathbf{f}$ at the immersed boundary is known, the pressure-increment procedure for problems with immersed interfaces is analogous to the projection method presented in [25]. The discretization of the Navier–Stokes equations at those grid points near the embedded boundary needs to be modified to account for the jump conditions across the boundary due to the presence of singular forces at the boundary. The discreted expressions will include coefficients $\mathbf{C}_1$, $\mathbf{C}_2$, etc., which will be evaluated later. Given the velocity $\mathbf{u}^n$ and the pressure $p^{n-1/2}$, the velocity $\mathbf{u}^{n+1}$ and pressure $p^{n+1/2}$ at the next time step are computed as follows:

Step 1: Compute an intermediate velocity field $\mathbf{u}^*$ by solving

$$\frac{\mathbf{u}^*-\mathbf{u}^n}{\Delta t}+(\mathbf{u}\cdot\nabla\mathbf{u})^{n+1/2}=-\frac{1}{\rho}\nabla p^{n+1/2}+\frac{\mu}{2\rho}(\Delta_h\mathbf{u}^*+\Delta_h\mathbf{u}^n)+\frac{1}{\rho}\mathbf{g}^{n+1/2}+\mathbf{C}_1 \tag{20}$$

$$\mathbf{u}^*|_{\partial\Omega}=\mathbf{u}_b^{n+1} \tag{21}$$

where the advection term is extrapolated using the formula

$$(\mathbf{u}\cdot\nabla\mathbf{u})^{n+1/2}=\tfrac{3}{2}(\mathbf{u}\cdot\nabla_h\mathbf{u})^n-\tfrac{1}{2}(\mathbf{u}\cdot\nabla_h\mathbf{u})^{n-1}+\mathbf{C}_2 \tag{22}$$

and the pressure gradient is approximated simply as

$$\nabla p^{n+1/2}=G^{\mathrm{MAC}}p^{n-1/2}+\mathbf{C}_3 \tag{23}$$

DOI: 10.1002/fld

The above step can be rewritten in the following Helmholtz equations form:

$$\lambda_0 \mathbf{u}^* + \Delta_h \mathbf{u}^* = \text{RHS} \tag{24}$$

where RHS is the right hand, which includes the correction terms and $\lambda_0 = -2\rho/\mu\Delta t$.

Step 2: Compute a pressure update ϕ^{n+1} by solving the Poisson equation

$$\Delta_h \phi^{n+1} = \frac{\rho}{\Delta t} D^{\text{MAC}} \mathbf{u}^* + C_4 \tag{25}$$

with boundary condition

$$\mathbf{n} \cdot \nabla \phi^{n+1}|_{\partial\Omega} = 0$$

Step 3: Once ϕ^{n+1} is obtained by solving Equation (25), both the pressure and velocity field $(p^{n+1/2}, \mathbf{u}^{n+1})$ are updated as

$$\mathbf{u}^{n+1} = \mathbf{u}^* - \frac{\Delta t}{\rho} G^{\text{MAC}} \phi^{n+1} + \mathbf{C}_5 \tag{26}$$

$$p^{n+1/2} = p^{n-1/2} + \phi^{n+1} - \frac{\mu}{2\rho} D^{\text{MAC}} \mathbf{u}^* + C_6 \tag{27}$$

The coefficients $\mathbf{C}_1, \mathbf{C}_2, \mathbf{C}_3, C_4, \mathbf{C}_5$, and C_6 are the spatial correction terms, which are added to the finite difference equations at the points near the boundary to improve the accuracy of the local finite difference approximations. These correction terms can be computed by using the generalized finite difference formulas if the jumps in the solution and their derivatives are known and will be evaluated later. In the above expressions, ∇_h and Δ_h are the standard central difference operators, G^{MAC} and D^{MAC} are the MAC gradient and divergence operators, respectively. These operators are defined as

$$\nabla_h \mathbf{u}_{i,j} = \left(\frac{\mathbf{u}_{i+1,j} - \mathbf{u}_{i-1,j}}{2h}, \frac{\mathbf{u}_{i,j+1} - \mathbf{u}_{i,j-1}}{2h} \right) \tag{28}$$

$$\Delta_h \mathbf{u}_{i,j} = \frac{\mathbf{u}_{i+1,j} + \mathbf{u}_{i-1,j} + \mathbf{u}_{i,j+1} + \mathbf{u}_{i,j-1} - 4\mathbf{u}_{ij}}{h^2} \tag{29}$$

$$(G^{\text{MAC}} p)_{ij} = \left(\frac{p_{i+1,j} - p_{i,j}}{h}, \frac{p_{i,j+1} - p_{i,j}}{h} \right) \tag{30}$$

$$(D^{\text{MAC}} \mathbf{u})_{i,j} = \frac{u_{i+1,j} - u_{i,j}}{h} + \frac{v_{i,j+1} - v_{i,j}}{h} \tag{31}$$

In our projection method, two Helmholtz equations for $\mathbf{u}^*$ in (21) or (24) and one Poisson-like equation for ϕ^{n+1} in (25) need to be solved at each time step. Since the correction terms in (21) or (24) and (25) only affect the right-hand sides of the discrete systems for the Helmholtz and Poisson equations, there are many methods which can be used to solve these linear system, for example, the conjugate gradient method and multigrid method. In the present work, the fast solvers from FISHPACK [23] are used to solve these equations.

DOI: 10.1002/fld

3.2. Correction terms calculation

In this section, the correction terms $\mathbf{C}_1$, $\mathbf{C}_2$, $\mathbf{C}_3$, C_4, $\mathbf{C}_5$, and C_6 will be evaluated. One of the basic components for determining the correction terms is the generalized finite difference formulas which will be briefly reviewed in this section. Here four generalized finite difference formulas are shown for demonstration. Assume that the boundary cuts a grid line between two grid points at $x=\alpha$, $x_i \leqslant \alpha < x_{i+1}$, $x_i \in \Omega^-$, $x_{i+1} \in \Omega^+$, where Ω^- and Ω^+ denote the region inside and outside the boundary, respectively. Then, the following approximations hold for a piecewise twice differentiable function $q(x)$:

$$q_x(x_i) = \frac{q_{i+1} - q_{i-1}}{2h} - \frac{1}{2h} \sum_{m=0}^{2} \frac{(h^+)^m}{m!} [q^{(m)}]_\alpha + O(h^2) \tag{32a}$$

$$q_{xx}(x_i) = \frac{q_{i+1} - 2q_i + q_{i-1}}{h^2} - \frac{1}{h^2} \sum_{m=0}^{2} \frac{(h^+)^m}{m!} [q^{(m)}]_\alpha + O(h) \tag{32b}$$

where $q^{(m)}$ denotes the mth derivative of q, $q_i = q(x_i)$, $h^+ = x_{i+1} - \alpha$, $h^- = x_i - \alpha$ and h is the mesh width in x direction. The jump in q and its derivatives are defined as

$$[q^{(m)}]_\alpha = \lim_{x \to \alpha, x \in \Omega^+} q^{(m)}(x) - \lim_{x \to \alpha, x \in \Omega^-} q^{(m)}(x) \tag{33}$$

in short, $[\cdot] = [\cdot]_\alpha$, and $q^{(0)} = q$. Note that if the boundary cuts a grid line between two grid points $x_i \in \Omega^+$ and $x_{i+1} \in \Omega^-$, these expressions need to be modified by changing the sign of the second terms on the respective right-hand sides. Expressions involving two or more boundary crossings could also be derived, the readers are referred to [26] for details. From Equations (32a) and (32b) the correction terms for $q_x(x_i)$ and $q_{xx}(x_i)$ can be defined as

$$C\{q_x(x_i)\} = -\frac{1}{2h} \sum_{m=0}^{2} \frac{(h^+)^m}{m!} [q^{(m)}] \tag{34}$$

$$C\{q_{xx}(x_i)\} = -\frac{1}{h^2} \sum_{m=0}^{2} \frac{(h^+)^m}{m!} [q^{(m)}] \tag{35}$$

Thus, the finite difference approximation near the boundary, for the derivatives of a function q, includes the standard central difference terms plus the additional correction terms. Accordingly, the correction terms $\mathbf{C}_1$, $\mathbf{C}_2$, $\mathbf{C}_3$, C_4, $\mathbf{C}_5$ and C_6 are evaluated as follows:

$$\mathbf{C}_1 = \frac{1}{2\rho}(C\{\Delta \mathbf{u}^*\} + C\{\Delta \mathbf{u}^n\}) \tag{36a}$$

$$\mathbf{C}_2 = \tfrac{3}{2} C\{(\mathbf{u} \cdot \nabla \mathbf{u})^n\} - \tfrac{1}{2} C\{(\mathbf{u} \cdot \nabla \mathbf{u})^{n-1}\} \tag{36b}$$

$$\mathbf{C}_3 = C\{\nabla p^{n-1/2}\} \tag{36c}$$

$$C_4 = \rho \frac{C\{\nabla \cdot \mathbf{u}^*\}}{\Delta t} - C\{\nabla(\nabla p^{n+1/2})\} + C\{\nabla(\nabla p^{n-1/2})\} \tag{36d}$$

DOI: 10.1002/fld

$$\mathbf{C}_5 = -\frac{\Delta t}{\rho}(C\{\nabla p^{n+1/2}\} - C\{\nabla p^{n-1/2}\}) \tag{36e}$$

$$C_6 = -\frac{\mu}{2\rho} C\{\nabla \cdot \mathbf{u}^*\} \tag{36f}$$

It is noted that all the correction terms are evaluated at least for first-order accuracy. This is sufficient to guarantee second-order accuracy globally since our numerical scheme is second order away from the boundary and only the irregular points near the boundary are treated with a first-order scheme. In (36a), (36d), and (36f), the jump conditions for $\mathbf{u}^{n+1}$ are used to approximate the jump conditions for $\mathbf{u}^*$ as it is expected that $\mathbf{u}^*$ is a good approximation for $\mathbf{u}^{n+1}$. To evaluate the correction term $C\{\Delta\mathbf{u}^*\}$ of (36a) at a point (i, j) as depicted in Figure 3, $[\mathbf{u}_x^*]$ and $[\mathbf{u}_{xx}^*]$ at the intersection point α, and $[\mathbf{u}_y^*]$ and $[\mathbf{u}_{yy}^*]$ at β using the force strength at time level $n+1$ need to be computed. The correction term $C\{\Delta\mathbf{u}^*\}$ is calculated as follows:

$$C\{\Delta\mathbf{u}^*\}_{i,j} = -\frac{[\mathbf{u}^*] + h^+[\mathbf{u}_x^*]_\alpha + \dfrac{(h^+)^2}{2}[\mathbf{u}_{xx}^*]_\alpha}{h^2} - \frac{[\mathbf{u}^*] + k^-[\mathbf{u}_y^*]_\beta + \dfrac{(k^-)^2}{2}[\mathbf{u}_{yy}^*]_\beta}{h^2}$$

where $h^+ = x_{i+1} - x_\alpha$, $k^- = y_{j-1} - y_\beta$, and x_α and y_β are the x-coordinate of the intersection point α and the y-coordinate of the intersection point β as shown in Figure 3, respectively. $\Delta\mathbf{u}^*$ is approximated at the irregular point (i, j) as

$$\Delta\mathbf{u}^*(i, j) = \Delta_h \mathbf{u}_{i,j}^* + C\{\Delta\mathbf{u}^*\}_{i,j} + O(h)$$

Similarly, other correction terms in (36b)–(36f) can be computed as follows:

$$C\{\nabla \cdot \mathbf{u}\}_{i,j} = -\frac{[u] + h^+[u_x]_\alpha + \dfrac{(h^+)^2}{2}[u_{xx}]_\alpha}{h} + \frac{[v] + k^-[v_y]_\beta + \dfrac{(k^-)^2}{2}[v_{yy}]_\beta}{h}$$

$$C\{\nabla p\}_{i,j} = \left(-\frac{[p] + h^+[p_x]_\alpha + \dfrac{(h^+)^2}{2}[p_{xx}]_\alpha}{h}, \frac{[p] + k^-[p_y]_\beta + \dfrac{(k^-)^2}{2}[p_{yy}]_\beta}{h} \right)$$

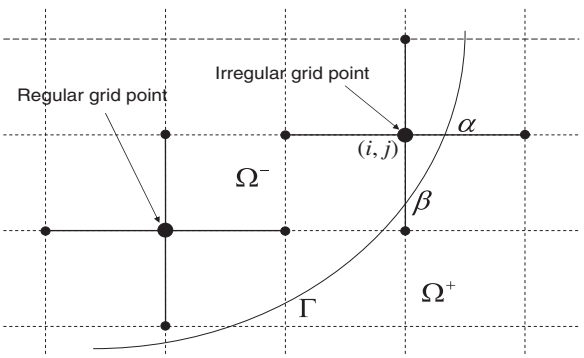

Figure 3. The embedded boundary and mesh geometry near the irregular grid point (i, j).

DOI: 10.1002/fld

3.3. Level set representation of boundaries

The zero level set of a two-dimensional function $\varphi(x, y)$ is used to represent the boundary. That is, $\varphi(x, y)$ is such a function such that

$$\Gamma = \{(x, y) : \varphi(x, y) = 0\} \tag{37}$$

Generally, $\varphi(x, y)$ is chosen as the signed distance function from the boundary. $\Omega^+ = \{(x, y), |\, \varphi(x, y) \geqslant 0\}$ and $\Omega^- = \{(x, y), |\, \varphi(x, y) < 0\}$ are denoted. In this work, the level set function is defined at the same location with the pressure (at the cell center), i.e $\varphi_{i,j} = \varphi(x_{i+1/2}, y_{j+1/2})$. A grid point $(x_{i+1/2}, y_{j+1/2})$ is called an *irregular inner grid point* if $\varphi_{i,j} < 0$ and any of the following four inequalities is true:

$$\begin{aligned} \varphi_{i,j} \cdot \varphi_{i-1,j} &\leqslant 0, \quad \varphi_{i,j} \cdot \varphi_{i+1,j} \leqslant 0 \\ \varphi_{i,j} \cdot \varphi_{i,j-1} &\leqslant 0, \quad \varphi_{i,j} \cdot \varphi_{i,j+1} \leqslant 0 \end{aligned} \tag{38}$$

Similarly, a grid point $(x_{i+1/2}, y_{j+1/2})$ is an *irregular outer grid point* if $\varphi_{i,j} \geqslant 0$ and any of the following four inequalities is true:

$$\begin{aligned} \varphi_{i,j} \cdot \varphi_{i-1,j} &< 0, \quad \varphi_{i,j} \cdot \varphi_{i+1,j} < 0 \\ \varphi_{i,j} \cdot \varphi_{i,j-1} &< 0, \quad \varphi_{i,j} \cdot \varphi_{i,j+1} < 0 \end{aligned} \tag{39}$$

Then the projection points $\{\mathbf{X}^*\} = \{(X^*, Y^*)\}$ of the irregular grid points onto the boundary are computed, as illustrated in Figure 4. Assume that $\mathbf{x}_{i+1/2,j+1/2}$ is an irregular grid point. The corresponding orthogonal projection point on the boundary satisfies

$$\mathbf{X}^* = \mathbf{x}_{i+1/2,j+1/2} + \gamma \mathbf{w} \tag{40}$$

where $\mathbf{w} = (\partial \varphi_{ij}/\partial x, \partial \varphi_{ij}/\partial y)$. Using the Taylor expansion of $\varphi(\mathbf{X}^*) = \varphi(\mathbf{x}_{ij} + \gamma \mathbf{w}) = 0$ at $\mathbf{x}_{ij}$, the following quadratic equation for the unknown variable γ is obtained:

$$\varphi_{ij} + (\nabla \varphi_{ij} \cdot \mathbf{w})\gamma + \tfrac{1}{2}(\mathbf{w}^{\mathrm{T}} \mathbf{H}(\varphi_{ij})\mathbf{w})\gamma^2 = 0 \tag{41}$$

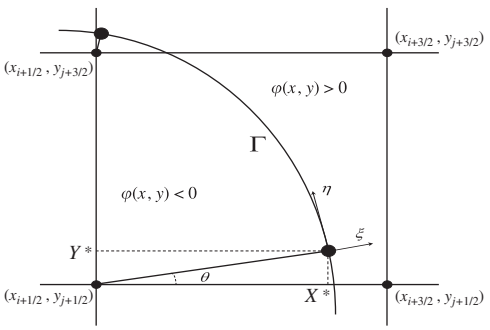

Figure 4. The corresponding projection point $\mathbf{X}^* = (X^*, Y^*)$ of an irregular grid point $\mathbf{x} = (x_i, y_j)$ on the boundary Γ, where (ξ, η) is the local coordinate system at $\mathbf{X}^*$.

DOI: 10.1002/fld

where the Hessian matrix $\mathbf{H}$ is defined as

$$\mathbf{H}(\varphi_{ij}) = \begin{pmatrix} \dfrac{\partial^2 \varphi_{ij}}{\partial x^2} & \dfrac{\partial^2 \varphi_{ij}}{\partial x \partial y} \\ \dfrac{\partial^2 \varphi_{ij}}{\partial y \partial x} & \dfrac{\partial^2 \varphi_{ij}}{\partial y^2} \end{pmatrix} \tag{42}$$

The detailed algorithm for finding the projection points is explained in [27, 28]. The spatial derivatives of the level set function are approximated using the following central finite difference schemes at $(x_{i+1/2}, y_{j+1/2})$:

$$\frac{\partial \varphi_{ij}}{\partial x} \approx \frac{\varphi_{i+1,j} - \varphi_{i-1,j}}{2h}, \quad \frac{\partial \varphi_{ij}}{\partial y} \approx \frac{\varphi_{i,j+1} - \varphi_{i,j-1}}{2h} \tag{43}$$

$$\frac{\partial^2 \varphi_{ij}}{\partial x^2} \approx \frac{\varphi_{i+1,j} - 2\varphi_{i,j} + \varphi_{i-1,j}}{h^2}, \quad \frac{\partial^2 \varphi_{ij}}{\partial y^2} \approx \frac{\varphi_{i,j+1} - 2\varphi_{i,j} + \varphi_{i,j-1}}{h^2} \tag{44}$$

$$\frac{\partial^2 \varphi_{ij}}{\partial x \partial y} = \frac{\partial^2 \varphi_{ij}}{\partial y \partial x} \approx \frac{\varphi_{i+1,j+1} + \varphi_{i-1,j-1} - \varphi_{i+1,j-1} - \varphi_{i-1,j+1}}{4h^2} \tag{45}$$

At each orthogonal projection point $\mathbf{X}^*$, the geometrical information needed includes the curvature κ and the angle θ between the normal direction and the x-axis at $\mathbf{X}^*$. The curvature is defined as usual

$$\kappa = -\frac{\dfrac{\partial^2 \varphi}{\partial x^2}\left(\dfrac{\partial \varphi}{\partial y}\right)^2 - 2\dfrac{\partial^2 \varphi}{\partial x \partial y}\dfrac{\partial \varphi}{\partial x}\dfrac{\partial \varphi}{\partial y} + \dfrac{\partial^2 \varphi}{\partial y^2}\left(\dfrac{\partial \varphi}{\partial x}\right)^2}{\left(\left(\dfrac{\partial \varphi}{\partial x}\right)^2 + \left(\dfrac{\partial \varphi}{\partial y}\right)^2\right)^{3/2}} \tag{46}$$

The unit normal direction is determined by

$$n_1 = \frac{\dfrac{\partial \varphi}{\partial x}}{\sqrt{\left(\dfrac{\partial \varphi}{\partial x}\right)^2 + \left(\dfrac{\partial \varphi}{\partial y}\right)^2}}, \quad n_2 = \frac{\dfrac{\partial \varphi}{\partial y}}{\sqrt{\left(\dfrac{\partial \varphi}{\partial x}\right)^2 + \left(\dfrac{\partial \varphi}{\partial y}\right)^2}} \tag{47}$$

The derivatives of φ at $\mathbf{X}^*$ can be approximated using the bilinear interpolation formula from the derivatives already calculated at four neighboring grid points using (43)–(45). Similarly, κ and $\mathbf{n}$ at $\mathbf{X}^*$ can be calculated.

3.4. Calculation of surface derivatives

To determine the above correction terms (36a)–(36f), a key part is to compute the first and second derivatives of the force $\mathbf{f}$ along the boundary. It is assumed that the values of the boundary function $\phi(s)$ are known at all the projection points $\{\mathbf{X}_K^*\}$ from the inner irregular grid points and below is

DOI: 10.1002/fld

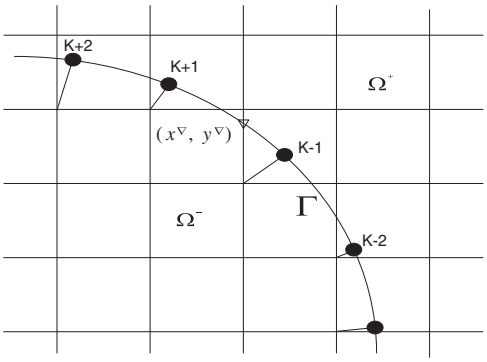

Figure 5. A diagram used to interpolate the surface derivatives at the intersections of the boundary Γ and grid lines, where dark circles represent the projection points from the inner irregular grid points.

how to interpolate $\phi(s)$, $\phi'(s)$, and $\phi''(s)$ at the intersections of the boundary and grid lines, and $\phi'(s)$ and $\phi''(s)$ at the projection points from the inner irregular grid points, see Figure 5 for an illustration. The least-squares interpolation scheme for approximating $\phi(s^\nabla)$ can be written as

$$\sum_k \gamma_k \phi(s_k) \tag{48}$$

where γ_k's are the coefficients, which need to be determined from the Taylor expansion at the interpolation point s^∇

$$\phi(s_k) = \phi(s^\nabla) + \phi'(s^\nabla)\Delta s_k + \tfrac{1}{2}\phi''(s^\nabla)(\Delta s_k)^2 + O((\Delta s_k)^3) \tag{49}$$

where $\Delta s_k = s_k - s^\nabla$ is the signed arc-length from the boundary point at $s = s_k$ to the point at $s = s^\nabla$. Defining

$$a_1 = \sum_k \gamma_k, \quad a_2 = \sum_k (s_k - s^\nabla)\gamma_k, \quad a_3 = \sum_k \tfrac{1}{2}(s_k - s^\nabla)^2 \gamma_k \tag{50}$$

and substituting (49) into (48), the coefficients γ_k can be obtained by setting

$$a_1 = 1, \quad a_2 = 0, \quad a_3 = 0 \tag{51}$$

In order to keep the symmetry strictly, in our computations, the first closest two projection points to the intersection point $(X(s^\nabla), Y(s^\nabla))$ from each side of this point are chosen for the interpolation, respectively, then the SVD method is used to solve the system of equations (51). Similarly, other surface derivatives $\phi'(s^\nabla)$ and $\phi''(s^\nabla)$ can be obtained.

3.5. Determination of force **f** *at projection points*

If the force **f** at the immersed boundary is known as assumed before, the velocity field $\mathbf{u}^{n+1}$ at all the grid points can be computed via the projection method as discussed in Section 3.1. The velocity at the projection points from the outside of the boundary, $\mathbf{U}_k$, can be interpolated from the velocity $\mathbf{u}^{n+1}$ at the grid points as in [16]. Thus, it can be written as

$$\mathbf{U}_k = \mathbf{U}(\mathbf{X}_k) = \mathcal{I}(\mathbf{u}^{n+1}) \tag{52}$$

DOI: 10.1002/fld

where $\mathscr{I}$ is the interpolation operator that includes the appropriate correction terms required to guarantee at least second-order accuracy when the derivatives of the velocity are discontinuous. Two ways are to use the third-order accurate least-square interpolation scheme in [14] and the modified bilinear interpolation with jump conditions in [16]. As an alternative approach, three nearby points are used to perform linear interpolation that is modified to incorporate the jump condition at the boundary. The readers are referred to [13] for details. Since the relationships between the singular forces and the jumps in the solution or its derivatives are linear and all the implicit equations solved at each time step of the projection method are linear, the velocity at the immersed boundary can be written as

$$\mathbf{U}_k = \mathbf{U}_k^0 + \mathbf{A}\mathbf{f} \tag{53}$$

where $\mathbf{U}_k^0$ corresponds to the velocity at the projection points obtained by solving Equations (4) and (5) with $\mathbf{f}=0$, given $\mathbf{u}^n$ and $p^{n-1/2}$. $\mathbf{A}$ is a $2N_b \times 2N_b$ matrix, where N_b is the number of projection points. The vector $\mathbf{Af}$ is the velocity at the projection points obtained by solving the following equations:

$$\frac{\mathbf{u}_\mathbf{f}^*}{\Delta t} = \frac{\mu}{2\rho}\Delta_h \mathbf{u}_\mathbf{f}^* + \bar{\mathbf{C}}_1, \quad \mathbf{u}_\mathbf{f}^*|_{\partial\Omega} = 0 \tag{54}$$

$$\Delta_h \phi_\mathbf{f}^{n+1} = \frac{\rho}{\Delta t} D^{\mathrm{MAC}} \mathbf{u}_\mathbf{f}^* + \bar{C}_4, \quad \mathbf{n}\cdot\nabla\phi_\mathbf{f}^{n+1}|_{\partial\Omega} = 0 \tag{55}$$

$$\mathbf{u}_\mathbf{f}^{n+1} = \mathbf{u}_\mathbf{f}^* - \frac{\Delta t\,\mu}{\rho} G^{\mathrm{MAC}} \phi_\mathbf{f}^{n+1} + \bar{\mathbf{C}}_5 \tag{56}$$

$$\mathbf{Af} = \mathscr{I}(\mathbf{u}_\mathbf{f}^{n+1}) \tag{57}$$

with $\mathbf{f}$ being the singular force at the immersed boundary. Here, $\bar{\mathbf{C}}_1$, $\bar{C}_4$, and $\bar{\mathbf{C}}_5$ are the correction terms that take into account the effect of the singular force $\mathbf{f}$ at the immersed boundary. From Equation (53), with the prescribed velocity $\mathbf{U}_p$ at the immersed boundary, the singular force $\mathbf{f}$ at the immersed boundary is determined by solving

$$\mathbf{Af} = \mathbf{U}_p - \mathbf{U}_k^0 \tag{58}$$

Note that the matrix $\mathbf{A}$ depends on the location of the boundary and the time step Δt. If the boundary and the time step are fixed, the matrix $\mathbf{A}$ will be same at every time step. Therefore, the matrix $\mathbf{A}$ needs to be formed and factorized only once in this case. In order to compute the coefficients of $\mathbf{A}$, Equations (54)–(56) are solved for $2N_b$ times, i.e. once for each column. Each time, the singular force $\mathbf{f}$ is set to zero except for the entry corresponding to the column that is to be calculated, which is set to one. Once the matrix $\mathbf{A}$ has been calculated, only the terms on the right-hand side, $\mathbf{U}_p - \mathbf{U}_k^0$, need to be computed at each time step. The resulting small system of Equation (58) is then solved at each time step for $\mathbf{f}$ via back substitution. Finally, Equations (21)–(27) are solved to obtain $\mathbf{u}^{n+1}$ and $p^{n+1/2}$. In actual computation, the SVD method is used to solve the system of Equation (58).

DOI: 10.1002/fld

4. NUMERICAL IMPLEMENTATION

In this section, a basic implementation of the proposed algorithm is described. The coefficient matrix using SVD is factorized as

$$A = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\mathrm{T}} \tag{59}$$

where $\mathbf{U}=[u_1,\ldots,u_{N_b}]$ and $\mathbf{V}=[v_1,\ldots,v_{N_b}]$ are orthogonal matrices and $\mathbf{\Sigma}=\mathrm{diag}(\sigma_1,\ldots,\sigma_{N_b})$ is a diagonal matrix whose elements are the singular values of the original matrix such that

$$\sigma_1 \geqslant \sigma_2 \geqslant \cdots \geqslant \sigma_{N_b} \geqslant 0 \tag{60}$$

$\mathbf{U}, \mathbf{V}$, and $\mathbf{\Sigma}$ are stored for solving the force $\mathbf{f}$ at every time step. At each time step, given the velocity field $\mathbf{u}^n$ and pressure field $p^{n-1/2}$, the proposed algorithm for finding $\mathbf{u}^{n+1}$, $p^{n-1/2}$, and the singular force $\mathbf{f}$ to enforce the prescribed velocity $\mathbf{U}_p$ at the immersed boundary can be summarized as follows:

Algorithm The implementation of the IIM with the prescribed velocity at the boundary

Step 1: Compute the right-hand side of (58) by calculating $\mathbf{U}_p - \mathbf{U}_k^0$.

- Set $\mathbf{f}=0$, and solve (24)–(26) for the velocity at all the grid points.
- Interpolate the velocity at the projection points $\mathbf{U}_k^0$ as in (52).
- Compute the right-hand side vector $\mathbf{b}=\mathbf{U}_p-\mathbf{U}_k^0$.

Step 2: Compute the singular force $\mathbf{f}$ by solving (58) using the SVD method. The singular force $\mathbf{f}$ can be written in terms of the SVD as

$$\mathbf{w} = \sum_{i=1}^{N_b} \frac{u_i^{\mathrm{T}}\mathbf{b}}{\sigma_i} v_i \tag{61}$$

Step 3: Compute $\mathbf{u}^{n+1}$ and $p^{n+1/2}$ using the projection method with the incorporation of the appropriate correction terms.

5. NUMERICAL EXAMPLES

In this section, several numerical experiments are carried out to demonstrate the capabilities and the accuracy of the proposed algorithm in this work. Throughout this section, ρ is taken to be 1 in all simulations.

Example 5.1 (*Circular Couette flow*)
In the first example, the steady circular Couette flow between two rotating translating concentric cylinders is considered. The domain of the simulation and the geometry of the two rotating concentric cylinders are shown in Figure 6. The solution domain is the rectangle of the size $[-a_x/2, a_x/2]\times[-b_x/2, b_x/2]$. The angular velocities of the inner and outer cylinders are denoted

DOI: 10.1002/fld

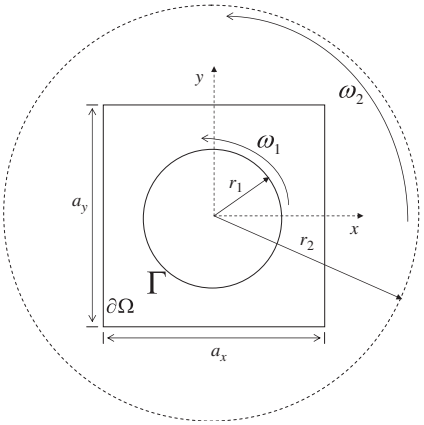

Figure 6. The domain of the simulation and the geometry of circular Couette flow.

as ω_1 and ω_2, respectively. In the simulations, $r_1=0.5,\ r_2=2.0,\ a_x=a_y=2$. The analytical solution of the steady flow between the two cylinders is given by

$$u=-\left(A+\frac{B}{r^2}\right)y \tag{62}$$

$$v=\left(A+\frac{B}{r^2}\right)x \tag{63}$$

$$p=\frac{A^2r^2}{2}-\frac{B^2}{2r^2}+AB\cdot\ln(r^2)+p_0 \tag{64}$$

where $r=\sqrt{x^2+y^2},\ p_0$ is an arbitrary constant, and A_1 and A_2 are

$$A=\frac{\omega_2r_2^2-\omega_1r_1^2}{r_2^2-r_1^2} \tag{65}$$

$$B=\frac{(\omega_1-\omega_2)r_1^2r_2^2}{r_2^2-r_1^2} \tag{66}$$

The Dirichlet boundary conditions are applied for the velocity at the far-field boundary $\partial\Omega$ in Figure 6, and they are obtained from the analytical solution. In the current numerical setup, only the inner cylinder Γ is contained in the simulation domain. It is easy to verify that the velocity satisfies the incompressibility constraint, and it is continuous but has a finite jump in the normal direction across the boundary.

In our computations, the simulation is performed with a 64×64 grid, and the time step is taken as $\Delta t=\Delta x/4$. We first take $\omega_1=1,\ \omega_2=-1$, and $\mu=0.1$. In Figure 7, the plots for the computed u-component velocity and the velocity field at $t=10$ are presented. From Figure 7(a), it can be seen that the velocity u is continuous but not smooth, as expected. In this case, the grid refinement

DOI: 10.1002/fld

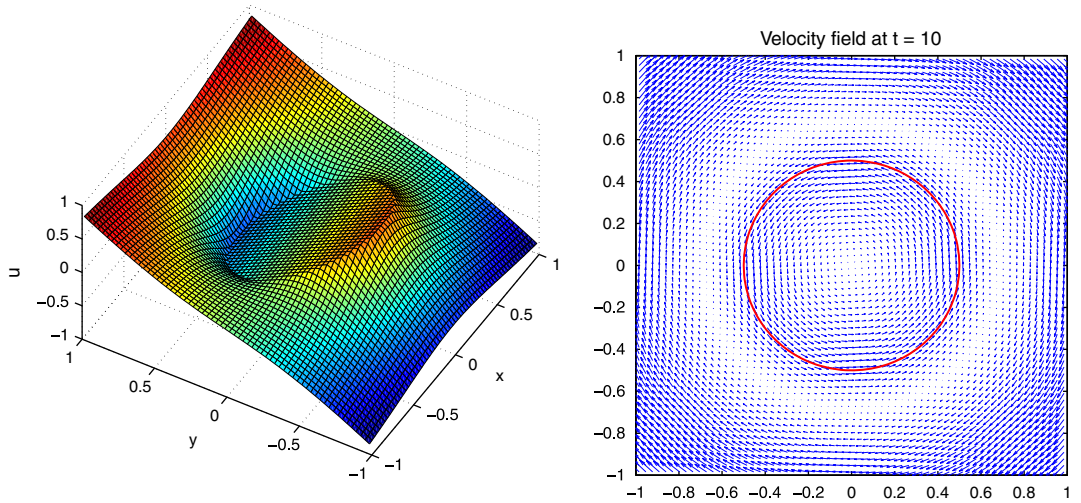

Figure 7. For Example 5.1 on circular Couette flow: (a) the x-component of velocity field $\mathbf{u}$ and (b) the velocity field $\mathbf{u}$ with $\omega_1 = 1, \omega_2 = -1$ and $\mu = 0.1$ at $t = 10$.

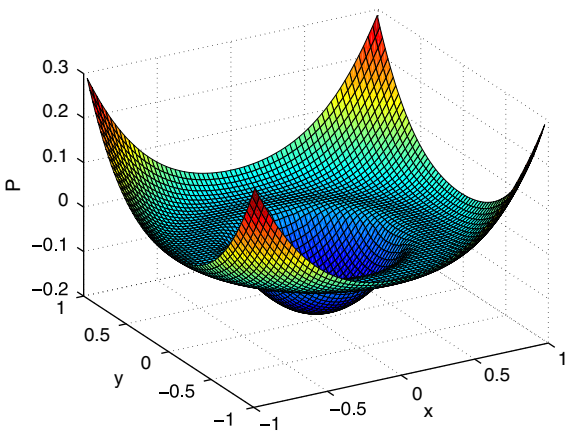

Figure 8. For Example 5.1 on circular Couette flow. Pressure distribution with $\omega_1 = 1, \omega_2 = -1$ and $\mu = 0.1$ at $t = 10$.

analysis is performed to determine the order of convergence of the algorithm. The order of accuracy is estimated as (Figure 8)

$$\text{order} = \frac{\log(\| E_u(N) \|_\infty / \| E_u(2N) \|_\infty)}{\log 2} \tag{67}$$

DOI: 10.1002/fld

Table I. Grid refinement analysis for Example 5.1 with $\omega_1=1, \omega_2=-1$ and $\mu=0.1$ at $t=2$.

N	$\| E_{\mathbf{u}} \|_\infty$	Order	$\| E_p \|_\infty$	Order
32	1.2114E−02	—	1.5329E−02	—
64	2.7106E−03	2.16	4.1359E−03	1.89
128	6.5912E−04	2.04	1.1315E−03	1.87
256	1.5698E−04	2.07	3.0108E−04	1.91

here, $\| E_u(N) \|_\infty$ is the maximum error

$$\| E_u(N) \|_\infty = \max_{i,j} |U_{ij} - u(x_i, y_j)| \tag{68}$$

where $u(x_i, x_j)$ is the exact solution at (x_i, x_j) and U_{ij} is the numerical solution.

The result of the convergence rate analysis is shown in Table I. From Table I, one can easily see that the velocity is second-order accurate, and the pressure is nearly second-order accurate. Next we take $\omega_1=2,\ \omega_2=-2$, and a relatively low viscosity $\mu=0.001$. The corresponding computed u-component velocity and the velocity field at $t=10$ are presented in Figure 9. It demonstrates that the method is stable at the relatively low viscous viscosity.

Example 5.2 (*Rotational flow*)
In this example, a fixed interface problem with no exact solution taken from [16] is considered. The interface is a circle with radius $r=\frac{1}{2}$, which is located at the center of the square domain $[-1,1]\times[-1,1]$. The initial velocity and pressure are taken to be zero on the square domain. On the boundary of Ω, the no-slip boundary conditions are set. The boundary is prescribed to rotate with angular velocity $\omega=2$. The viscosity μ is set to be 0.02, and the solution at $t=10$ is considered.

In the computations, a 64×64 grid is used and the time step $\Delta t=\Delta x/4$ is taken. The flow converges to a steady state in the end, as shown in Figure 10, which corresponds to a rigid-body motion inside the interface. Figures 10(a) and (b) show the x-component of the velocity field $\mathbf{u}$ and the velocity field at $t=10$, respectively. From Figure 10(a) it can be observed that the velocity u is continuous but not smooth, as expected. The plot of the pressure at $t=10$ is presented in Figure 11. Finally, a grid refinement analysis is carried, using a referenced grid of 512×512, to determine the order of the convergence of the algorithm. The results in Table II indicate that the velocity is second-order accurate and the pressure is nearly second-order accurate.

Example 5.3 (*Flow past a stationary cylinder*)
In this example, an unsteady flow past a circular cylinder of diameter $d=0.1$ immersed in a rectangular domain $\Omega=[0,3]\times[0,1.5]$ is simulated. The center of the cylinder is located at $(1.6, 0.75)$. The free stream velocity is set to unity, $U_\infty=1$. Simulations are carried out at Reynolds number ($Re=U_\infty d/\mu$) of 20, 40, 100, and 200 on a 512×256 computational mesh. The free stream velocity at the domain inlet is specified and a homogeneous Neumann boundary condition for the velocity at the domain outlet is applied. The homogeneous Neumann boundary condition and homogeneous Dirichlet boundary condition are set for the x-component and y-component of the velocity, respectively, at the top and bottom boundaries. The homogeneous Neumann boundary

DOI: 10.1002/fld

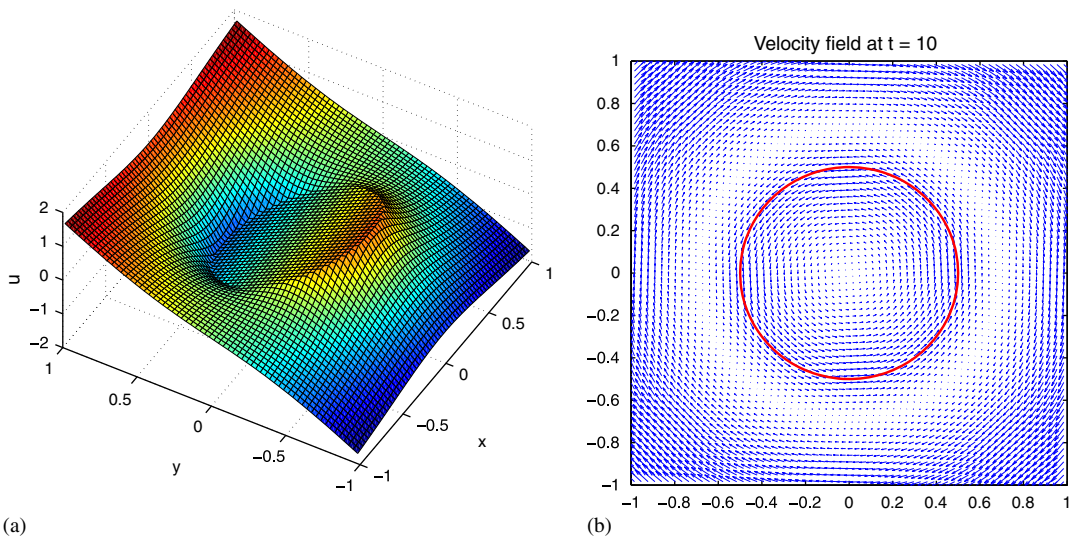

Figure 9. For Example 5.1 on circular Couette flow: (a) the x-component of velocity field $\mathbf{u}$ and (b) the velocity field $\mathbf{u}$ with $\omega_1 = 2, \omega_2 = -2$ and $\mu = 0.001$ at $t = 10$.

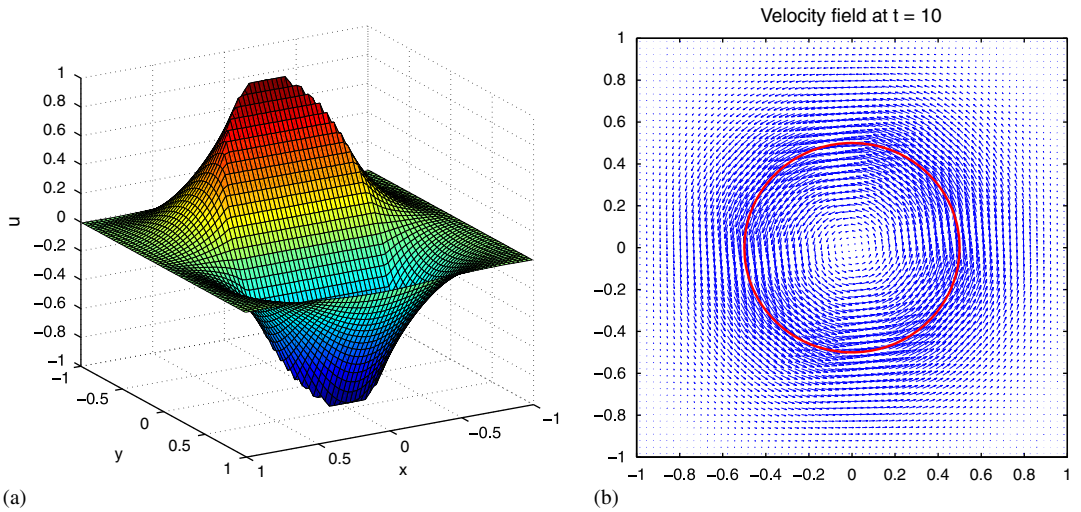

Figure 10. For Example 5.2 on rotational flow: (a) the x-component of velocity field $\mathbf{u}$ and (b) the velocity field $\mathbf{u}$ at $t = 10$.

condition is specified for the pressure increment. For all the simulations, the free stream velocity is used as the initial velocity and the initial pressure is set to zero. Once the velocity field and pressure field have been computed, the drag and lift coefficients and the Strouhal number can be computed from the force at the projection points as found in [1, 7].

DOI: 10.1002/fld

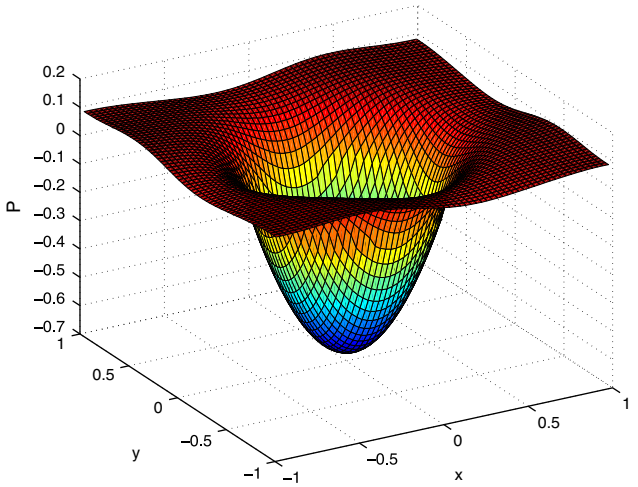

Figure 11. For Example 5.2 on rotational flow. Pressure distribution at $t = 10$.

Table II. Grid refinement analysis for Example 5.2 with $\mu = 0.02$ at $t = 10$.

N	$\| E_{\mathbf{u}} \|_\infty$	Order	$\| E_p \|_\infty$	Order
64	1.6705E−03	—	5.3145E−03	—
128	4.3235E−04	1.95	1.4044E−03	1.92
256	1.0155E−04	2.09	3.8421E−04	1.87

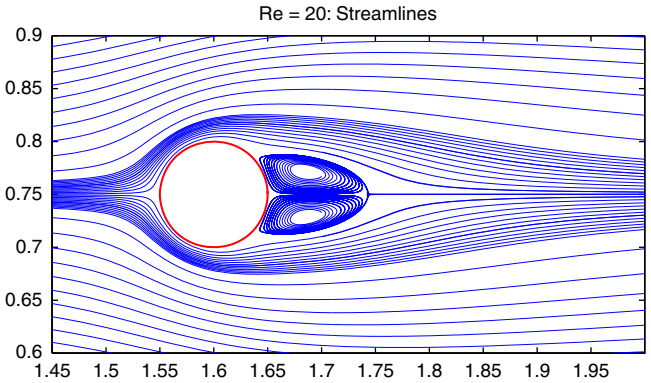

Figure 12. For Example 5.3 on flow past a stationary cylinder. Streamlines for $Re = 20$.

The plots of streamline for $Re = 20$ and $Re = 40$ at steady state are shown in Figures 12 and 14, respectively. For these low Reynolds numbers, as expected, the flow gradually attains a steady state and the wake forms behind the cylinder symmetrically. In Figures 13 and 15, the corresponding plots of the pressure contours for $Re = 20$ and $Re = 40$ are also shown, respectively. These results

DOI: 10.1002/fld

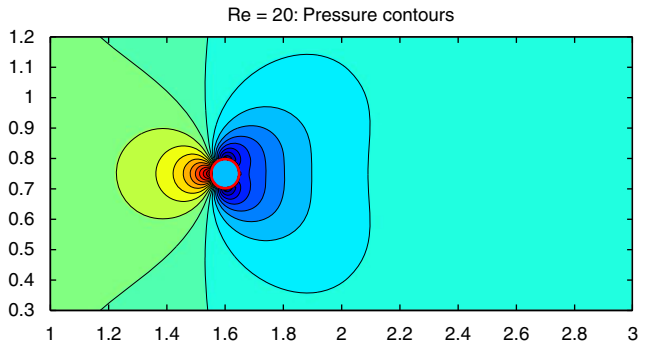

Figure 13. For Example 5.3 on flow past a stationary cylinder. Pressure contours for $Re = 20$.

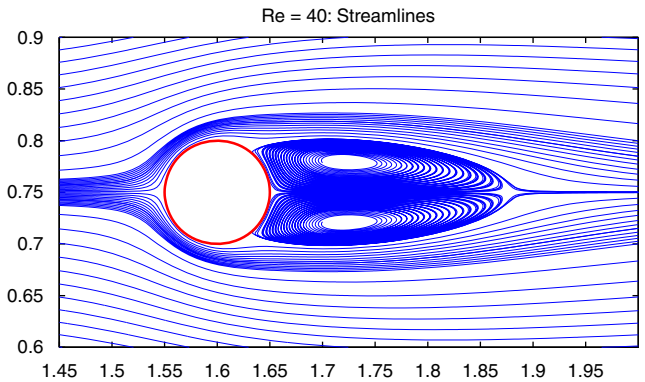

Figure 14. For Example 5.3 on flow past a stationary cylinder. Streamlines for $Re = 40$.

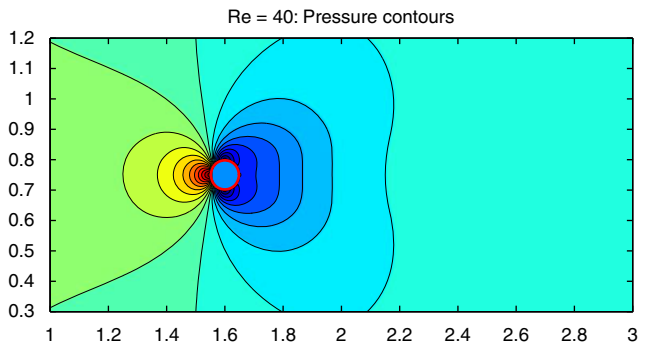

Figure 15. For Example 5.3 on flow past a stationary cylinder. Pressure contours for $Re = 40$.

DOI: 10.1002/fld

Table III. Length of the recirculation zone (L/d) and drag coefficient (C_D) for $Re=20$ and $Re=40$.

	$Re=20$		$Re=40$	
	L/d	C_D	L/d	C_D
Tritton [31]	—	2.22	—	1.48
Coutanceau and Bouard [30]	0.73	—	1.89	—
Fornberg [29]	0.91	2.00	2.24	1.50
Calhoun [18]	0.91	2.19	2.18	1.62
Russell and Wang [20]	0.94	2.13	2.29	1.60
Ye *et al.* [21]	0.92	2.03	2.27	1.52
Le *et al.* [16]	0.93	2.05	2.22	1.56
Present	0.93	2.06	2.24	1.57

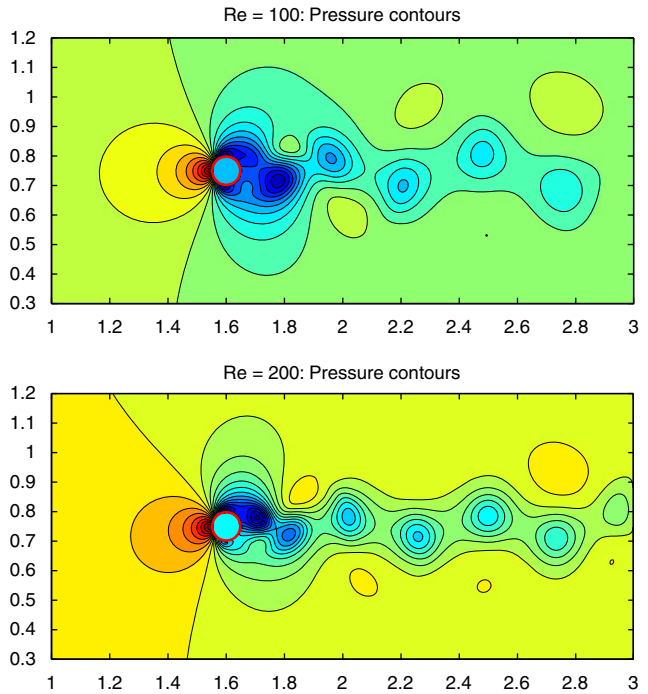

Figure 16. For Example 5.3 on flow past a stationary cylinder. Pressure contours for $Re=100$ and $Re=200$.

are found in a very good agreement with the results of [16]. At the steady state, the drag coefficients and the length of the recirculation zone are computed and are compared with other numerical results [7, 16, 18, 20, 21, 29] as well as experimental results [30, 31] in Table III. It is clear that our drag coefficients are in reasonably good agreement with them. At $Re=100$ and $Re=200$, the flow is unsteady, and Figure 16 shows the pressure field at $Re=100$ and $Re=200$. The instability and vortex shedding can be seen from this figure. In Table IV, the drag coefficients, lift coefficients,

DOI: 10.1002/fld

Table IV. Summary of results for $Re=100$ and $Re=200$.

	$Re=100$			$Re=200$		
	C_L	C_D	S_t	C_L	C_D	S_t
Braza *et al.* [32]	± 0.250	1.36 ± 0.015	—	± 0.75	1.40 ± 0.050	—
Liu *et al.* [33]	± 0.339	1.35 ± 0.012	0.164	± 0.69	1.31 ± 0.049	0.192
Calhoun [18]	± 0.298	1.33 ± 0.014	0.175	± 0.67	1.17 ± 0.058	0.202
Russell and Wang [20]	± 0.300	1.38 ± 0.007	0.169	± 0.50	1.29 ± 0.022	0.195
Le *et al.* [16]	± 0.323	1.37 ± 0.009	0.160	± 0.43	1.34 ± 0.030	0.187
Present	± 0.329	1.37 ± 0.010	0.162	± 0.56	1.36 ± 0.040	0.193

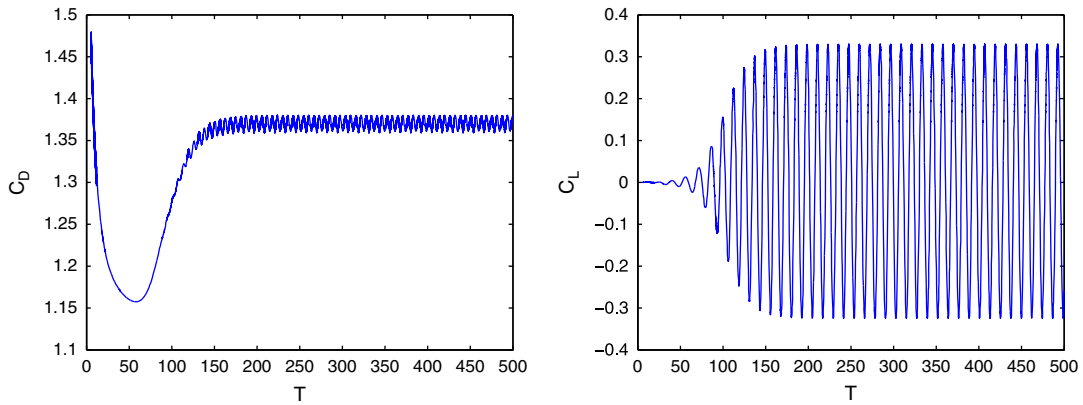

Figure 17. For Example 5.3 on flow past a stationary cylinder. Drag and lift coefficients versus time for $Re=100$.

and Strouhal numbers at $Re=100$ and $Re=200$ are compared with other numerical results. Good agreement is again found from this table. In particular, the plots of time evolution of the drag and lift coefficients at $Re=100$ are presented in Figure 17.

6. CONCLUDING REMARKS

In this paper, a level set-based immersed interface algorithm is presented for solving the incompressible Navier–Stokes equations with the prescribed velocity at the boundary. The method combines the IIM with a level set representation of the interface on a uniform Cartesian grid. The main advantage of the method is that the prescribed boundary condition is exactly satisfied. The grid convergence analysis shows that current algorithm can achieve second-order accurate in both the velocity and pressure. It is a rather straightforward manner to extend the current algorithm to solve the problems with multi-connected domains and moving geometry. Our method is capable of solving incompressible flow problems involving flexible interface in irregular domains by incorporate the current approach with the earlier work [15] which is based on level set method for

DOI: 10.1002/fld

problems with deformable interfaces. The present method can be also easily extended to 3D. A 3D version of the method is under development and will be reported in the future.

REFERENCES

1. Lai M-C, Peskin CS. An immersed boundary method with formal second order accuracy and reduced numerical viscosity. *Journal of Computational Physics* 2000; **160**:707–719.
2. Su S-W, Lai M-C, Lin C-A. An immersed boundary method for simulating the interaction of a fluid with moving boundaries. *Computational Fluids* 2007; **36**:313–324.
3. Peskin CS. The immersed boundary method. *Acta Numerica* 2002; **11**:479–517.
4. Peskin CS. Numerical analysis of blood flow in the heart. *Journal of Computational Physics* 1977; **25**:220–252.
5. Fogelson AL. Continuum models of platelet aggregation: formulation and mechanical properties. *SIAM Journal on Applied Mathematics* 1992; **52**:1089–1110.
6. Wang NT, Fogelson AL. Computational methods for continuum models of platelet aggregation. *Journal of Computational Physics* 1999; **151**:649–675.
7. Lima E, Sliva ALF, Silveira-Neto A, Damasceno JJR. Numerical simulation of two-dimensional flows over a circular cylinder using the immersed boundary method. *Journal of Computational Physics* 2003; **189**:351–370.
8. Goldstein D, Handler R, Sirovich L. Modeling a no-slip flow with an external force field. *Journal of Computational Physics* 1993; **105**:354–366.
9. Mohd-Yusof J. Combined immersed boundary/B-splines methods for simulations of flows in complex geometry. *Annual Research Briefts*, Center for Turbulence Research, 1997; 317–327.
10. Fadlun EA, Verzicco R, Orlandi P. Combined immersed boundary finite-difference methods for three-dimensional complex flows simulations. *Journal of Computational Physics* 2000; **161**:35–60.
11. Uhlmann M. An immersed boundary method with direct forcing for the simulation of particulate flows. *Journal of Computational Physics* 2005; **209**:448–476.
12. LeVeque RJ, Li Z. The immersed interface method for elliptic equations with discontinuous coefficients and singular sources. *SIAM Journal on Numerical Analysis* 1994; **31**:1019–1044.
13. LeVeque RJ, Li Z. Immersed interface methods for Stokes flow with elastic boundaries or surface tension. *SIAM Journal on Scientific Computing* 1997; **18**:709–735.
14. Li Z, Ito K. The immersed interface method—numerical solutions of PDEs involving interfaces and irregular domains. *SIAM Frontiers in Applied Mathematics* 2006; **33**:332.
15. Li Z, Lai MC. The immersed interface method for the Navier–Stokes equations with singular forces. *Journal of Computational Physics* 2001; **171**:822–842.
16. Le DV, Khoo BC, Peraire J. An immersed interface method for viscous incompressible flows involving rigid and flexible boundaries. *Journal of Computational Physics* 2006; **220**:109–138.
17. Lee L, LeVeque RJ. An immersed interface method for incompressible Navier–Stokes equations. *SIAM Journal on Scientific Computing* 2003; **25**:832–856.
18. Calhoun D. A Cartesian grid method for solving the two-dimensional stream function-vorticity equations in irregular regions. *Journal of Computational Physics* 2002; **176**:231–275.
19. Li Z, Wang C. A fast finite difference method for solving Navier–Stokes equations on irregular domains. *Communications in Mathematical Sciences* 2003; **1**:180–196.
20. Russell D, Wang ZJ. A Cartesian grid method for modeling multiple moving objects in 2D incompressible viscous flow. *Journal of Computational Physics* 2003; **191**:177–205.
21. Ye T, Mittal R, Udaykumar HS, Shyy W. An accurate Cartesian grid method for viscous incompressible flows with complex immersed boundary. *Journal of Computational Physics* 1999; **156**:209–240.
22. Udaykumar HS, Mittal R, Rampunggoon P, Khanna A. A sharp interface Cartesian grid method for simulating flows with complex moving boundaries. *Journal of Computational Physics* 2001; **174**:345–380.
23. Adams J, Swarztrauber P, Sweet R. FISHPACK: efficient FORTRAN subprograms for the solution of separable elliptic partial differential equations, 1999. Available from the web at: http://www.scd.ucar.edu/css/software/fishpack/.
24. Harlow FH, Welch JE. Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. *Physics of Fluids* 1965; **8**:2182–2189.
25. Brown DL, Cortez R, Minion ML. Accurate projection methods for the incompressible Navier–Stokes equations. *Journal of Computational Physics* 2001; **168**:464–499.

DOI: 10.1002/fld

26. Wiegmann A, Bube KP. The explicit-jump immersed interface method: finite difference methods for PDEs with piecewise smooth solutions. *SIAM Journal on Numerical Analysis* 2000; **37**:827–862.
27. Hou T, Li Z, Osher S, Zhao H. A hybrid method for moving interface problems with application to the Hele–Shaw flow. *Journal of Computational Physics* 1997; **134**:236–252.
28. Li Z, Zhao H, Gao H. A numerical study of electro-migration voiding by evolving level set functions on a fixed cartesian grid. *Journal of Computational Physics* 1999; **152**:281–304.
29. Fornberg B. A numerical study of steady viscous flow past a circular cylinder. *Journal of Fluid Mechanics* 1980; **98**:819–855.
30. Coutanceau M, Bouard R. Experimental determination of the main features of the viscous flow in the wake of a circular cylinder in uniform translation, part 1. Steady flow. *Journal of Fluid Mechanics* 1977; **79**:231–256.
31. Tritton DJ. Experiments on the flow past a circular cylinder at low Reynolds numbers. *Journal of Fluid Mechanics* 1959; **6**:547–567.
32. Braza M, Chassaing P, Ha Minh H. Numerical study and physical analysis of the pressure and velocity fields in the near wake of a circular cylinder. *Journal of Fluid Mechanics* 1986; **165**:79–130.
33. Liu C, Sheng X, Sung CH. Preconditioned multigrid methods for unsteady incompressible flows. *Journal of Computational Physics* 1998; **139**:35–57.

DOI: 10.1002/fld