COMMUNICATIONS IN COMPUTATIONAL PHYSICS
Vol. **6**, No. 5, pp. 997-1021

Commun. Comput. Phys.
November 2009

An Indirect-Forcing Immersed Boundary Method for Incompressible Viscous Flows with Interfaces on Irregular Domains

Zhijun Tan[1,*], K. M. Lim[1,2], B. C. Khoo[1,2] and Desheng Wang[3]

[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.*
[3] *Division of Mathematical Sciences, School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore 637371, Singapore.*

Received 14 August 2008; Accepted (in revised version) 20 March 2009

Available online 8 May 2009

Abstract. An indirect-forcing immersed boundary method for solving the incompressible Navier-Stokes equations involving the interfaces and irregular domains is developed. The rigid boundaries and interfaces are represented by a number of Lagrangian control points. Stationary rigid boundaries are embedded in the Cartesian grid and singular forces at the rigid boundaries are applied to impose the prescribed velocity conditions. The singular forces at the interfaces and the rigid boundaries are then distributed to the nearby Cartesian grid points using the immersed boundary method. In the present work, the singular forces at the rigid boundaries are computed implicitly by solving a small system of equations at each time step to ensure that the prescribed velocity condition at the rigid boundary is satisfied exactly. For deformable interfaces, the forces that the interface exerts on the fluid are computed from the configuration of the elastic interface and are applied to the fluid. The Navier-Stokes equations are discretized using finite difference method on a staggered uniform Cartesian grid by a second order accurate projection method. The ability of the method to simulate viscous flows with interfaces on irregular domains is demonstrated by applying to the rotational flow problem, the relaxation of an elastic membrane and flow in a constriction with an immersed elastic membrane.

AMS subject classifications: 65N06, 76D05, 65M12

Key words: Incompressible Navier-Stokes equation, fast Poisson solvers, immersed boundary method, projection method, Cartesian grid, irregular domain, finite difference methods.

*Corresponding author. *Email addresses:* `smatz@nus.edu.sg` (Z.-J. Tan), `mpelimkm@nus.edu.sg` (K. M. Lim), `mpekbc@nus.edu.sg` (B. C. Khoo), `desheng@ntu.edu.sg` (D. S. Wang)

1 Introduction

Flow problems involving the deformable interface and complex geometries still pose a difficult challenge in computational fluid dynamics. One of the challenges in these problems is that the fluid motion, the motion of the deformable interface and the interaction with the immersed rigid boundaries must be computed simultaneously. This is necessary in order to account for the complex interaction between the fluid, the interfaces and the immersed boundaries. Fig. 1 shows an illustration of such problems involving the rigid boundary and fixed/deformable interface embedded in a uniform Cartesian grid.

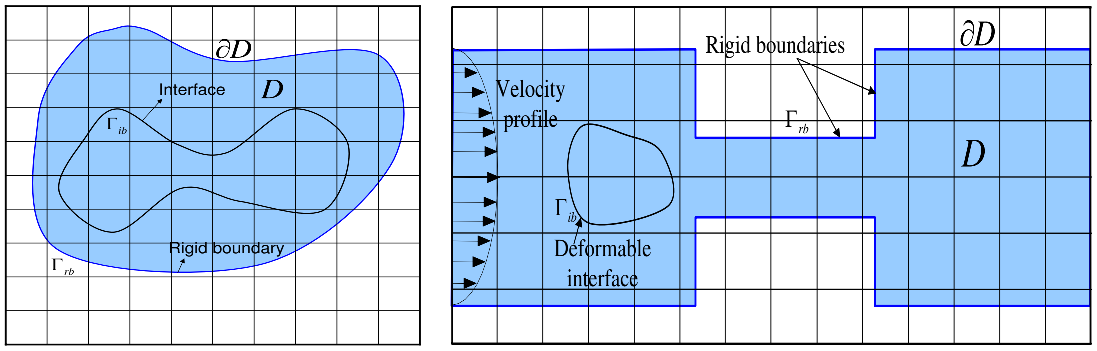


Figure 1: Two typical domains with the rigid boundary and fixed/deformable interface immersed in a uniform Cartesian grid.

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 and popular approach for solving complex viscous flows involving the interfaces on a complex geometry is the Cartesian grid method which 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 [25]. This method was originally developed to study the fluid dynamics of blood flow in the human heart [24]. The method was developed further and has been applied to many biological problems involving flexible boundaries [7,34]. The immersed boundary method has also been applied to handle problems with immersed boundaries [11,20]. In order to deal with rigid immersed boundaries, Lai and Peskin [11] proposed to evaluate the force density using an expression of the form,

$$\mathbf{f}(s,t)=K_r(\mathbf{X}^e(s)-\mathbf{X}(s,t)), \tag{1.1}$$

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 Eq. (1.1) is a particular case of the feedback forcing formulation proposed by Goldstein et al. [9] with $\beta = 0$. In [9], the force is expressed as

$$\mathbf{f}(s,t) = \alpha \int_0^t \mathbf{U}(s,t')dt' + \beta \mathbf{U}(s,t), \tag{1.2}$$

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 timestep, Mohd-Yusof [22] and Fadlun et al. [6] 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 Silva et al. [20] 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 [33] 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. In the immersed boundary method, 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.

A second Cartesian grid method is the immersed interface method (IIM). The IIM was originally proposed by LeVeque and Li [16] for solving elliptic equations, and later extended to Stokes flow with elastic boundaries or surface tension [15]. We refer the interested readers to the newly published book by Li and Ito [17]. The method was developed further for the Navier-Stokes equations in [13, 14, 18] for problems with flexible boundaries. Recently, the immersed interface method [35] has been employed to solve for viscous flows with static rigid immersed boundaries [4, 13, 19, 27]. In [4, 19], the no-slip boundary conditions are imposed directly by determining the correct jump conditions for streamfunction and vorticity. In [27], a Cartesian grid method for modelling multiple moving objects in incompressible viscous flow is considered. Le et al. [13] has presented an immerse interface method for viscous flows involving rigid and flexible boundaries. In [13], 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. [35] and Udaykumar et al. [32] 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 contrast to IIM, the disadvantage of the immersed boundary method is that it is only first-order accurate for problems with non-smooth but continuous solutions since the immersed boundary method smears out sharp interface to a thickness of order of the mesh width, but the advantage is simple and much easier implementation on a Cartesian grid, especially if to think about the further extension of proposed method in this paper to the three dimensional problems. Considering these advantages, in this paper, we will use this approach. To our knowledge, the immersed boundary is developed to handle mostly the fluid problem involving only interfaces, it has also been used to simulated the flow with only rigid boundaries. However, there is comparatively less work on the application of the immersed boundary method to the incompressible viscous flow problems involving both the interfaces and rigid boundaries.

In this work, we shall present an immersed boundary method with indirect forcing for solving the incompressible Navier-Stokes equations in the presence of both the interfaces and irregular boundaries simultaneously, which is based on the approach of Le et al. [12, 13] in terms of the evaluation of the forcing term. The method combines the immersed boundary method with a front tracking representation of the interface and rigid boundary on a uniform Cartesian grid. In the proposed method, the singular force at the rigid boundary is determined to enforce the prescribed velocity condition at the rigid boundary of the irregular domain. At each time step, the singular force at the rigid boundary is computed implicitly by solving a small, dense linear system of equations using SVD iterative method. Once the force at the rigid boundary is computed, the Navier-Stokes equations are discretized on a staggered Cartesian grid by a second order accurate projection method for the pressure and velocity. Fast solvers from the FISHPACK software library [1] are used to solve the resulting discrete systems of equations.

This paper is organized as follows. In Section 2, we present the governing equations including the immersed boundary formulation. In Section 3, we briefly describe the immersed boundary method and the projection method. We present the determination of the forcing term at the rigid boundary. The numerical algorithm and numerical implementation are presented in Section 4. In Section 5, some numerical examples are presented. Finally some concluding remarks will be made in Section 6.

2 Governing equations

This paper considers the viscous incompressible flows involving the interfaces for two dimensional problems on irregular domain. In a two dimensional bounded domain D containing a immersed interface Γ_{ib} with the irregular domain ∂D, the incompressible Navier-Stokes equations formulated in primitive variables is considered and written as

$$\rho(\mathbf{u}_t+(\mathbf{u}\cdot\nabla)\mathbf{u})+\nabla p=\mu\Delta\mathbf{u}+\mathbf{F}_{ib}(\mathbf{x},t)+\mathbf{g}(\mathbf{x},t), \quad \mathbf{x}\in D, \tag{2.1}$$

$$\nabla\cdot\mathbf{u}=0, \quad \mathbf{x}\in D, \tag{2.2}$$

with initial and boundary conditions

$$\mathbf{u}(\mathbf{x},0)=\mathbf{u}_0, \qquad \mathbf{u}|_{\partial D}=\mathbf{u}_p, \tag{2.3}$$

where $\mathbf{u}=(u,v)^T$ is the fluid velocity, p is the fluid pressure, ρ is the fluid density, μ is the viscosity of the fluid, $\mathbf{x}=(x,y)$ is the Cartesian coordinate variable, $\mathbf{g}(\mathbf{x},t)=(g_1,g_2)^T$ is an external force, and the effect of the interface immersed in the fluid results in a singular force $\mathbf{F}_{ib}$ which has the form

$$\mathbf{F}_{ib}(\mathbf{x},t)=\int_{\Gamma_{ib}} \mathbf{f}^{ib}(s,t)\delta(\mathbf{x}-\mathbf{X}^{ib}(s,t))ds. \tag{2.4}$$

Here $\mathbf{X}^{ib}(s,t)$ is the arc-length parametrization of the interface Γ_{ib}, s is the arc-length function (in some reference configuration), $\mathbf{f}^{ib}=(f_1^{ib},f_2^{ib})^T$ is the corresponding force density, and $\delta(\cdot)$ is the Dirac delta function defined in the distribution sense. The motion of the interface satisfies

$$\frac{\partial \mathbf{X}^{ib}(s,t)}{\partial t}=\mathbf{u}(\mathbf{X}^{ib}(s,t),t)=\int_{\Omega}\mathbf{u}(\mathbf{x},t)\delta(\mathbf{x}-\mathbf{X}^{ib}(s,t))d\mathbf{x}. \tag{2.5}$$

In this model, for deformable interface problem, we consider an immersed moving interface problems which involves an elastic membrane, where the force strength $\mathbf{f}^{ib}$ exerted by elastic membrane on the fluid of the form is given by

$$\mathbf{f}^{ib}(s,t)=\frac{\partial}{\partial s}\Big(T(s,t)\boldsymbol{\tau}(s,t)\Big), \tag{2.6}$$

with the tension $T(s,t)$ given by

$$T(s,t)=T_0\Big(\Big|\frac{\partial \mathbf{X}^{ib}(s,t)}{\partial s}\Big|-1\Big). \tag{2.7}$$

Here, the tension coefficient T_0 is the stiffness constant which describes the elastic property of the membrane. The vector tangential to Γ is given by $\boldsymbol{\tau}(s,t)$, where

$$\boldsymbol{\tau}(s,t)=\frac{\partial \mathbf{X}^{ib}}{\partial s}\Big/\Big|\frac{\partial \mathbf{X}^{ib}}{\partial s}\Big|.$$

Thus, the force density can be computed directly from the location $\mathbf{X}^{ib}$ of the membrane Γ_{ib}.

The irregular domain D can be extended to a larger rectangular domain Ω by an embedding technique, see Fig. 1. In order to impose the prescribed velocity condition at the irregular boundary, i.e., $\mathbf{u}|_{\partial D}=\mathbf{u}_p$, the boundary ∂D can be treated as an immersed boundary Γ_{rb} that exerts force to the fluid [11,30]. These singular forces at the immersed boundary Γ_{rb} can be introduced as the augmented variables so that the irregular boundary condition is satisfied, i.e., $\mathbf{u}|_{\Gamma_{rb}}=\mathbf{u}_p$. Therefore, finding the solutions of Eqs. (2.1)-(2.3) is equivalent to solving the following equations:

$$\rho(\mathbf{u}_t+(\mathbf{u}\cdot\nabla)\mathbf{u})+\nabla p=\mu\Delta\mathbf{u}+\mathbf{F}(\mathbf{x},t)+\mathbf{g}(\mathbf{x},t), \quad \mathbf{x}\in\Omega, \tag{2.8}$$

$$\nabla\cdot\mathbf{u}=0, \quad \mathbf{x}\in\Omega, \tag{2.9}$$

$$\mathbf{u}|_{\Gamma_{rb}}=\mathbf{u}_p, \tag{2.10}$$

with the boundary condition $\mathbf{u}|_{\partial\Omega}=\mathbf{u}_b$. Here Eq. (2.10) is the corresponding augmented equation and the total singular force $\mathbf{F}$ has the form of

$$\mathbf{F}(\mathbf{x},t)=\mathbf{F}_{ib}+\int_{\Gamma_{rb}}\mathbf{f}^{rb}(s,t)\delta(\mathbf{x}-\mathbf{X}^{rb}(s,t))ds. \tag{2.11}$$

Here $\mathbf{X}^{rb}(s,t)$ is the arc-length parametrization of the rigid boundary Γ_{rb}, and $\mathbf{f}^{rb}=(f_1^{rb},f_2^{rb})^T$ is the corresponding force density.

Throughout this paper, we assume that the fluid density and fluid viscosity μ are constants over the whole domain. We refer the readers to Fig. 1 for an illustration of the problem.

3 Numerical algorithm

Our numerical algorithm is based on the pressure-increment projection algorithm [3] for the discretization of the Navier-Stokes equations using the immersed boundary method. The spatial discretization is carried out on a standard marker-and-cell (MAC) staggered grid similar to that found in Kim and Moin [10]. We use a uniform MAC gird with mesh width $h=\Delta x=\Delta y$ in the computation. With the MAC mesh, the pressure field is defined at the cell center (i,j), where $i\in\{1,2,\cdots,N_x\}$ and $j\in\{1,2,\cdots,N_y\}$. The velocity fields u and v are defined at the vertical edges and horizontal edges of a cell, respectively. We use the notation

$$u_{i,j}=u(x_i,y_j+\frac{1}{2}h), \qquad i=1,\cdots,N_x+1, \quad j=1,\cdots,N_y, \tag{3.1}$$

$$v_{i,j}=u(x_i+\frac{1}{2}h,y_j), \qquad i=1,\cdots,N_x, \quad j=1,\cdots,N_y+1, \tag{3.2}$$

$$p_{i,j}=p(x_i+\frac{1}{2}h,y_j+\frac{1}{2}h), \quad i=1,\cdots,N_x, \quad j=1,\cdots,N_y, \tag{3.3}$$

where $x_i=ih$, $x_j=jh$. The pressure and velocity components u and v are arranged as seen in Fig. 2.

3.1 Description of immersed boundary method

In our numerical scheme, we use two sets of control points $\mathbf{X}_k^{ib}=(X_k^{ib},Y_k^{ib})$, $k=1,\cdots,N_{ib}$ and $\mathbf{X}_k^{rb}=(X_k^{rb},Y_k^{rb})$, $k=1,\cdots,N_{rb}$ to represent the immersed interface and the rigid boundary, respectively. The force densities are computed at these control points and are spread to the Cartesian grid points by a discrete representation of the delta function,

$$\mathbf{F}(\mathbf{x}(i,j),t)=\sum_{k=1}^{N_{ib}}\mathbf{f}_k^{ib}(t)D_h(\mathbf{x}(i,j)-\mathbf{X}_k^{ib}(t))\Delta s+\sum_{k=1}^{N_{rb}}\mathbf{f}_k^{rb}(t)D_h(\mathbf{x}(i,j)-\mathbf{X}_k^{rb}(t))\Delta s, \tag{3.4}$$

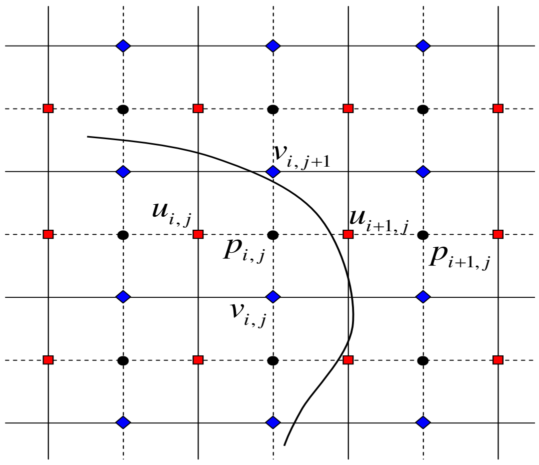


Figure 2: A diagram of the interface cutting through a staggered grid with a uniform mesh width h, where the velocity component u is at the left-right face of the cell and v is at the top-bottom face, and the pressure p is at the cell center.

where $\mathbf{f}_k^{\mathcal{Q}}(t)$ is the force density at the control point $\mathbf{X}_k^{\mathcal{Q}}$, $\mathcal{Q}=ib$, rb, $\mathbf{x}(i,j)$ and $\mathbf{F}(\mathbf{x}(i,j))$ are the coordinate of grid point (i,j) and the force at that point, respectively. $D_h(x)$ is a two-dimensional discrete delta function,

$$D_h(\mathbf{x})=\frac{1}{h^2}\phi\left(\frac{x}{h}\right)\phi\left(\frac{y}{h}\right), \tag{3.5}$$

where ϕ a continuous function which was derived in [23,25] as

$$\phi(r)=\begin{cases}\frac{1}{8}\left(3-2|r|+\sqrt{1+4|r|-4|r|^2}\right), & |r|\le 1,\\ \frac{1}{8}\left(5-2|r|-\sqrt{-7+12|r|-4r^2}\right), & 1\le|r|\le 2,\\ 0, & \text{otherwise.}\end{cases} \tag{3.6}$$

Once the force densities are computed at the control points and spread to the grid, the Navier-Stokes equations with the forcing terms are then solved for the pressure and velocity field at Cartesian grid points using projection method [4]. The velocity field is then interpolated to find the velocity at the control points as

$$\mathbf{U}(\mathbf{X}_k^{\mathcal{Q}},t)=\sum_{i,j}\mathbf{u}(\mathbf{x}(i,j),t)D_h(\mathbf{x}(i,j)-\mathbf{X}_k^{\mathcal{Q}}(t))h^2, \qquad \mathcal{Q}=ib,rb. \tag{3.7}$$

3.2 Projection method

Given the velocity $\mathbf{u}^n$, and the pressure $p^{n-1/2}$, we compute the velocity $\mathbf{u}^{n+1}$ and pressure $p^{n+1/2}$ at the next time step as follows:

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

$$\begin{aligned}&\frac{\mathbf{u}^*-\mathbf{u}^n}{\Delta t}+(\mathbf{u}\cdot\nabla\mathbf{u})^{n+\frac{1}{2}}=-\frac{1}{\rho}\nabla p^{n+\frac{1}{2}}+\frac{\mu}{2\rho}(\Delta_h\mathbf{u}^*+\Delta_h\mathbf{u}^n)+\frac{1}{\rho}\mathbf{F}^{n+\frac{1}{2}}+\frac{1}{\rho}\mathbf{g}^{n+\frac{1}{2}},\\ &\mathbf{u}^*|_{\partial\Omega}=\mathbf{u}_b^{n+1},\end{aligned} \tag{3.8}$$

where the advection term is extrapolated using the formula,

$$(\mathbf{u}\cdot\nabla\mathbf{u})^{n+\frac{1}{2}}=\frac{3}{2}(\mathbf{u}\cdot\nabla_h\mathbf{u})^n-\frac{1}{2}(\mathbf{u}\cdot\nabla_h\mathbf{u})^{n-1}, \tag{3.9}$$

and the pressure gradient is approximated simply as

$$\nabla p^{n+\frac{1}{2}}=G^{\mathrm{MAC}}p^{n-\frac{1}{2}}. \tag{3.10}$$

Above step can be rewritten in the following Helmholtz equations form:

$$\lambda_0\mathbf{u}^*+\Delta_h\mathbf{u}^*=\mathbf{RHS}, \tag{3.11}$$

where $\mathbf{RHS}$ is the right hand terms and $\lambda_0=-\frac{2\rho}{\mu\Delta t}$. In general, This intermediate velocity field does not satisfy the divergence-free condition (2.9).

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

$$\Delta_h\phi^{n+1}=\frac{\rho}{\Delta t}D^{\mathrm{MAC}}\mathbf{u}^*, \tag{3.12}$$

with boundary condition

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

Step 3: Once ϕ^{n+1} is obtained by solving Eq. (3.12), both the pressure and velocity field $(p^{n+\frac{1}{2}},\mathbf{u}^{n+1})$ are updated as

$$\mathbf{u}^{n+1}=\mathbf{u}^*-\frac{\Delta t}{\rho}G^{\mathrm{MAC}}\phi^{n+1}, \tag{3.13}$$

$$p^{n+\frac{1}{2}}=p^{n-\frac{1}{2}}+\phi^{n+1}-\frac{\mu}{2\rho}D^{\mathrm{MAC}}\mathbf{u}^*. \tag{3.14}$$

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

$$\begin{aligned}&\nabla_h\mathbf{u}_{i,j}=\left(\frac{\mathbf{u}_{i+1,j}-\mathbf{u}_{i-1,j}}{2h},\frac{\mathbf{u}_{i,j+1}-\mathbf{u}_{i,j-1}}{2h}\right),\\ &\Delta_h\mathbf{u}^*_{i,j}=\frac{1}{h^2}\left(\mathbf{u}^*_{i+1,j}+\mathbf{u}^*_{i-1,j}+\mathbf{u}^*_{i,j+1}+\mathbf{u}^*_{i,j-1}-4\mathbf{u}^*_{ij}\right),\\ &G^{\mathrm{MAC}}_{ij}=\left(\frac{p_{i+1,j}-p_{i,j}}{h},\frac{p_{i,j+1}-p_{i,j}}{h}\right),\\ &(D^{\mathrm{MAC}}\mathbf{u})_{i,j}=\frac{u_{i+1,j}-u_{i,j}}{h}+\frac{v_{i,j+1}-v_{i,j}}{h}.\end{aligned} \tag{3.15}$$

In our projection method, we need to solve, at each time step, two Helmholtz equations for $\mathbf{u}^*$ in (3.8) or (3.11) and one Poisson-like equation for ϕ^{n+1} in (3.12). In the present work, we take advantage of the fast solvers from FISHPACK [1] to solve these equations.

3.3 Determination of indirect forcing $\mathbf{f}^{rb}$

Assuming that the force $\mathbf{f}^{rb}$ at the control points of the rigid boundary is known, the force at the Cartesian grid points $\mathbf{F}$ can be computed as in Eq. (3.4), which can be written in the matrix-vector form as

$$\mathbf{F}=\mathcal{D}^{ib}\mathbf{f}^{ib}+\mathcal{D}^{rb}\mathbf{f}^{rb}. \tag{3.16}$$

Here, $\mathcal{D}^{ib}$ and $\mathcal{D}^{rb}$ are the distribution operators in the sense that singular forces at the control points of the interface and rigid boundary are distributed to the nearby Cartesian grid points. Once the force at the Cartesian grid points are computed, the velocity field $\mathbf{u}^{n+1}$ at all the grid points is then computed via the projection method as discussed in Section 3.2. The velocity field $\mathbf{u}^{n+1}$ is used to evaluate the velocity at the control points as shown in Eq. (3.7).

In summary, the equations that need to be solved in order to calculate $\mathbf{u}^{n+1}$ and $\mathbf{U}_k^{rb}$ at the rigid boundary, can be written symbolically as,

$$\text{Eq. (3.11)} \longrightarrow \quad \mathbf{H}\mathbf{u}^*=\mathbf{C}+\mathbf{B}^{ib}\mathbf{f}^{ib}+\mathbf{B}^{rb}\mathbf{f}^{rb}, \tag{3.17}$$

$$\text{Eq. (3.12)} \longrightarrow \quad \mathbf{L}\boldsymbol{\phi}^{n+1}=\mathbf{D}\mathbf{u}^*, \tag{3.18}$$

$$\text{Eq. (3.13)} \longrightarrow \quad \mathbf{u}^{n+1}=\mathbf{u}^*-\mathbf{G}\boldsymbol{\phi}^{n+1}, \tag{3.19}$$

$$\text{Eq. (3.16)} \longrightarrow \quad \mathbf{U}_k^{rb}=\mathbf{M}\mathbf{u}^{n+1}, \tag{3.20}$$

where

$$\mathbf{C}=-\frac{2\rho}{\mu\Delta t}\mathbf{u}^n-\Delta_h\mathbf{u}^n+\frac{\rho}{\mu}\Big(3(\mathbf{u}\cdot\nabla_h\mathbf{u})^n-(\mathbf{u}\cdot\nabla_h\mathbf{u})^{n-1}\Big)+\frac{2}{\mu}\Big(G^{\mathrm{MAC}}p^{n-\frac{1}{2}}-\mathbf{g}^{n+\frac{1}{2}}\Big),$$

which can be computed based on the solutions at the previous time step. The operators $\mathbf{H}$, $\mathbf{B}^{ib}$, $\mathbf{B}^{rb}$, $\mathbf{L}$, $\mathbf{D}$ and $\mathbf{G}$ correspond to $\Delta_h-\frac{2\rho}{\mu\Delta t}$, $-\frac{2}{\mu}\mathcal{D}^{ib}$, $-\frac{2}{\mu}\mathcal{D}^{rb}$, Δ_h, $\frac{\rho}{\Delta t}D^{\mathrm{MAC}}$ and $\frac{\Delta t}{\rho}G^{\mathrm{MAC}}$, respectively. $\mathbf{M}$ is the interpolation operator in the sense of Eq. (3.7). Eliminating $\mathbf{u}^*$, $\boldsymbol{\phi}^{n+1}$ and $\mathbf{u}^{n+1}$ from Eqs. (3.17)-(3.20), we can compute the velocity $\mathbf{U}_k^{rb}$ at the control points of the rigid boundary as follows,

$$\begin{aligned}\mathbf{U}_k^{rb}=&\,\mathbf{M}\Big(\mathbf{I}-\mathbf{G}\mathbf{L}^{-1}\mathbf{D}\Big)\mathbf{H}^{-1}\mathbf{C}+\mathbf{M}\Big(\mathbf{I}-\mathbf{G}\mathbf{L}^{-1}\mathbf{D}\Big)\mathbf{H}^{-1}\mathbf{B}^{ib}\mathbf{f}^{ib}\\&+\mathbf{M}\Big(\mathbf{I}-\mathbf{G}\mathbf{L}^{-1}\mathbf{D}\Big)\mathbf{H}^{-1}\mathbf{B}^{rb}\mathbf{f}^{rb},\end{aligned} \tag{3.21}$$

where $\mathbf{I}$ is an identity matrix. For simplicity, we can write Eq. (3.21) as,

$$\mathbf{U}_k^{rb}=\mathbf{U}_k^{rb,0}+\mathbf{A}\mathbf{f}^{rb}, \tag{3.22}$$

where $\mathbf{U}_k^{rb,0}$ corresponds to the velocity at the control points of the rigid boundary obtained by solving Eq. (3.11) with $\mathbf{f}^{rb}=0$, and solving Eqs. (3.12), (3.13) and the interpolation (3.20), given $\mathbf{u}^n$ and $p^{n-1/2}$. $\mathbf{A}$ is a $2N_{rb}\times 2N_{rb}$ matrix, where N_{rb} is the number of control points of the rigid boundary. The vector $\mathbf{Af}^{rb}$ is the velocity at the control points of the rigid boundary obtained by solving Eq. (3.11) with $C=0$ and $\mathbf{f}^{ib}=0$, and solving Eqs. (3.12), (3.13) and the interpolation (3.20). From Eq. (3.22), with the prescribed velocity $\mathbf{U}_p^{rb}$ at the rigid boundary, the singular force $\mathbf{f}^{rb}$ at the rigid boundary can be determined by solving

$$\mathbf{Af}^{rb}=\mathbf{U}_p^{rb}-\mathbf{U}_k^{rb,0}, \tag{3.23}$$

Eq. (3.23) can be solved using the generalized minimum residual (GMRES) method [28]. Each GMRES iteration involves one vector-matrix product $A\mathbf{f}^{rb}$ with a known $\mathbf{f}^{rb}$. In the present work, the rigid boundary is fixed. Because the matrix $\mathbf{A}$ depends on the location of the boundary and the time step Δt, we will have the same matrix $\mathbf{A}$ at every time step if we use the same Δt throughout. Therefore, the matrix $\mathbf{A}$ needs to be formed and factorized only once. In order to compute the coefficients of $\mathbf{A}$, we solve Eqs. (3.17)-(3.20) for $2N_{rb}$ times, i.e. once for each column. Each time, the force $\mathbf{f}^{rb}$ is set to zero except for the entry corresponding to the column we want to calculate, which is set to one. Once the matrix $\mathbf{A}$ has been calculated, only the terms on the right hand side, $\mathbf{U}_p^{rb}-\mathbf{U}_k^{rb,0}$, needs to be computed at each time step. The resulting small system of Eq. (3.23) is then solved at each time step for $\mathbf{f}^{rb}$ via back substitution. Finally, we solve Eqs. (3.8)-(3.14) to obtain $\mathbf{u}^{n+1}$ and $p^{n+1/2}$. In actual computation, we use the singular value decomposition (SVD) method to solve the system of Eq. (3.23).

4 Numerical implementation

In this section, we describe a basic implementation of our algorithm for involving the deformable interface and rigid boundary. We factorize the coefficient matrix using singular value decomposition as,

$$A=\mathbf{U\Sigma V}^T, \tag{4.1}$$

where $\mathbf{U}=[u_1,\cdots,u_{2N_{rb}}]$ and $\mathbf{V}=[v_1,\cdots,v_{2N_{rb}}]$ are the orthogonal matrices, and $\mathbf{\Sigma}=\text{diag}(\sigma_1,\cdots,\sigma_{2N_{rb}})$ is a diagonal matrix whose elements are the singular values of the original matrix such that

$$\sigma_1\geq\sigma_2\geq\cdots\geq\sigma_{2N_{rb}}\geq 0.$$

We store $\mathbf{U}$, $\mathbf{V}$ and $\mathbf{\Sigma}$ for solving the force $\mathbf{f}^{rb}$ at every time step. At each time step, given the location of control points at the interface $\mathbf{X}_k^{ib,n}$, the velocity field $\mathbf{u}^n$ and pressure field $p^{n-1/2}$, our algorithm for finding $\mathbf{X}_k^{ib,n+1}$, $\mathbf{u}^{n+1}$, $p^{n-1/2}$ and the singular force $\mathbf{f}^{rb}$ to enforce the prescribed velocity $\mathbf{U}_p^{rb}$ at the immersed boundary can be summarized as follows:

Algorithm 4.1: The implementation with the deformable interface and rigid boundary

Step 1: Compute the singular force $\mathbf{f}^{ib}$ at the deformable interface using expression (2.6) via using cubic splines.

Step 2: Compute the right hand side of (3.23) by calculating $\mathbf{U}_p^{rb}-\mathbf{U}_k^{rb,0}$.

- Set $\mathbf{f}^{rb}=0$, and solve (3.11)-(3.13) for the velocity at all the grid points.
- Interpolate the velocity at the control points of the rigid boundary $\mathbf{U}_k^{rb,0}$ by solving Eq. (3.20).
- Compute the right hand side vector $\mathbf{b}=\mathbf{U}_p^{rb}-\mathbf{U}_k^{rb,0}$.

Step 3: Compute the singular force $\mathbf{f}^{rb}$ by solving (3.23) using the SVD method. The singular force $\mathbf{f}^{rb}$ can be written in terms of the SVD as

$$\mathbf{w}=\sum_{i=1}^{N_{rb}}\frac{u_i^T\mathbf{b}}{\sigma_i}v_i. \tag{4.2}$$

Step 4: Spreading the singular force $\mathbf{f}^{ib}$ and $\mathbf{f}^{rb}$ to the nearby Cartesian grid points. Solve Eqs. (3.11), (3.12), (3.13) and (3.14) to obtain $\mathbf{u}^{n+1}$ and $p^{n+1/2}$ using the projection method.

Step 5: Interpolate the new velocity on the grids onto the control points of the interface via using Eq. (3.7), $\mathbf{U}_k^{ib,n+1}$, and move the control points of the interface to the new positions, $\mathbf{X}_k^{ib,n+1}$, via using

$$\mathbf{X}_k^{ib,n+1}=\mathbf{X}_k^{ib,n}+\Delta t\mathbf{U}_k^{ib,n+1}. \tag{4.3}$$

5 Numerical examples

In this section, several numerical examples are carried out to demonstrate the capabilities of our proposed algorithm in this work. Throughout this section, we take $\rho\equiv 1$ in all simulations.

Example 5.1. Flow past a circular cylinder

In this benchmark example with no interface involved, an unsteady flow past a stationary circular cylinder of diameter $d=0.1$ immersed in a rectangular domain $\Omega=[-1,2]\times[-0.75,0.75]$ is first simulated to test the validation of the present algorithm on irregular domain. The center of the cylinder is located at $(0,0)$. 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 condition is specified for the pressure increment. The free stream velocity is used as the initial velocity and the initial pressure is set to zero. Once

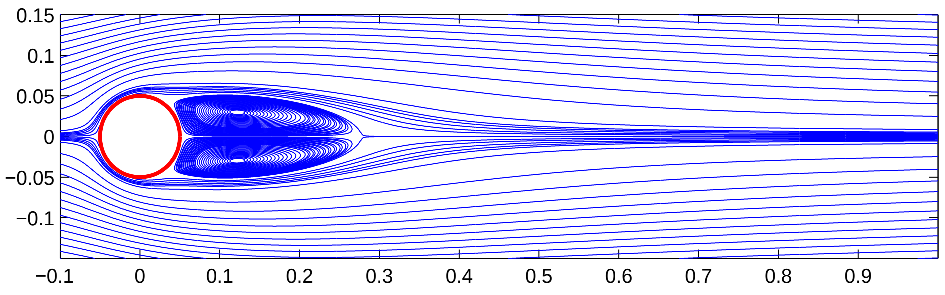


Figure 3: For Example 5.1 on flow past a stationary cylinder. Streamlines for $Re=40$.

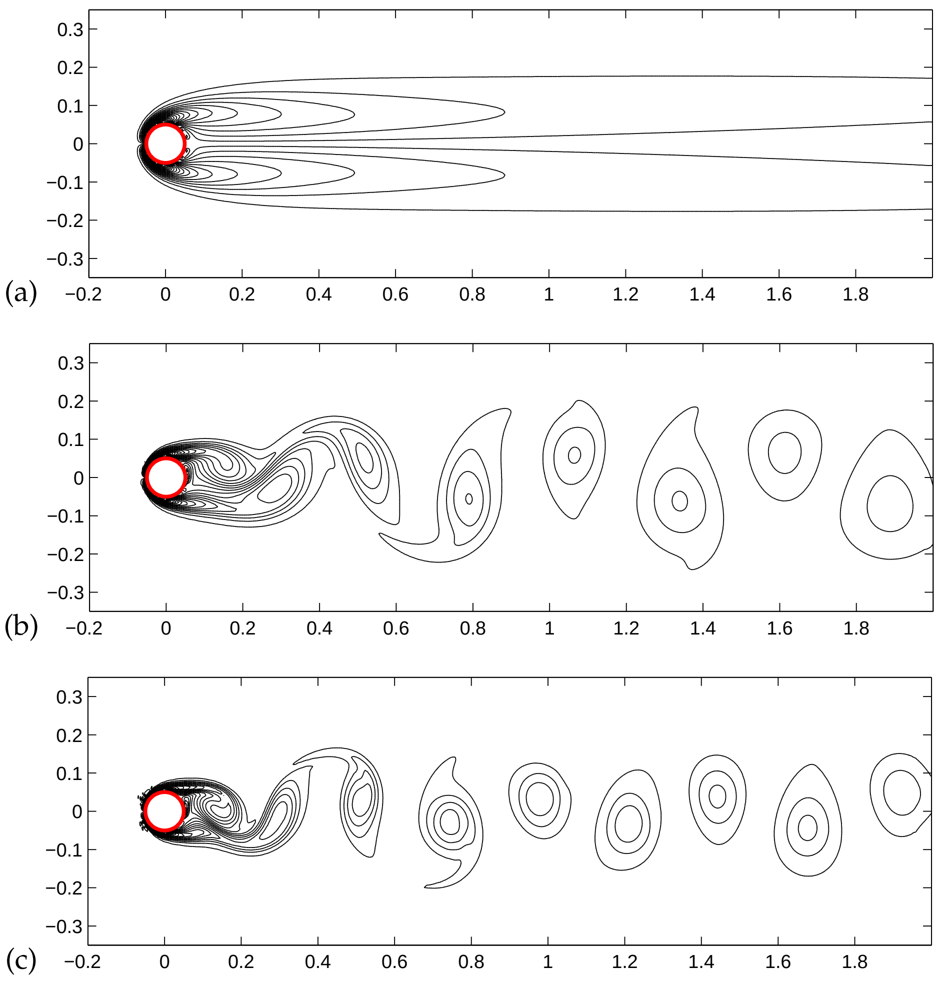


Figure 4: For Example 5.1 on flow past a stationary cylinder. Vorticity contours for (a) $Re=40$; (b) $Re=100$; and (c) $Re=200$.

Table 1: 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 [29]	–	2.22	–	1.48
Coutanceau and Bouard [5]	0.73	–	1.89	–
Fornberg et al. [8]	0.91	2.00	2.24	1.50
Calhoun [4]	0.91	2.19	2.18	1.62
Russell and Wang [27]	0.94	2.13	2.29	1.60
Ye et al. [35]	0.92	2.03	2.27	1.52
Le et al. [13]	0.93	2.05	2.22	1.56
Present	0.97	2.08	2.31	1.59

Table 2: 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. [2]	±0.250	1.36 ± 0.015	–	±0.75	1.40 ± 0.050	–
Liu et al. [21]	±0.339	1.35 ± 0.012	0.164	±0.69	1.31 ± 0.049	0.192
Calhoun [4]	±0.298	1.33 ± 0.014	0.175	±0.67	1.17 ± 0.058	0.202
Russell et al. [27]	±0.300	1.38 ± 0.007	0.169	±0.50	1.29 ± 0.022	0.195
Le et al. [13]	±0.323	1.37 ± 0.009	0.160	±0.43	1.34 ± 0.030	0.187
Present	±0.343	1.39 ± 0.009	0.161	±0.65	1.37 ± 0.040	0.194

the velocity field and pressure field have been computed, the drag and lift coefficients and the Strouhal number can be computed from the force as found in [11,20]. The time step is taken as $\Delta x/4$. The plot of streamline for $Re=40$ at steady state are shown in Fig. 3. For low Reynolds number, as expected, the flow gradually attains a steady state and the wake forms behind the cylinder symmetrically. The corresponding plot of the vorticity contours for $Re=40$ is also shown in Fig. 4(a). These results are found in a very good agreement with the results of [13]. At the steady state, the drag coefficients and the length of the recirculation zone are computed and are compared with other numerical results [4,8,13,20,27,35] as well as experimental results [5,29] in Table 1. 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 Figs. 4(b) and 4(c) show the vorticity contours at $Re=100$ and $Re=200$, respectively. The instability and vortex shedding can be seen from these figures. In Table 2, the drag coefficients, lift coefficients 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 $Re=100$ and $Re=200$ are presented in Fig. 5.

Next, the flow past an in-line oscillating cylinder is simulated to explore the capability of the present method in handling the case when the rigid boundary is moving. The simulation is performed in uniform flow at Reynolds number of 100 with the same numerical setup as the previous flow past a stationary cylinder except the present cylinder

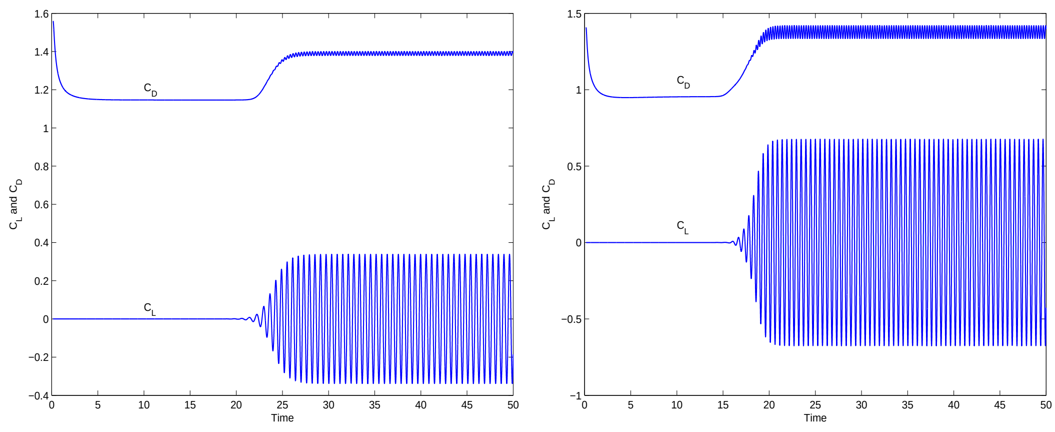


Figure 5: For Example 5.1 on flow past a stationary cylinder. Drag and lift coefficients versus time for $Re=100$ (left) and $Re=200$ (right).

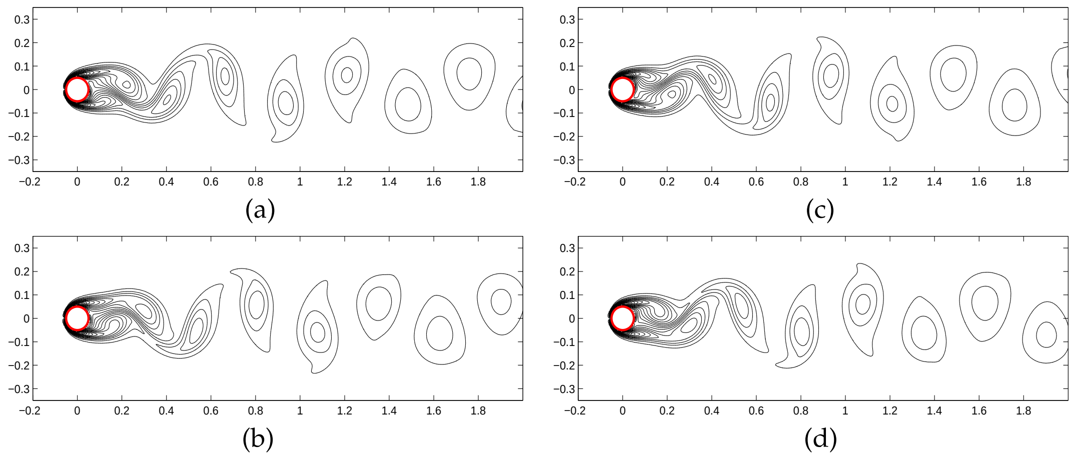


Figure 6: For Example 5.1 on flow past an in-line oscillating cylinder. Instantaneous vorticity contours for $Re=100$: (a) $t=\frac{T}{2}$, (b) $t=T$, (c) $t=\frac{3T}{2}$, (d) $t=2T$, where T is the oscillation period of the cylinder.

is now oscillating parallel to the free stream at a frequency $f_c=2f_q$, where f_q is the vortex shedding frequency of the fixed cylinder flow. The horizontal and vertical velocities of the oscillating cylinder are prescribed by $u_p=0.14d\cos(2\pi f_c t)$ and $v_p=0$, respectively. As expected, the in-line oscillation of the cylinder causes the phase-locking of the vortex shedding with the cylinder motion. The present average drag and lift coefficients are $C_L=0.98$ and $C_D=1.69$, respectively, which agree well with the results obtained in [30]. The plots of instantaneous vorticity contours over two oscillating periods of the cylinder are also shown in Fig. 6. These results can be comparable with those in [30].

Example 5.2. Rotational flow

In this example, we consider the rotational flow problem involving a fixed interface and irregular domain. Fig. 7 shows two typical irregular domains and the geometry of

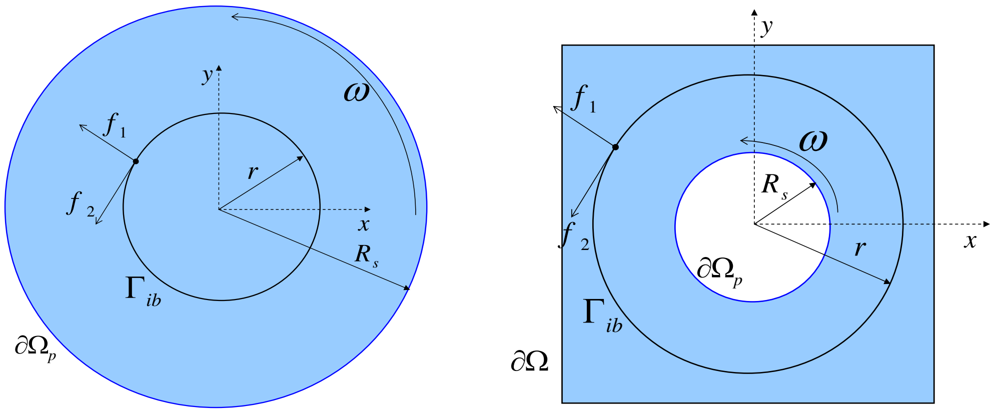


Figure 7: The domain of the simulation and the geometry of rotational flow.

simulation in the computation. In the first case, we first consider the external irregular domain as in Fig. 7 (left). In this case, the irregular domain Ω_p is a circle with radius $r=1$; the involved fixed interface is also a circle but with radius $r=\frac{1}{2}$, which is located at the center of the circular domain. We set 96 control points on both the rigid boundary and the inner interface in the simulation. The initial velocity and pressure are taken to be zero on the domain. We set $\mu=0.02$ and consider the solution at $t=10$. Along the inner interface, the normal and tangential stress are f_1 and $f_2=10\mu$, respectively. In the computation, we use a 64×64 grid on a computation domain of $[-1.5,1.5]\times[-1.5,1.5]$ and take the time step $\Delta t=\Delta x/16$. On the boundary of Ω_p, we first set the no-slip boundary conditions, i.e., $\omega=0$. The flow converges to a steady state in the end, as shown in Fig. 8, which corresponds to a rigid body motion inside the interface. Figs. 8(a) and 8(b) show the x-component of the velocity $\mathbf{u}$ and velocity field at $t=10$, respectively. Next we prescribe the boundary of irregular domain to rotate with angular velocity $\omega=-1$. The x-component of the velocity $\mathbf{u}$ and velocity field at $t=10$ are shown in Figs. 9(a) and 9(b), respectively. The motion of the steady is a clockwise rotation along the outer boundary and an anti-wise rotation along the inner interface. This point can be also seen from Fig. 10 more clearly, where we show the velocity distribution along the boundary and the interface.

In the second case, we consider the internal irregular domain as in Fig. 7 (right). In this case, the internal boundary of the irregular domain Ω_p is a circle with radius $r=0.3$; the involved fixed interface is a circle with radius $r=0.7$, which is located at the center of the square domain $[-1,1]\times[-1,1]$. We use 96 control points and 48 control points to represent the rigid boundary and the inner interface, respectively. The initial velocity and pressure are taken to be zero on the domain. We set $\mu=0.02$ and consider the solution at $t=10$. Along the outer interface, the normal and tangential stress are f_1 and $f_2=10\mu$, respectively. In this computation, we use a 64×64 grid and take the time step $\Delta t=\Delta x/16$. On both the outer and inner boundary of Ω_p, we first set the no-slip boundary conditions. Finally, the flow converges to a steady state, as shown in Fig. 11, which corresponds to

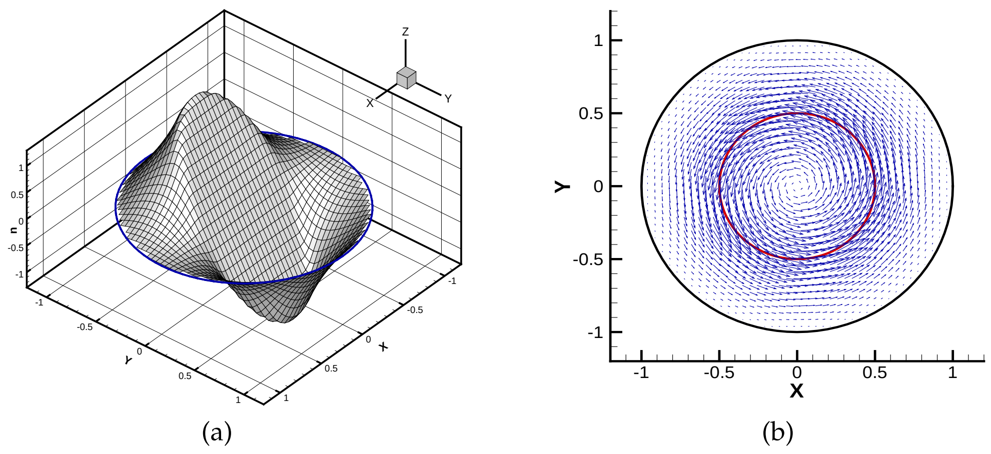


Figure 8: For Example 5.2 on rotational flow. (a) The x-component of the velocity field $\mathbf{u}$ and (b) the velocity field $\mathbf{u}$ with $\omega=0$ and $\mu=0.02$ at $t=10$.

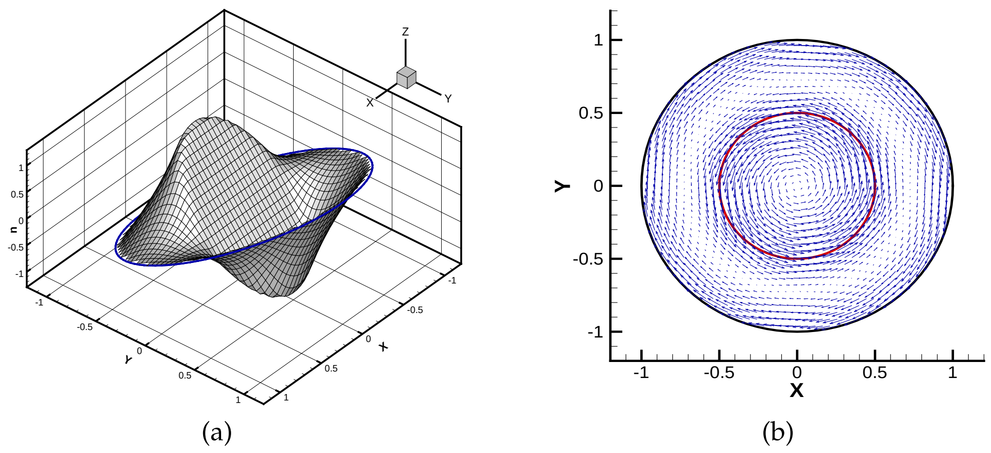


Figure 9: Same as Fig. 8 except with $\omega=-1$ and $\mu=0.02$ at $t=10$.

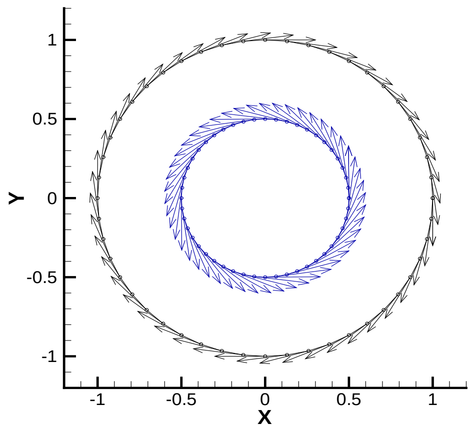


Figure 10: For Example 5.2 on rotational flow. The velocity distribution along the interface and the boundary with $\omega=-1$ and $\mu=0.02$ at $t=10$.

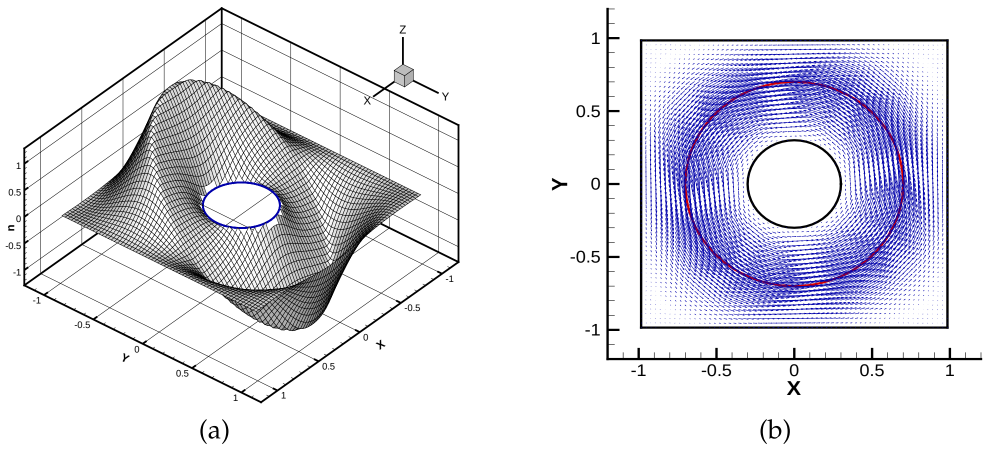


Figure 11: For Example 5.2 on circular Couette flow. (a) The x-component of velocity field $\mathbf{u}$ and (b) the velocity field $\mathbf{u}$ with $\omega=0$ and $\mu=0.02$ at $t=10$.

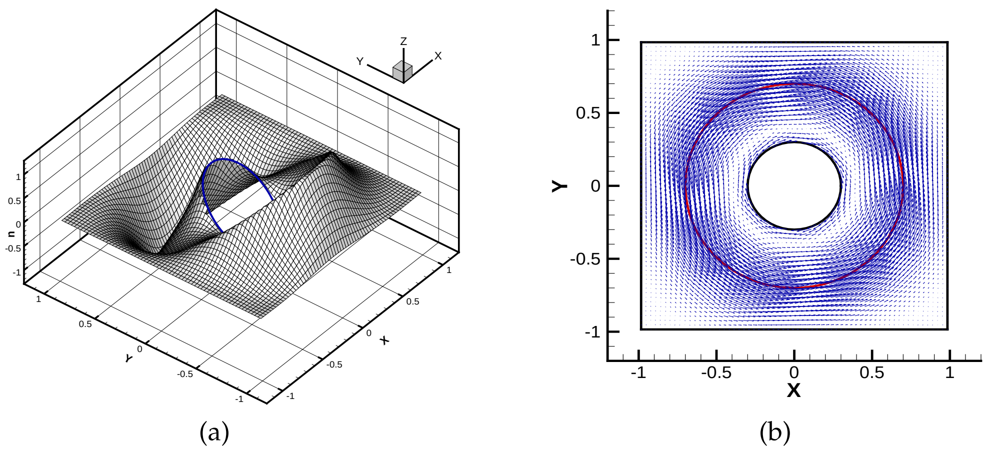


Figure 12: Same as Fig. 11 except with $\omega=-2$ and $\mu=0.02$ at $t=10$.

a rigid body motion inside the interface. Figs. 11(a) and 11(b) show the x-component of the velocity $\mathbf{u}$ and velocity field at $t=10$, respectively. Next we prescribe the boundary of irregular domain to rotate with angular velocity $\omega=-2$. The x-component of the velocity $\mathbf{u}$ and velocity field at $t=10$ are shown in Figs. 12(a) and 12(b), respectively. The motion of the steady is a clockwise rotation along the inner boundary and an anti-wise rotation along the outer interface.

Example 5.3. Elastic membrane

In the third example, we consider a deformable interface problem which involves an elastic membrane on the irregular circular domain. On the regular domain, this problem is used by Tu and Peskin [31] to test their immersed boundary method, by LeVeque and Li [15] to test their immersed interface method for Stokes flows, and by Lee and LeVeque

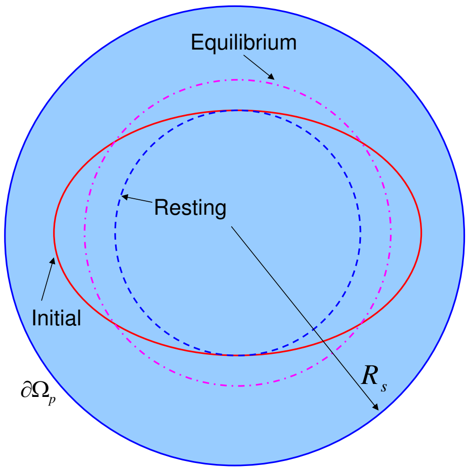


Figure 13: The interface at different states.

[14] to test their immersed interface method for Navier-Stokes equations. In our problem, the boundary of the irregular domain is a circular with radius $r=1.2$. The initial state of membrane (the solid line in Fig. 13, labeled "Initial") is an ellipse with the semi-major and semi-minor axes $a=0.75$, $b=0.5$, respectively, and the ellipse is located at the center of the circular domain. The unstretched state of membrane (the dashed line in Fig. 13, labeled "Resting") is a circle with radius $r_0=0.5$. The tension coefficient T_0 is set to 10 in this example unless it is stated otherwise, compared to the tension coefficient taken in the literatures [14], the problem of this example in this work is more stiffer.

Due to the restoring force, the ellipse will converge to a circle (the dash-dot line in Fig. 13, labeled "Equilibrium") with radius $r_e=\sqrt{ab}\approx 0.61237$, which is larger than the unstretched interface but has the same area as the initial ellipse because of the incompressibility of the enclosed fluid. So the interface is still stretched at the equilibrium state. We start our simulation by setting the initial velocity and pressure fields to zero, and the prescribed velocity at the rigid boundary of the irregular domain is set to zero, i.e, $\mathbf{u}|_{\partial\Omega_p}=\mathbf{0}$. In this example, we perform the simulations with a 64×64 grid on a computational domain of $[-1.5,1.5]\times[-1.5,1.5]$, 96 control points and 128 control points to represent the interface and the boundary of the irregular domain for all the cases unless otherwise stated. In our computations, we take the boundary condition as $\mathbf{u}|_{\partial\Omega}=\mathbf{0}$, and take the time step $\Delta t=\Delta x/16$.

In Figs. 14(a) and 14(b), we show the x-component of the velocity $\mathbf{u}$ and velocity field at $t=0.5$, respectively. The pressure distributions at $t=0.5$ and $t=1.2$ are presented in Figs. 15(a) and 15(b), respectively. The evolution of the semi-major and semi-minor axes with time is shown in Fig. 16. The interface oscillates as it settles down to the equilibrium state. These results can be comparable to those in [13, 14]. When $\mu=0.02$, we use a 128×128 grid and the same 160 control points to represent the interface and the boundary of the irregular domain. The x-component of the velocity $\mathbf{u}$ and pressure field at $t=1$ are

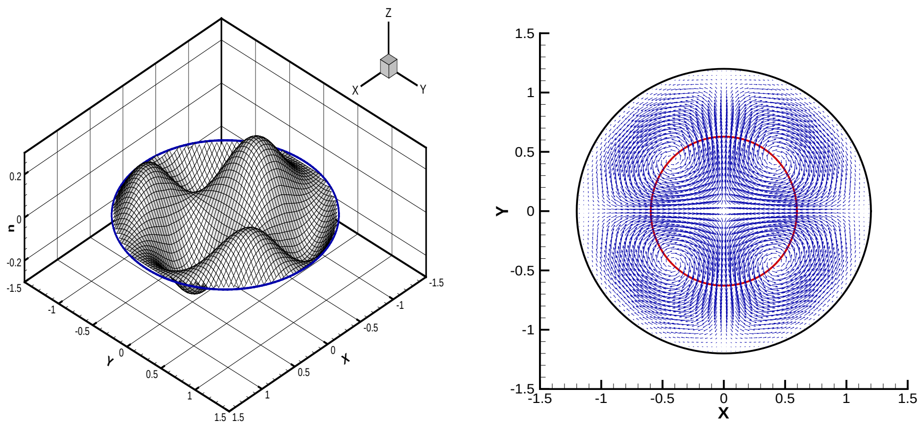


Figure 14: For Example 5.3 elastic membrane. (a) The x-component of the velocity field $\mathbf{u}$ and (b) velocity field $\mathbf{u}$ at $t=0.5$ with $\mu=0.1$ and $T_0=10$.

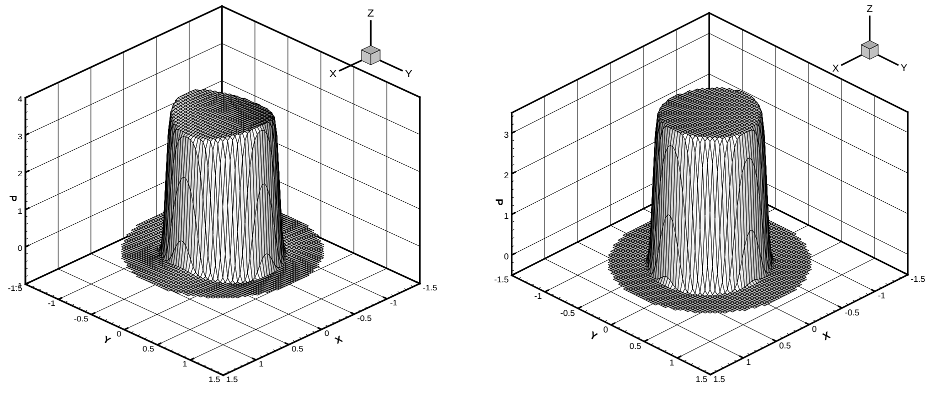


Figure 15: For Example 5.3 on elastic membrane. The pressure distribution at $t=0.5$ (a) and $t=1.2$ (b), with $\mu=0.1$ and $T_0=10$.

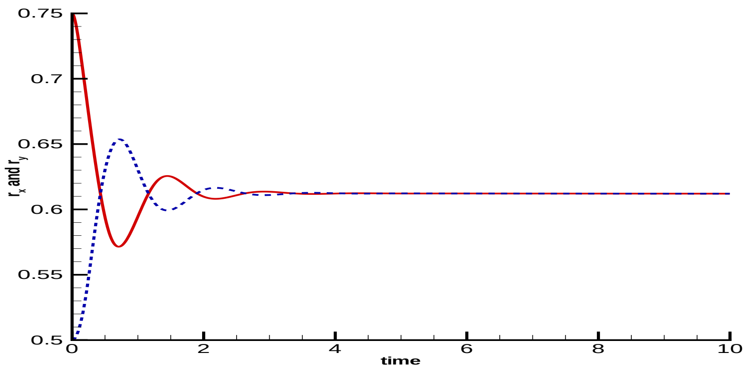
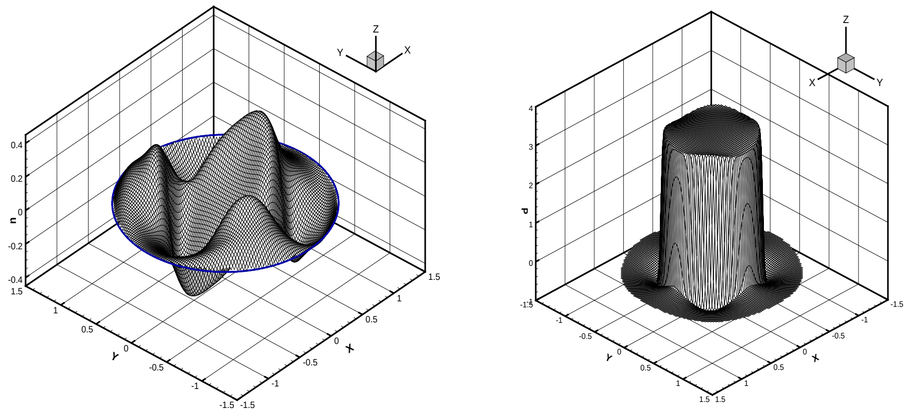


Figure 16: For Example 5.3 on elastic membrane. The evolution of r_x and r_y with $\mu=0.1$ and $T_0=10$. The interface oscillates as it converges to the equilibrium state.



(a) (b)

Figure 17: For Example 5.3 elastic membrane. (a) The x-component of the velocity field $\mathbf{u}$ and (b) pressure distribution at $t=1$ with $\mu=0.02$ and $T_0=10$.

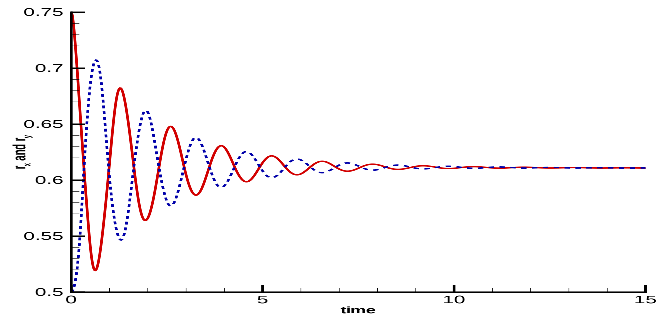


Figure 18: For Example 5.3 on elastic membrane. The evolution of r_x and r_y with $\mu=0.02$ and $T_0=10$. The interface oscillates as it converges to the equilibrium state.

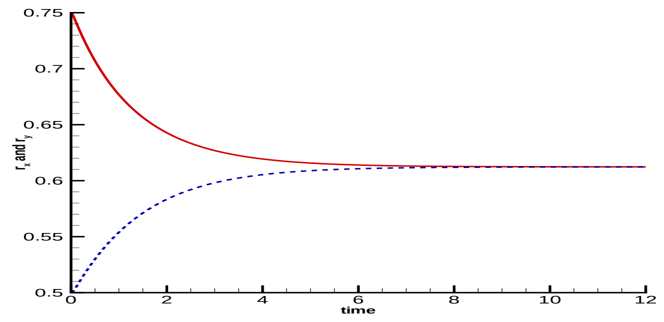


Figure 19: For Example 5.3 on elastic membrane. The evolution of r_x and r_y with $\mu=1$ and $T_0=10$. The interface relaxes gradually to the equilibrium state without oscillations.

presented in Figs. 17(a) and 17(b), respectively. In Fig. 18, we also show the evolution of the semi-major and semi-minor axes with time. Since the fluid viscosity is lower, the elastic membrane takes a longer time to oscillate before settling down to the equilibrium state. Fig. 19 shows the evolution of the semi-major and semi-minor axes of the ellipse with time when $\mu = 1$. In this case, the interface relaxes gradually to the equilibrium state without oscillations. It is noted that area lose often appears in the simulation of the immersed boundary method. An improved volume conservation scheme are constructed for the immersed boundary method in [26]. Here, however, the area loss is not significant (for example, the area loss is about 0.09% with maximum absolute error of 1.0874×10^{-3} in area when $\mu=0.1$), thus no modification is applied to improve volume conservation in our computation.

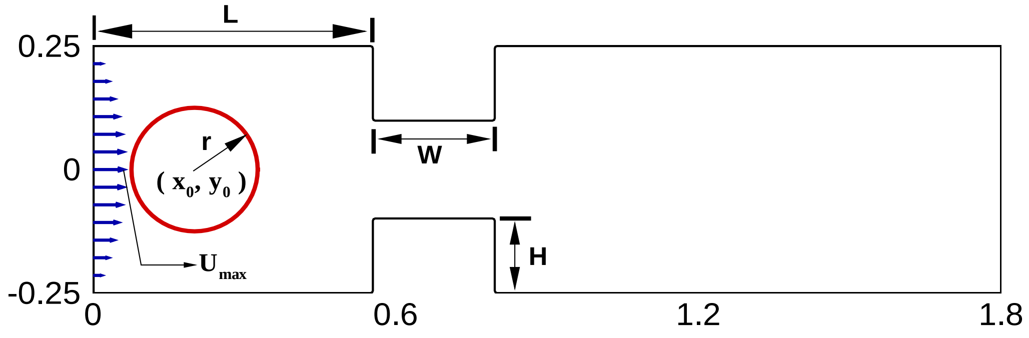


Figure 20: An elastic membrane in a constricted channel.

Example 5.4. Elastic membrane squeeze through a constriction

In the final example, as an application to simulate the similar biological processes like the motion of deformable cell in the micro-channel geometry, we consider the motion of a membrane in an irregular channel with a constriction. Fig. 20 illustrates the geometry of the constricted channel and the initial position of a membrane in the constriction used in the simulation. The geometric parameters W, H represent the width and height of constricted part, respectively, and L represents the distance between the left boundary of irregular domain and the entrance into constriction. (x_0,y_0) represents the initial position of the placed elastic membrane, and r represents the radius of elastic membrane. In the computation, the radii of inlet of channel and the height of the constriction is 0.25 and 0.15, respectively, thus the standard 5:1 contraction geometry is considered to investigate the motion of a single membrane through the constriction. In this example, we use a computational domain of $[0,1.8]\times[-0.3,0.3]$ and a 384×128 grid. The fluid viscosity $\mu=0.02$ and the stiff constant T_0= 1 have been used. We specify a parabolic velocity profile with $U_{\max}=1$ for the velocity at the inflow boundary. A homogeneous Neumann boundary condition for velocity is applied at the right boundary. The velocity is set to zero at the top and bottom boundaries. The no-slip boundary condition at the immersed rigid boundaries is enforced by imposing appropriate singular forces at the rigid boundaries.

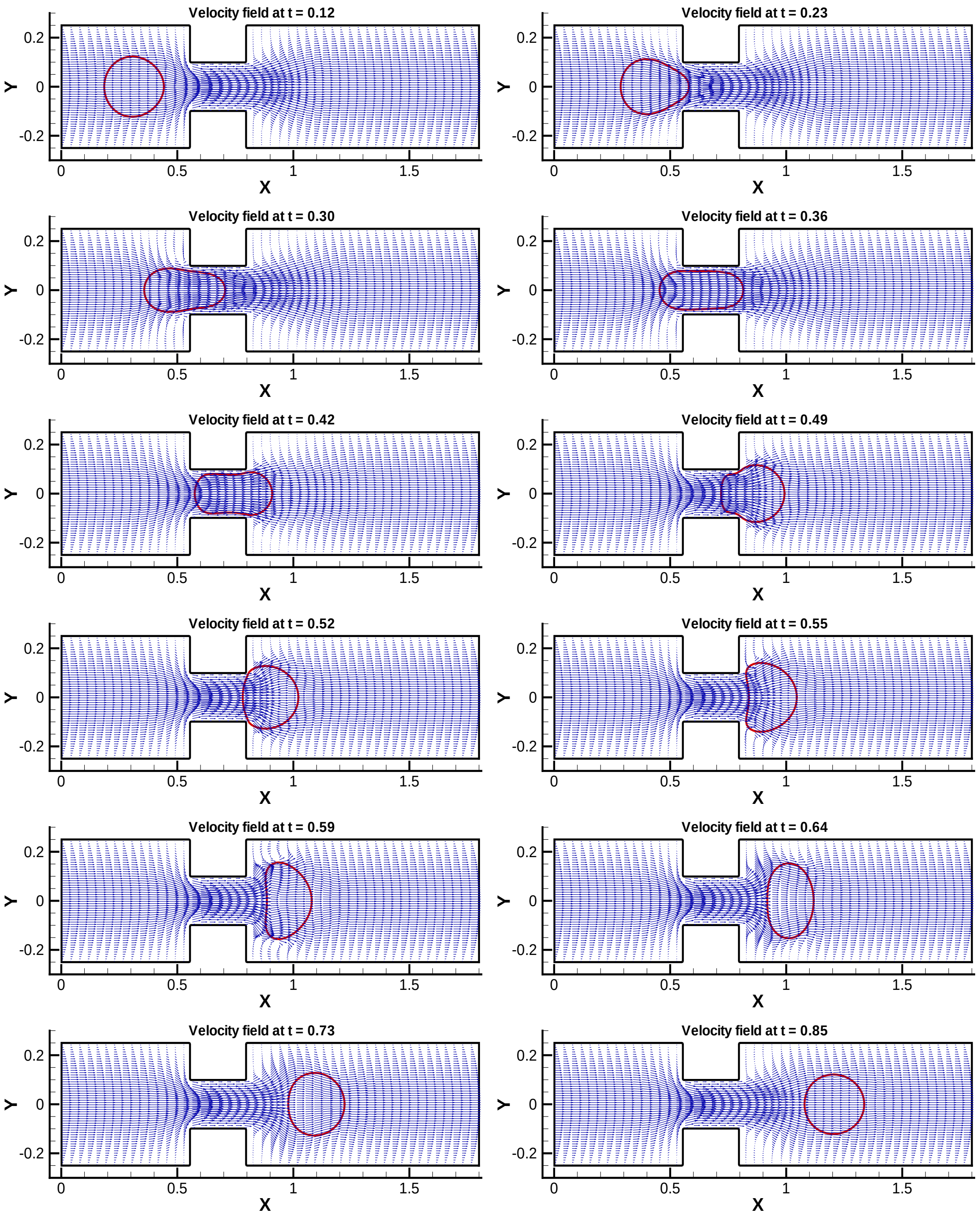


Figure 21: The positions and shapes of the elastic membrane and velocity fields at different times.

In the simulation, $(x_0,y_0)=(0.2,0.0)$ is taken, and the radius of initial elastic membrane is 0.125. The elastic membrane is pre-stretched from the undeformed state with a diameter of 0.2. We use 180 control points to represent the elastic membrane, and use 548 control points to present the rigid boundaries of the constricted channel. We take the time step $\Delta t=\Delta x/16$. Fig. 21 shows the positions and shapes of the elastic membrane squeezing through the constriction and the corresponding velocity fields at a series of times. From these figures, we can see the deformed elastic membrane how to squeeze through the constriction clearly.

6 Concluding remarks

In this paper, we have presented an indirect-forcing immersed boundary method for solving the incompressible Navier-Stokes equations involving the interfaces and irregular domains. The method combines the immersed boundary method with a front tracking representation of the interface and rigid boundary on a uniform Cartesian grid. The main advantage of the method is that the prescribed velocity condition at the rigid boundary is exactly satisfied. It is a rather straightforward manner to extend the current algorithm to solve the problems involving the multi-connected irregular domains and multiple interfaces. Current method can be also extended straightforwardly to general two phase flow with different density and viscosity on irregular domain. Our method is capable of solving 3D incompressible biological flow problems involving the deformable interfaces and irregular domains like the motion of deformable cell in a complex 3D micro-channel geometry. A 3D version of the method will be reported in the near future.

Acknowledgments

The authors would like to thank the referees for the valuable suggestions on the revision of the manuscript. The last author's research is supported by the grant AcRF RG59/08 M52110092.

References

[1] J. Adams, P. Swarztrauber, and R. Sweet, FISHPACK: Efficient FORTRAN subprograms for the solution of separable eliptic partial differential equations, 1999. Available on the web at `http://www.scd.ucar.edu/css/software/fishpack/`.

[2] M. Braza, P. Chassaing, and 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), pp. 79–130.

[3] D.L. Brown, R. Cortez, and M.L. Minion, Accurate projection methods for the incompressible Navier-Stokes equations, J. Comput. Phys., 168 (2001), pp. 464–499.

[4] D. Calhoun, A Cartesian grid method for solving the two-dimensional streamfunction-vorticity equations in irregular regions, J. Comput. Phys., 176 (2002), pp. 231–275.

[5] M. Coutanceau and R. Bouard, Experimental determination of the main features of the viscous flow in the wake of a circular cylinder in uniform translation, Part 1. Steady flow, J. Fluid. Mech., 79 (1977), pp. 231–256.

[6] E.A. Fadlun, R. Verzicco, and P. Orlandi, Combined immersed boundary finite-difference methods for three-dimensional complex flows simulations, J. Comput. Phys., 161 (2000), pp. 35–60.

[7] A.L. Fogelson, Continuum models of platelet aggregation: Formulation and mechanical properties, SIAM J. Applied Math., 52 (1992), pp. 1089–1110.

[8] B. Fornberg, A numerical study of steady viscous flow past a circular cylinder, J. Fluid. Mech., 98 (1980), pp. 819–855.

[9] D. Goldstein, R. Handler, and L. Sirovich, Modeling a no-slip flow with an external force field, J. Comput. Phys., 105 (1993), pp. 354–366.

[10] J. Kim and P. Moin, Application of a fractional step method to incompressible Navier-Stokes equations, J. Comput. Phys., 59 (1985), pp. 308–323.

[11] M.-C. Lai and C.S. Peskin, An immersed boundary method with formal second order accuracy and reduced numerical viscosity, J. Comput. Phys., 160 (2000), pp. 707–719.

[12] D.V. Le, B.C. Khoo, and Z. Li, An Implicit-forcing Immersed Interface Method for the Incompressible Navier-Stokes Equations, AMS Contemporary Mathematics, 466 (2008), pp. 73–94.

[13] D.V. Le, B.C. Khoo, and J. Peraire, An immersed interface method for viscous incompressible flows involving rigid and flexible boundaries, J. Comput. Phys., 220 (2006), pp. 109–138.

[14] L. Lee, An immersed interface method for incompressible Navier-Stokes equations, SIAM J. Sci. Comput., 25 (2003), pp. 832–856.

[15] R.J. LeVeque and Z. Li, Immersed interface methods for Stokes flow with elastic boundaries or surface tension, SIAM J. Sci. Comput., 18 (1997), pp. 709–735.

[16] R.J. LeVeque and Z. Li, The immersed interface method for elliptic equations with discontinuous coefficients and singular sources, SIAM J. Numer. Anal., 31 (1994), pp. 1019–1044.

[17] Z. Li and K. Ito, The immersed interface method-numerical solutions of PDEs involving interfaces and irregular domains. SIAM Frontiers Appl Math 2006; 33.

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

[19] Z. Li and C. Wang, A fast finite difference method for solving Navier-Stokes equations on irregular domains, Comm. Math. Sci., 1 (2003), pp. 180–196.

[20] A.L.F. Lima E. Silva, A. Silveira-Neto, and J.J.R Damasceno, Numerical simulation of two-dimensional flows over a circular cylinder using the immersed boundary method, J. Comput. Phys., 189 (2003), pp. 351–370.

[21] C. Liu, X. Sheng, and C. H. Sung, Preconditioned multigrid methods for unsteady incompressible flows, J. Comput. Phys., 139 (1998), pp. 35–57.

[22] J. Mohd-Yusof, Combined immersed boundary/B-splines methods for simulations of flows in complex geometry, Annual Research Brieft s, Center for Turbulence Research, pages 317–327, 1997.

[23] Y. Mori and C.S. Peskin, Implicit second-order immersed boundary methods with boundary mass, Compt. Methods Appl. Mech. Engrg., 197 (2008), pp. 2049–2067.

[24] C.S. Peskin, Numerical analysis of blood flow in the heart, J Comput Phys., 25 (1977), pp. 220–52.

[25] C.S. Peskin, The immersed boundary method, Acta Numerica, 11 (2002), pp. 479–517.

[26] C.S. Peskin and B.F. Printz, Improved volume conservation in the computation of flows with

immersed elastic boundaries, J. Comput. Phys., 105 (1993), pp. 33–36.

[27] D. Russell and Z.J. Wang, A Cartesian grid method for modeling multiple moving objects in 2D incompressible viscous flow, J. Comput. Phys., 191 (2003), pp. 177–205.

[28] Y. Sadd, GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems, SIAM J. Sci. Stat. Comput., 7 (1986), pp. 856–869.

[29] D.J. Tritton, Experiments on the flow past a circular cylinder at low Reynolds numbers, J. Fluid. Mech., 6 (1959), pp. 547–567.

[30] S-W. Su, M-C. Lai, and C-A. Lin, An immersed boundary method for simulating the interaction of a fluid with moving boundaries, Comput. Fluids, 36 (2007), pp. 313-324.

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

[32] H.S. Udaykumar, R. Mittal, P. Rampunggoon, and A. Khanna, A sharp interface Cartesian grid method for simulating flows with complex moving boundaries, J. Comput. Phys., 174 (2001), pp. 345–380.

[33] M. Uhlmann, An immersed boundary method with direct forcing for the simulation of particulate flows, J. Comput. Phys., 209 (2005), pp. 448–476.

[34] N.T. Wang and A.L. Fogelson, Computational methods for continuum models of platelet aggregation, J. Comput. Phys, 151 (1999), pp. 649–675.

[35] T. Ye, R. Mittal, H.S. Udaykumar, and W. Shyy, An accurate Cartesian grid method for viscous incompressible flows with complex immersed boundary, J. Comput. Phys., 156 (1999), pp. 209–240.