Commun. Comput. Phys.
doi: 10.4208/cicp.OA-2020-0137

Vol. **30**, No. 1, pp. 227-254
July 2021

A Simple 3D Immersed Interface Method for Stokes Flow with Singular Forces on Staggered Grids

Weiyi Wang[1] and Zhijun Tan[1,2,*]

[1] *School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou 510275, China.*
[2] *Guangdong Province Key Laboratory of Computational Science, Sun Yat-sen University, Guangzhou 510275, China.*

Received 15 July 2020; Accepted (in revised version) 8 December 2020

Abstract. In this paper, a fairly simple 3D immersed interface method based on the CG-Uzawa type method and the level set representation of the interface is employed for solving three-dimensional Stokes flow with singular forces along the interface. The method is to apply the Taylor's expansions only along the normal direction and incorporate the jump conditions up to the second normal derivatives into the finite difference schemes. A second order geometric iteration algorithm is employed for computing orthogonal projections on the surface with third-order accuracy. The Stokes equations are discretized involving the correction terms on staggered grids and then solved by the conjugate gradient Uzawa type method. The major advantages of the present method are the special simplicity, the ability in handling the Dirichlet boundary conditions, and no need of the pressure boundary condition. The method can also preserve the volume conservation and the discrete divergence free condition very well. The numerical results show that the proposed method is second order accurate and efficient.

AMS subject classifications: 65M06, 76D07

Key words: 3D immersed interface method, CG-Uzawa method, Stokes flow, level set method, staggered grids, singular forces.

1 Introduction

Numerical simulations for the interface problems have become one of the hot topics in computational fluid mechanics because these problems have important practical implications in many applications in physics and biology. The challenges for flow problems

*Corresponding author. *Email addresses:* `tzhij@mail.sysu.edu.cn` (Z. Tan), `wangwy37@mail2.sysu.edu.cn` (W. Wang) with moving interfaces are how to track and evolve the motion of the interface accurately, how to conserve the volume of the enclosed interface well, and how to preserve the discrete divergence free constraint well. This paper will focus on three-dimensional incompressible viscous Stokes flows in a Cartesian grid. C.Peskin proposed an immersed boundary method (IB-Method) which is originally aimed at the fluid dynamics of blood flow in [24,25]. This method converts the complex structure of the interface into the force source terms in the momentum equations in Cartesian grid and applies these forces to grid points near the interface. More precisely, they track the interface with the front-tracking method which has better volume conservation than level-set method, but there are still some problems in three-dimensional case. In the past few decades the IB-Method was widely applied to many other problems such as biofilm processes and biomechanics, see [3, 11, 23, 26] for a detail. And there are some other Cartesian grid methods such as [1,2,4,5,7,15,35] which have been proposed for the interface problem. This series of approaches involve the methods such as spectral method, finite volume method and finite element method. LeVeque employed the implicit immersed boundary method to simulate viscous incompressible flows with immerse elastic membranes for three-dimensional problem [14] in 2009. IB-Method is very efficient but it doesn't achieve global second order spatial accuracy in general. About the analysis for convergence, the interested readers are referred to the references [42, 43].

Alternative method to solve the interface problem is immersed interface method (IIM) which was originally proposed for elliptic interface problem by Li and LeVeque [16]. A standard five-point central finite difference scheme is used at the points away from the interface in IIM. For the points near the interface the correction terms computed from the jump conditions are incorporated to the difference scheme. So that the accuracy can be guaranteed in the whole domain. The most significant advantage of the IIM is that it can achieve global second order accuracy [19, 34]. Then the IIM was applied to Stokes and Navier-Stokes flows in [17, 22]. In [18], a fast iterative IIM for elliptic interface problem with discontinuous coefficient was proposed. The core idea of this fast algorithm is to introduce an augmented variable. The GMRES iterative method was adopted to compute the augmented variable. Later on, the fast algorithm was extended to Stokes equations with continuous viscosity in [33] and discontinuous viscosity in [21]. A detail overview of IIM can be found in [20]. In recent years, a coupled immersed interface and level set method for three-dimensional steady Stokes equations has been proposed in [37]. In this method the Stokes equations were divided into two Poisson systems, one for the velocity field and the other for the pressure field. As discussed in [27], the boundary conditions for pressure should be considered. For testing purpose the Dirichlet and Neumann boundary conditions have been used. But there are some additional conditions for pressure.

Recently, Lai proposed a simple version of IIM [13] for Stokes equations in 2D, which applies the Taylor expansion only along the normal direction and then incorporates the correction terms into the difference scheme. Fewer jump conditions are required when computing the correction terms compared to the original IIM, which leads to fairly simple implementation and great advantages especially in 3D case. But in the process of actual numerical simulations, the accuracy of orthogonal projection points should be more accurate. The numerical results show that this Stokes solver based on simplified IIM can produce second order solutions for velocity. However the Stokes solver proposed by Lai is based on the method which decomposes the Stokes equations into several separate Poisson systems. The boundary conditions for pressure field need to be considered. Alternative method to compute pressure field in Stokes equations is Uzawa type method. Tau [32] applied a CG Uzawa type method in two-dimensional and three-dimensional steady Stokes equations in a continuous domain. This solver was applied to two-dimensional Stokes equations with interface based on IIM in [30] and [29] by Tan. And they further extend their work into two-phase flows in [31]. The Uzawa type method can reach second order accuracy for both velocity and pressure. And the iteration process is quite efficient. Later the simplified IIM was extended to electrohydrodynamic simulations in [9, 38].

Some works for three-dimensional flow problems have been proposed over last few years, e.g. LeVeque [14], they consider an implicit IB-method simulating viscous incompressible flows with immerse elastic membranes. The necessary Cartesian jump conditions computed from given principal jump conditions are discussed in [40] based on triangular mesh representation of an interface. And IB-method is applied to do simulation of elastic capsules in three-dimensional shear flows in [10].

The main purpose of this paper is to apply the simple version of IIM to three-dimensional Stokes equations with immersed interface. The algorithm couples a level-set representation of the interface. A CG Uzawa type method similar to that mentioned in [32] on a MAC staggered grid for Stokes equations is applied, where the number of CG iteration is independent of the mesh size. Thus the 3D IIM Stokes solver proposed here is very efficient. The gradient and divergence of both velocity and pressure can be easily evaluated with the staggered grid. The present method avoids considering the boundary conditions for pressure and is very flexible for solving different types of flow boundary conditions. And the method can also preserve well volume conservation and discrete divergence free condition. Although the particular interest in this work is focused on steady Stokes flows with interfaces, it is fairly straightforward to extend the proposed solver for almost all other types of fluid flows (such as the 3D unsteady Stokes equations, the 3D steady Navier-Stokes equations and the 3D unsteady Navier-Stokes equations) with interfaces and singular forces, which only needs a simple extension from the present Stokes solver to the efficient generalized Stokes solver or the Navier-Stokes solver. So the proposed method is also flexible for different types of incompressible viscous flow problems with interfaces and singular forces. The above advantages mentioned especially in fairly simple implement make the application of the method more general. Another contribution of this work is to devise a second order geometric iteration algorithm for computing orthogonal projections on the surface with third-order accuracy of projections referring to the exact orthogonal projections. The numerical results show that the overall scheme is indeed second order accurate. As for the time step, a 5th WENO and 3rd TVD-RK method is used to move the interface and reinitialize the level-set function. Jump conditions for pressure and velocity in three-dimensional case are derived in [12].

An outline of this paper is as follow. In Section 2, a solver for three-dimensional Poisson equations with an interface using simple IIM will be presented. The Stokes equations with an immersed interface is presented in Section 3. The jump conditions across the interface for Stokes flows are given in Section 4. The numerical algorithm are presented in Section 5. Some numerical results are shown in Section 6 and some conclusions are given in Section 7.

2 A simple 3D IIM Poisson solver

2.1 Summary of the simple 3D IIM Poisson solver

In order to describe the implement of a simple 3D IIM, the 3D Poisson equation on a computational domain Ω with an immersed interface Γ is considered. The interface Γ divides the domain into two regions Ω^+ and Ω^- with $\Omega=\Omega^+\cup\Gamma\cup\Omega^-$, where Ω^+ is used to express the exterior region of the interface, and Ω^- is enclosed by the interface. The Poisson interface problem can be written as

$$\Delta u = g, \quad \text{in } \Omega\backslash\Gamma, \tag{2.1}$$

$$[u]=\omega, \quad \left[\frac{\partial u}{\partial \mathbf{n}}\right]=\psi, \quad \text{on } \Gamma, \tag{2.2}$$

$$u=u_{\rm b}, \quad \text{on } \partial\Omega. \tag{2.3}$$

The right hand side term g may have a finite jump across the interface Γ. Here ω and ψ are the known functions defined only on the interface Γ, and $\mathbf{n}$ is the unit normal vector pointing to Ω^+ side. The jump $[u]$ is defined as the difference of the limiting value of u from Ω^+ to Ω^- sides.

Different from classical IIM [16] making the Taylor expansion along each axis direction, an extension of the simple 2D IIM in [13] to 3D case is employed in this work. Let introduce an uniform Cartesian grid in Ω with the mesh width $h=\Delta x=\Delta y=\Delta z$, and denote $u_{i,j,k}$ as the discretized solution at the grid point $\boldsymbol{x}_{i,j,k}=(x_{\min}+ih, y_{\min}+jh, z_{\min}+kh)$. As described in [6], the grid point is classified as either a regular point or an irregular point. A standard seven-point difference scheme is used at regular points, while a correction term is added to the finite difference scheme at an irregular point. In order to maintain second order spatial accuracy in the solution, the correction term, which depends on the jumps $[u]$ and $[\frac{\partial u}{\partial \mathbf{n}}]$, will be obtained below.

Without loss of generality, it is assumed that $\boldsymbol{x}_{i,j,k}$ is an irregular point inside the interface, while the grid points $\boldsymbol{x}_{i-1,j,k}$, $\boldsymbol{x}_{i,j+1,k}$ and $\boldsymbol{x}_{i,j,k-1}$ are outside the interface, as shown in Fig. 1. Then the seven-point finite difference scheme for Laplacian equation at $\boldsymbol{x}_{i,j,k}$ can

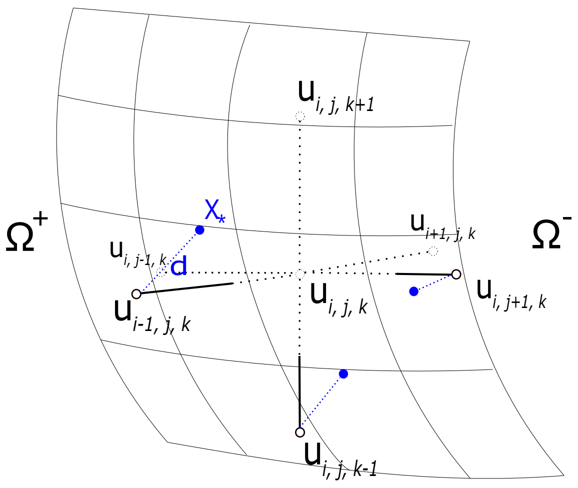


Figure 1: The point $u_{i,j,k}$ and its six adjacent points. The dotted line is used to show the points inside the interface, while the solid line is used to show the points outside the interface.

be written as

$$\begin{aligned}
\Delta_h u_{i,j,k} =& \frac{u_{i-1,j,k}-2u_{i,j,k}+u_{i+1,j,k}}{h^2}+\frac{u_{i,j-1,k}-2u_{i,j,k}+u_{i,j+1,k}}{h^2}+\frac{u_{i,j,k-1}-2u_{i,j,k}+u_{i,j,k+1}}{h^2}\\
=&\frac{u^+_{i-1,j,k}-2u^-_{i,j,k}+u^-_{i+1,j,k}}{h^2}+\frac{u^-_{i,j-1,k}-2u^-_{i,j,k}+u^+_{i,j+1,k}}{h^2}+\frac{u^+_{i,j,k-1}-2u^-_{i,j,k}+u^-_{i,j,k+1}}{h^2}\\
=&\frac{u^-_{i-1,j,k}-2u^-_{i,j,k}+u^-_{i+1,j,k}}{h^2}+\frac{u^-_{i,j-1,k}-2u^-_{i,j,k}+u^-_{i,j+1,k}}{h^2}+\frac{u^-_{i,j,k-1}-2u^-_{i,j,k}+u^-_{i,j,k+1}}{h^2}\\
&+\frac{u^+_{i-1,j,k}-u^-_{i-1,j,k}}{h^2}+\frac{u^+_{i,j+1,k}-u^-_{i,j+1,k}}{h^2}+\frac{u^+_{i,j,k-1}-u^-_{i,j,k-1}}{h^2}\\
=&u_{xx}\left(\mathbf{x}_{i,j,k}\right)+u_{yy}\left(\mathbf{x}_{i,j,k}\right)+u_{zz}\left(\mathbf{x}_{i,j,k}\right)+\mathcal{O}\left(h^2\right)+\frac{u^c_{i-1,j,k}}{h^2}+\frac{u^c_{i,j+1,k}}{h^2}+\frac{u^c_{i,j,k-1}}{h^2}\\
=&\Delta u_{i,j,k}-\mathcal{C}\{\Delta u_{i,j,k}\}+\mathcal{O}\left(h^2\right)=g_{i,j,k}-\mathcal{C}\{\Delta u_{i,j,k}\}+\mathcal{O}\left(h^2\right),
\end{aligned} \tag{2.4}$$

where $\mathcal{C}\{\Delta u_{i,j,k}\}=-\frac{1}{h^2}\left(u^c_{i-1,j,k}+u^c_{i,j+1,k}+u^c_{i,j,k-1}\right)$ is the correction term for discretizing Laplacian at an irregular point (i,j,k). The above symbol u^- is used for the points inside the interface, while u^+ for the points outside the interface. So the value $u^+_{i-1,j,k}$ is the real value of u at the grid point $(i-1,j,k)$. Let $u^-_{i-1,j,k}$ be defined as the ghost value of u at the grid point $(i-1,j,k)$, $u^-_{i,j+1,k}$ and $u^-_{i,j,k-1}$ are defined in the same way. The ghost value can be regarded as an extension from the inside of the interface to the outside. Let $\mathbf{X}_*$ be the orthogonal projection of $\boldsymbol{x}_{i-1,j,k}$ on the interface, and denote d as the signed distance from $\boldsymbol{x}_{i-1,j,k}$ to $\mathbf{X}_*$. By applying the Taylors expansion along the normal direction at $\mathbf{X}_*$, the correction term $u^c_{i-1,j,k}$ can be derived as

$$\begin{aligned}
u^c_{i-1,j,k} &= u^+_{i-1,j,k} - u^-_{i-1,j,k} \\
&= \left(u^+_* + d\frac{\partial u^+_*}{\partial \mathbf{n}} + \frac{d^2}{2}\frac{\partial^2 u^+_*}{\partial \mathbf{n}^2} + \mathcal{O}\left(h^3\right)\right) - \left(u^-_* + d\frac{\partial u^-_*}{\partial \mathbf{n}} + \frac{d^2}{2}\frac{\partial^2 u^-_*}{\partial \mathbf{n}^2} + \mathcal{O}\left(h^3\right)\right) \\
&= [u]_{\mathbf{X}_*} + d\left[\frac{\partial u}{\partial \mathbf{n}}\right]_{\mathbf{X}_*} + \frac{d^2}{2}\left[\frac{\partial^2 u}{\partial \mathbf{n}^2}\right]_{\mathbf{X}_*} + \mathcal{O}\left(h^3\right).
\end{aligned} \tag{2.5}$$

The orthogonal projection $\mathbf{X}_*$ of grid point on the interface can be found in a way similar to that in [20].

Note the relationship between the surface Laplacian ($\nabla^2_s = (\mathbf{I}-\mathbf{n}\otimes\mathbf{n})\nabla]\cdot[(\mathbf{I}-\mathbf{n}\otimes\mathbf{n})\nabla]$) and the Laplacian ($\Delta$) of an arbitrary scalar function q mentioned in [39], i.e., $\Delta q = \nabla^2_s q + \kappa\frac{\partial q}{\partial \mathbf{n}} + \frac{\partial^2 q}{\partial \mathbf{n}^2}$, where $\kappa = 2H = \nabla\cdot\mathbf{n}$, and H is the mean curvature, thus the jump in the second normal derivative of u at projection point $\mathbf{X}_*$ (i.e., $\left[\frac{\partial^2 u}{\partial \mathbf{n}^2}\right]_{\mathbf{X}_*}$) can be written as

$$\left[\frac{\partial^2 u}{\partial \mathbf{n}^2}\right]_{\mathbf{X}_*} = [g]_{\mathbf{X}_*} - \kappa_{\mathbf{X}_*}\left[\frac{\partial u}{\partial \mathbf{n}}\right]_{\mathbf{X}_*} - \nabla^2_s[u]_{\mathbf{X}_*}. \tag{2.6}$$

Here $\nabla^2_s[u]_{\mathbf{X}_*}$ can be computed using the least squares interpolation in [6]. The correction term $u^c_{i-1,j,k}$ can be further written as

$$u^c_{i-1,j,k} = [u]_{\mathbf{X}_*} + d\left[\frac{\partial u}{\partial \mathbf{n}}\right]_{\mathbf{X}_*} + \frac{d^2}{2}\left([g]_{\mathbf{X}_*} - \kappa_{\mathbf{X}_*}\left[\frac{\partial u}{\partial \mathbf{n}}\right]_{\mathbf{X}_*} - \nabla^2_s[u]_{\mathbf{X}_*}\right) + \mathcal{O}\left(h^3\right). \tag{2.7}$$

The correction terms $u^c_{i,j+1,k}$ and $u^c_{i,j,k-1}$ can be derived in a similar way. Finally, the resulting linear system in Eq. (2.4) can be efficiently solved by some fast Poisson solvers like Fishpack [28].

2.2 An iteration method for orthogonal projections on the surface

In this work, the necessary orthogonal projections of irregular grid points on the surface need to be computed when applying Taylor expansion along the normal direction. An approximate orthogonal projection method in [20] is employed to find the approximate orthogonal projection point $\mathbf{X}^*_{ijk} = (X_i, Y_j, Z_k)$ on the surface at an irregular grid point $\mathbf{x}_{ijk} = (x_i, y_j, z_k)$, which is obtained by solving the following quadratic equation for α:

$$0 = \varphi\left(\mathbf{X}^*_{ijk}\right) = \varphi\left(\mathbf{x}_{ijk} + \alpha\mathbf{p}\right) \approx \varphi\left(\mathbf{x}_{ijk}\right) + \left|\nabla\varphi\left(\mathbf{x}_{ijk}\right)\right|\alpha + \frac{1}{2}\left(\mathbf{p}^T \mathrm{He}\left(\varphi\left(\mathbf{x}_{ijk}\right)\right)\mathbf{p}\right)\alpha^2. \tag{2.8}$$

Here $\mathbf{p} = \nabla\varphi_{ijk}/|\nabla\varphi_{ijk}|$ is the normal of the level set and $\mathrm{He}(\varphi(\mathbf{x}_{ijk}))$ is the Hessian matrix at $\mathbf{x}_{ijk}$, respectively, which are computed using the standard central finite difference

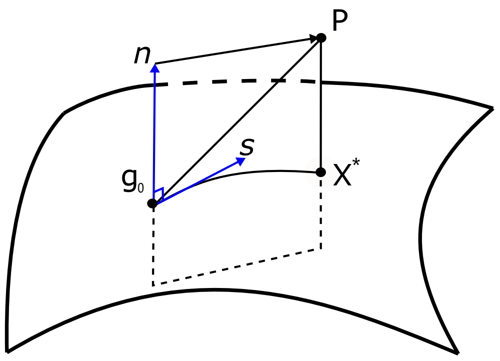


Figure 2: Description of geometric iteration method: P is the irregular point, and $\mathbf{X}^*$ is the orthogonal projection point; g_0 is initial guess; **n** and **s** are the normal direction and tangential direction of surface at point g_0, respectively.

scheme. The computed projections are typically on the surface with third-order accuracy. By applying the approximate orthogonal projection method, the classical IIM can achieve satisfactory second-order accuracy with using the general Taylor expansion. However, it should be pointed that the approximate orthogonal projection method can not be directly applied to the current simple IIM, otherwise it will lead to a decrease in overall accuracy of the solution for the case of not a spherical surface. This is because the present IIM makes the Taylor expansion only along the normal direction and needs also consider the accuracy referring to the exact orthogonal projection point. The above approximate orthogonal projection method computes the normal p at the irregular grid point $\mathbf{x}_{ijk}$ not at the projection point $\mathbf{X}^*_{ijk}$, which leads to only second-order accuracy of projection point referring to the exact orthogonal projection point. This can be seen from the later numerical experiments.

In this paper, a geometric iteration algorithm [36] based on a second-order Taylor iteration is employed. The core idea of this algorithm is to modify the projection point along the normal section line of the surface as shown in Fig. 2. The interested readers are referred to [8,36] for more details. In our actual computation, the approximate projection point computed by the approximate orthogonal projection method is chosen to be the initial point for the present iteration method, which is very close to the exact orthogonal projection point with second-order accuracy. Therefore, only few iterations (say about 2$\sim$ 3 times) in the actual computations are needed during the geometric iteration. The later numerical results show that the present geometric iteration algorithm can reach third-order accuracy referring to the exact orthogonal projection point.

3 The model of Stokes flow

Let Ω be a three-dimensional bounded domain involving an interface Γ. The interface Γ separates the fluid into two regions Ω^+ and Ω^- with $\Omega = \Omega^+ \cup \Gamma \cup \Omega^-$, where Ω^+ is used to express the exterior region of the interface, and Ω^- is inside the interface. The following Stokes equations with singular forces are considered:

$$\nabla p = \mu\Delta \boldsymbol{u} + \boldsymbol{F}(\boldsymbol{x}) + \boldsymbol{G}(\boldsymbol{x}), \quad \text{in } \Omega, \tag{3.1}$$

$$\nabla\cdot\boldsymbol{u} = 0, \quad \text{in } \Omega, \tag{3.2}$$

$$\boldsymbol{u} = \boldsymbol{u}_{\mathrm{b}}, \quad \text{on } \partial\Omega, \tag{3.3}$$

where $\boldsymbol{u} = (u,v,w)$ is the fluid velocity, p is the fluid pressure, μ is the fluid viscosity, and $\boldsymbol{x} = (x,y,z)$ is the Cartesian coordinate variable. Here $\boldsymbol{G}$ is the external force which may have a jump across the interface, and $\boldsymbol{F}$ is the singular force exerted by the immersed interface, which has a Dirac delta function singularity and can be written as

$$\boldsymbol{F}(\boldsymbol{x}) = \int_\Gamma \boldsymbol{f}\delta(\boldsymbol{x}-\mathbf{X})ds. \tag{3.4}$$

Here $\mathbf{X}$ is the point on the interface, $\boldsymbol{F}$ is the force density defined on the interface, and $\delta(\cdot)$ is the 3D Dirac delta function. For the model of Stokes flow with surface tension, $\boldsymbol{f} = \sigma\kappa\mathbf{n} - \nabla_s\sigma$, where σ is the surface tension coefficient. In this work, σ is taken as a constant for simplicity.

It is noted that Eq. (3.2) together with the Dirichlet boundary condition Eq. (3.3) leads to the compatibility condition that $\boldsymbol{u}_b$ must satisfy:

$$\int_{\partial\Omega} \boldsymbol{u}_{\mathrm{b}}\cdot\mathbf{n}_{\mathrm{b}} dS = 0, \tag{3.5}$$

where $\mathbf{n}_{\mathrm{b}}$ is the outer unit normal to $\partial\Omega$.

4 Jump conditions across the interface

In order to apply the simplified immersed interface method as described in Section 2 to Stokes equations and achieve second order spatial accuracy in the solution, the jumps in the velocity $\mathbf{u}$ and pressure p and up to the second normal derivative across the interface need to be derived. Suppose $\boldsymbol{\eta}$ and $\boldsymbol{\tau}$ are two orthogonal unit tangential directions of the interface Γ. The solution and its first normal derivative jumps of the pressure and velocity derived in [6,12,37,41] can be rewritten as

$$[p] = \boldsymbol{f}\cdot\mathbf{n}, \quad \left[\frac{\partial p}{\partial \mathbf{n}}\right] = \frac{\partial(\boldsymbol{f}\cdot\boldsymbol{\eta})}{\partial\boldsymbol{\eta}} + \frac{\partial(\boldsymbol{f}\cdot\boldsymbol{\tau})}{\partial\boldsymbol{\tau}} + [\boldsymbol{G}]\cdot\mathbf{n}, \tag{4.1a}$$

$$[\boldsymbol{u}] = 0, \quad \left[\frac{\partial \boldsymbol{u}}{\partial \mathbf{n}}\right] = \frac{1}{\mu}\Big((\boldsymbol{f}\cdot\mathbf{n})\mathbf{n} - \boldsymbol{f}\Big). \tag{4.1b}$$

Note the force $\boldsymbol{f}$ can be decomposed to its normal $\mathbf{n}$ and two tangential directions $\boldsymbol{\eta}$ and $\boldsymbol{\tau}$ as

$$\boldsymbol{f} = (\boldsymbol{f}\cdot\mathbf{n})\mathbf{n} + (\boldsymbol{f}\cdot\boldsymbol{\eta})\boldsymbol{\eta} + (\boldsymbol{f}\cdot\boldsymbol{\tau})\boldsymbol{\tau}. \tag{4.2}$$

Thus the first normal derivative jump of $\boldsymbol{u}$ in (4.1) can be further rewritten as

$$\left[\frac{\partial \boldsymbol{u}}{\partial \mathbf{n}}\right] = -\frac{1}{\mu}\Big((\boldsymbol{f}\cdot\boldsymbol{\eta})\boldsymbol{\eta} + (\boldsymbol{f}\cdot\boldsymbol{\tau})\boldsymbol{\tau}\Big). \tag{4.3}$$

By taking the divergence on both sides of Eq. (3.1) and using the incompressibility constraint of Eq. (3.2), it is easily known that the pressure satisfies $\Delta p = \nabla\cdot\boldsymbol{G}$. Thus the jump in the second normal derivative of pressure can be derived as

$$\left[\frac{\partial^2 p}{\partial \mathbf{n}^2}\right] = [\nabla\cdot\boldsymbol{G}] - \kappa\left[\frac{\partial p}{\partial \mathbf{n}}\right] - \nabla_s^2[p]. \tag{4.4}$$

From the velocity equation $\Delta \boldsymbol{u} = \frac{1}{\mu}(\nabla p - \boldsymbol{G})$, the jump in the second normal derivative of the velocity can be derived as

$$\left[\frac{\partial^2 \boldsymbol{u}}{\partial \mathbf{n}^2}\right] = \frac{1}{\mu}\Big([\nabla p] - [\boldsymbol{G}]\Big) - \kappa\left[\frac{\partial \boldsymbol{u}}{\partial \mathbf{n}}\right] - \nabla_s^2[\boldsymbol{u}], \tag{4.5}$$

where the pressure gradient jump $[\nabla p] = \left[\frac{\partial p}{\partial \mathbf{n}}\right]\mathbf{n} + \left[\frac{\partial p}{\partial \tau}\right]\boldsymbol{\tau} + \left[\frac{\partial p}{\partial \eta}\right]\boldsymbol{\eta} = \left[\frac{\partial p}{\partial \mathbf{n}}\right]\mathbf{n} + \frac{\partial [p]}{\partial \tau}\boldsymbol{\tau} + \frac{\partial [p]}{\partial \eta}\boldsymbol{\eta}$. The above surface derivatives can be computed using the least squares interpolation as in [6].

5 Numerical methods

In this work, a staggered grid with a uniform mesh width $h = \Delta x = \Delta y = \Delta z$ is used for the fluid variables as shown in Fig. 3, which shows one cell of the staggered grid. With the staggered grid, the pressure point $p_{i,j,k}$ is at the center of the cell, and the x-component of the velocity filed (i.e., $u_{i,j,k}$) is at the center of the front-back side of the cell, the y-component (i.e., $v_{i,j,k}$) is at the center of the left-right side of the cell, and the z-component (i.e., $w_{i,j,k}$) is at the center of the top-bottom side of the cell. Let introduce the following notations

$$u_{i,j,k} = u\left(x_i, y_j + \frac{1}{2}h, z_k + \frac{1}{2}h\right), \quad i = 1,\cdots,N_x+1, \quad j = 1,\cdots,N_y, \quad k = 1,\cdots,N_z, \tag{5.1a}$$

$$v_{i,j,k} = v\left(x_i + \frac{1}{2}h, y_j, z_k + \frac{1}{2}h\right), \quad i = 1,\cdots,N_x, \quad j = 1,\cdots,N_y+1, \quad k = 1,\cdots,N_z, \tag{5.1b}$$

$$w_{i,j,k} = w\left(x_i + \frac{1}{2}h, y_j + \frac{1}{2}h, z_k\right), \quad i = 1,\cdots,N_x, \quad j = 1,\cdots,N_y, \quad k = 1,\cdots,N_z+1, \tag{5.1c}$$

$$p_{i,j,k} = p\left(x_i + \frac{1}{2}h, y_j + \frac{1}{2}h, z_k + \frac{1}{2}h\right), \quad i = 1,\cdots,N_x, \quad j = 1,\cdots,N_y, \quad k = 1,\cdots,N_z. \tag{5.1d}$$

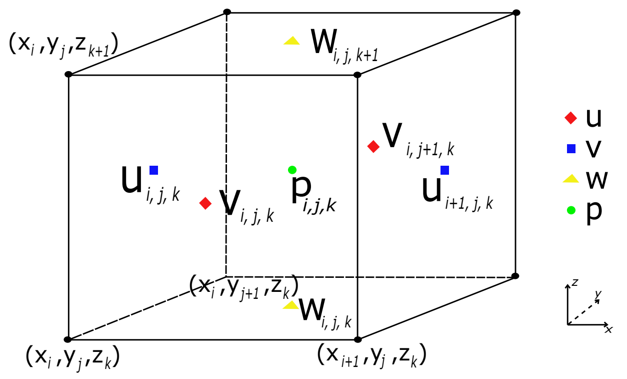


Figure 3: This figure shows the model of staggered grid. Four sets of grids for u,v,w and p are used. The position of the subscripts is also shown.

5.1 A simple 3D IIM Stokes solver

The discretization of the Stokes equation by second order MAC finite difference scheme leads to the following discrete scheme

$$-\Big(\mu\Delta_h \boldsymbol{u}_{i,j,k}+\mathcal{C}\{\mu\Delta\boldsymbol{u}_{i,j,k}\}\Big)+G^{\text{MAC}}p_{i,j,k}+\mathcal{C}\{\nabla_h p_{i,j,k}\}=\boldsymbol{G}_{i,j,k}, \tag{5.2}$$

$$D^{\text{MAC}}\boldsymbol{u}_{i,j,k}+\mathcal{C}\{\nabla_h\cdot\boldsymbol{u}_{i,j,k}\}=0, \tag{5.3}$$

with the approximated discrete boundary condition of the velocity treated in Eq. (3.3). Here Δ_h is the standard central difference operator, and G^{MAC} and D^{MAC} are the MAC gradient and the divergence operators, respectively, i.e.,

$$\Delta_h\boldsymbol{u}_{i,j,k}=\frac{\boldsymbol{u}_{i-1,j,k}+\boldsymbol{u}_{i+1,j,k}+\boldsymbol{u}_{i,j-1,k}+\boldsymbol{u}_{i,j+1,k}+\boldsymbol{u}_{i,j,k-1}+\boldsymbol{u}_{i,j,k+1}-6\boldsymbol{u}_{i,j,k}}{h^2}, \tag{5.4}$$

$$G^{\text{MAC}}p_{i,j,k}=\Big(\frac{p_{i,j,k}-p_{i-1,j,k}}{h},\frac{p_{i,j,k}-p_{i,j-1,k}}{h},\frac{p_{i,j,k}-p_{i,j,k-1}}{h}\Big), \tag{5.5}$$

$$D^{\text{MAC}}\boldsymbol{u}_{i,j,k}=\frac{u_{i+1,j,k}-u_{i-1,j,k}}{h}+\frac{v_{i,j+1,k}-v_{i,j-1,k}}{h}+\frac{w_{i,j,k+1}-w_{i,j,k-1}}{h}. \tag{5.6}$$

The above $\mathcal{C}\{\mu\Delta\boldsymbol{u}_{i,j,k}\}$, $\mathcal{C}\{\nabla_h p_{i,j,k}\}$) and $\mathcal{C}\{\nabla_h\cdot\boldsymbol{u}_{i,j,k}\}$ are the corresponding spatial correction terms to improve the accuracy of the local finite difference approximations, which are added to the finite difference equations and only non-zero at those irregular grid points. These corrections will be evaluated in Section 5.2.

Denoting $\boldsymbol{\mathcal{C}}^1_{i,j,k}=\mathcal{C}\{\Delta\boldsymbol{u}_{i,j,k}\}-\mathcal{C}\{\nabla p_{i,j,k}\}$ and $\mathcal{C}^2_{i,j,k}=-\mathcal{C}\{\nabla\cdot\boldsymbol{u}_{i,j,k}\}$, the system (5.2)-(5.3) can be rewritten as

$$-\mu\Delta_h\boldsymbol{u}_{i,j,k}+G^{\text{MAC}}p_{i,j,k}=\boldsymbol{G}_{i,j,k}+\boldsymbol{\mathcal{C}}^1_{i,j,k}, \tag{5.7}$$

$$D^{\text{MAC}}\boldsymbol{u}_{i,j,k}=\hat{\mathcal{C}}^2_{i,j,k}. \tag{5.8}$$

Here $\hat{\mathcal{C}}^2_{i,j,k}=\mathcal{C}^2_{i,j,k}-\bar{\mathcal{C}}^2_{i,j,k}$, where $\bar{\mathcal{C}}^2_{i,j,k}=\Sigma\mathcal{C}^2_{i,j,k}/M_p$, and M_p is the number of the p grid points in the computation. Note that the term $\mathcal{C}^2_{i,j,k}$ is perturbed to $\hat{\mathcal{C}}^2_{i,j,k}$ in Eq. (5.8), so as to satisfy the discrete compatibility condition in Eq. (3.5) and thereby ensure the solvability of the system (5.2)-(5.3). However, the order of the local truncation errors after the perturbation has not been changed yet as discussed in [17,31].

Denoting $\mathbf{B}_1$ and B_2 as the corresponding terms on the right hand side of Eqs. (5.7)-(5.8) in vector form, respectively, then the system can be written in the matrix-vector form as

$$\begin{pmatrix} -\mu\Delta_h & G^{\text{MAC}} \\ D^{\text{MAC}} & 0 \end{pmatrix}\begin{pmatrix} \boldsymbol{u} \\ p \end{pmatrix}=\begin{pmatrix} \mathbf{B}_1 \\ B_2 \end{pmatrix}. \tag{5.9}$$

The above system is solved by the CG-Uzawa type method as in [32]. The Uzawa procedure consists of two steps, i.e.,

Step 1. Solve $D^{\text{MAC}}(\mu\Delta_h)^{-1}G^{\text{MAC}}p=B_2+D^{\text{MAC}}(\mu\Delta_h)^{-1}\mathbf{B}_1$ for the pressure p;

Step 2. Solve $\mu\Delta_h\boldsymbol{u}=\mathbf{B}_1-G^{\text{MAC}}p$ for the velocity $\boldsymbol{u}$.

In **Step 1**, the pressure is solved by the conjugate gradient method. During the iteration, only the matrix-vector product of $D^{\text{MAC}}\mu\Delta_h^{-1}G^{\text{MAC}}$ and p need to be computed. The required inverse of $\mu\Delta_h$ corresponds to solving a Poisson equation, which can be solved by some fast methods, such as the FFT method, multigrid method, and so on. In this work, the fast solvers from FISHPACK [28] are used. Once the pressure is obtained, the velocity $\mathbf{u}$ can be solved by the fast solvers from FISHPACK [28] again via **Step 2**. The computational complexity for the fast Poisson solver from FISHPACK is $\mathcal{O}$(M log(M)), where M is the number of interior grid points of the computational domain. The present CG method converges fast as discussed in [32] and the number of iterations in the CG method is small and almost independent of the mesh size.

5.2 Calculation of correction terms

The calculation of correction term $\mathcal{C}\{\mu\Delta\mathbf{u}_{i,j,k}\}$ is the same as described in Section 2. The remaining question is how to compute the correction terms $\mathcal{C}\{\nabla_h p_{i,j,k}\})$ and $\mathcal{C}\{\nabla_h\cdot\mathbf{u}_{i,j,k}\}$. Without loss of generality, let us only consider the difference scheme of $\frac{\partial p}{\partial x}$ at the location of u-grid point here, the approximations of the derivatives $\frac{\partial p}{\partial y}$ and $\frac{\partial p}{\partial z}$ at the locations of v-grid points and w-grid points, respectively, can be made in a similar way.

Note that the location of $u_{i,j,k}$ is the midpoint of the locations of $p_{i,j,k}$ and $p_{i-1,j,k}$. A correction term needs to be added when the p-grid points cross the interface. Suppose the locations of $u_{i,j,k}$ and $p_{i,j,k}$ are inside the interface, while the location of $p_{i,j,k}$ is outside the interface, as illustrated in Fig. 4(a), then the difference $\frac{\partial p}{\partial x}$ at the location of $u_{i,j,k}$ can

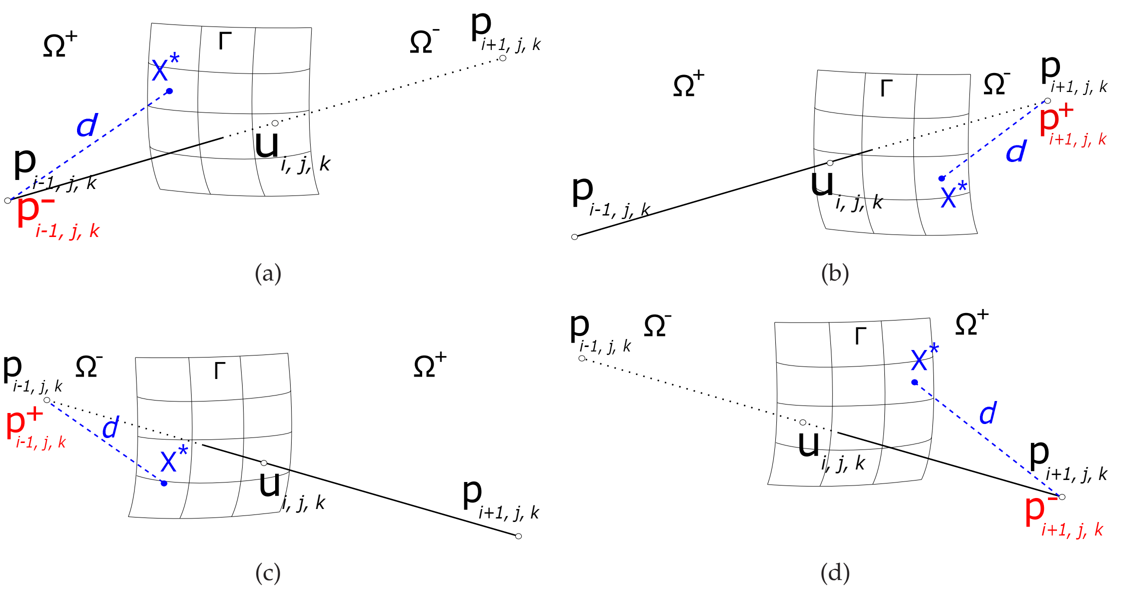


Figure 4: Four different cases for the relative positions of the u and p along the p-grid line where the interface crosses.

be approximated as

$$\begin{aligned}\left(\frac{\partial p}{\partial x}\right)_{i,j,k}&=\frac{p_{i,j,k}-p_{i-1,j,k}}{h}=\frac{p^-_{i,j,k}-p^+_{i-1,j,k}}{h}=\frac{p^-_{i,j,k}-p^-_{i-1,j,k}}{h}-\frac{p^+_{i-1,j,k}-p^-_{i-1,j,k}}{h}\\&=p_x(\boldsymbol{x}_{i+\frac{1}{2},j+\frac{1}{2},k+\frac{1}{2}})+\mathcal{O}\left(h^2\right)-\frac{p^c_{i-1,j,k}}{h}\\&=p_x(\boldsymbol{x}_{i+\frac{1}{2},j+\frac{1}{2},k+\frac{1}{2}})-\mathcal{C}\{p_x\}_{i,j,k}+\mathcal{O}\left(h^2\right).\end{aligned}\tag{5.10}$$

Here $\mathcal{C}\{p_x\}_{i,j,k}$ is the corresponding spational correction term, which can be computed by

$$\begin{aligned}\mathcal{C}\{p_x\}_{i,j,k}=p^c_{i-1,j,k}&=p^+_{i-1,j,k}-p^-_{i-1,j,k}\\&=\left(p^+_*+d\frac{\partial p^+_*}{\partial\mathbf{n}}+\frac{d^2}{2}\frac{\partial^2 p^+_*}{\partial\mathbf{n}^2}+\mathcal{O}\left(h^3\right)\right)-\left(p^-_*+d\frac{\partial p^-_*}{\partial\mathbf{n}}+\frac{d^2}{2}\frac{\partial^2 p^-_*}{\partial\mathbf{n}^2}+\mathcal{O}\left(h^3\right)\right)\\&=[p]_{\mathbf{X}_*}+d\left[\frac{\partial p}{\partial\mathbf{n}}\right]_{\mathbf{X}_*}+\frac{d^2}{2}\left[\frac{\partial^2 p}{\partial\mathbf{n}^2}\right]_{\mathbf{X}_*}+\mathcal{O}\left(h^3\right),\end{aligned}\tag{5.11}$$

where $\mathbf{X}_*$ is the orthogonal projection of $\boldsymbol{x}_{i-\frac{1}{2},j+\frac{1}{2},k+\frac{1}{2}}$ on the interface, and d is the signed distance from $\boldsymbol{x}_{i-\frac{1}{2},j+\frac{1}{2},k+\frac{1}{2}}$ to $\mathbf{X}_*$. Notice that if the $\boldsymbol{x}_{i-\frac{1}{2},j+\frac{1}{2},k+\frac{1}{2}}$ is outside of the interface, then d must be positive. For the other cases of Fig. 4, the correction terms can be evaluated in the same way.

Similar to $\mathcal{C}\{\nabla_h p_{i,j,k}\})$, the correction term $\mathcal{C}\{\nabla_h\cdot\boldsymbol{u}_{i,j,k}\}$ can be computed.

5.3 Level set method for moving interface

The interface is implicitly represented by the zero level of a level set function φ as in [20, 37], i.e., $\Gamma(t)=\{\boldsymbol{x}:\ \varphi(\boldsymbol{x},t)=0\}$. Geometrical quantities can be easily computed using the level set function. Assuming that $\{\boldsymbol{x}:\varphi(\boldsymbol{x},t)<0\}=\Omega^-$, the outward normal vector and curvature of the interface Γ are derived as

$$\mathbf{n}=\frac{\nabla\varphi}{|\nabla\varphi|},\qquad \kappa=\nabla\cdot\mathbf{n}=-\frac{\nabla\varphi\mathrm{He}(\varphi)\nabla\varphi^T-|\nabla\varphi|Trace(\mathrm{He})}{|\nabla\varphi|^3}. \tag{5.12}$$

The level set function is evolved according to the following linear advection equation

$$\varphi_t+\boldsymbol{u}\cdot\nabla\varphi=0. \tag{5.13}$$

At every time step, the level set function φ is re-initialized to be a distance function by solving the following Hamilton-Jacobi equation as in [20, 37]

$$\begin{cases}\varphi_\tau+\mathrm{sign}(\varphi_0)(|\nabla\varphi|-1)=0,\\ \varphi(\boldsymbol{x},0)=\varphi_0(\boldsymbol{x}),\end{cases} \tag{5.14}$$

where φ_0 is the level set function before the re-initialization, and τ is the pseudo-time. The fifth order WENO scheme [25] for the spatial derivatives and TVD RK [9] scheme for the time discretizations are employed to solve the level set equation and its re-initialization process, respectively.

5.4 Numerical implementation

In this section, the simple implementation of the proposed algorithm is described for the moving interface. For given $\boldsymbol{u}_h^n$, p_h^n, and φ_h^n, the algorithm for finding $\boldsymbol{u}_h^{n+1}$, p_h^{n+1}, and φ_h^{n+1} can be summarized as follows:

Step A: Compute $\boldsymbol{u}_h^{n+1}$ and p_h^{n+1} using the 3D IIM Stokes solver as described in Section 5.1. This step involves computing the appropriate correction terms as described in Section 5.2.

- Compute the right hand side of the pressure equation in **Step 1** of Section 5.1.
- Compute the pressure p_h^{n+1} using the CG method.
- Compute the velocity $\boldsymbol{u}_h^{n+1}$ using the fast Poisson solver in **Step 2** of Section 5.1.

Step B: Based on the velocity $\boldsymbol{u}_h^{n+1}$, move the interface by solving the level set equation, and then re-initialize the level set function to get φ_h^{n+1}.

6 Numerical results

In this section, numerical experiments will be carried out to demonstrate the accuracy and the effectiveness of our algorithm proposed in this work. Throughout this section, the computational domain is $\Omega=[-1,1]^3$, the viscosity $\mu=1$ and the stop tolerance of 10^{-8} for the CG method are taken in all simulations unless it is stated otherwise. All the simulations are done on a Laptop with 2.00GHz processor.

Example 6.1. The first example of Poisson solver.

In this example, a numerical test is performed to check the accuracy of the 3D IIM Poisson solver. The interface is a spherical surface with radius $r=0.5$. So the level-set function can be taken as $\varphi=\frac{x^2}{0.25}+\frac{y^2}{0.25}+\frac{z^2}{0.25}-1$. The exact solution is given by

$$u(\boldsymbol{x})=\begin{cases}\cos(x)\sin(y)\sin(z), & \boldsymbol{x}\in\Omega^+,\\ 0, & \boldsymbol{x}\in\Omega^-.\end{cases}$$

The right hand side term g, the boundary condition, and jumps in solution and its normal derivative can be determined from the exact solution.

A mesh refinement analysis in maximum norm is shown in Table 1, which indicates that second order accuracy of the solution is achieved. In the table, the variables with subscript *iter* represent the corresponding quantities computed by geometric iteration method. Denoting $\mathcal{C}_{exact}=\Delta_h u_{exact}-g$, the error between the computed correction term and $\mathcal{C}_{exact}$ is listed in the last column, which demonstrates that the local truncation error at an irregular grid point is $\mathcal{O}(h)$ as expected. Since the discrete partial derivatives are approximated exactly for this choice of level set function, the error of the orthogonal projection point referring to the exact orthogonal projection point reaches the machine precision as seen from the table. The computation is also performed by applying the approximate orthogonal projection method in [20]. It is found that the approximate orthogonal projection method produces the same results as the present iteration method due to the special spherical interface considered, which are not shown here.

Table 1: Grid refinement analysis in Example 6.1 with iteration method.

Mesh size	$\|u_{iter}-u_h\|_\infty$	Order	$\|\mathbf{X}^*_{iter}-\mathbf{X}^*_{exact}\|_\infty$	Order	$\|\mathcal{C}_{iter}-\mathcal{C}_{exact}\|_\infty$
32	1.4614E-5	-	1.6653E-16	-	6.3223E-3
64	3.6659E-6	2.00	2.2204E-16	-	2.5959E-3
128	9.0972E-7	2.01	2.2204E-16	-	1.6931E-3
256	2.2817E-7	2.00	2.2204E-16	-	8.1339E-4

Example 6.2. The second example of Poisson solver.

In this example, the taken solution is same with the first example. The interface is an ellipsoid with semi-axes $(a,b,c)=(0.7,0.6,0.5)$. The level set function is taken as $\varphi=\frac{x^2}{0.7^2}+\frac{y^2}{0.6^2}+\frac{z^2}{0.5^2}-1$.

Table 2: Grid refinement analysis in Example 6.2 with approximate projection method, $\varphi=\frac{x^2}{0.7^2}+\frac{y^2}{0.6^2}+\frac{z^2}{0.5^2}-1$.

Mesh size	$\|u_{appr}-u_h\|_\infty$	Order	$\|\mathbf{X}^*_{appr}-\mathbf{X}^*_{exact}\|_\infty$	Order	$\|\mathcal{C}_{appr}-\mathcal{C}_{exact}\|_\infty$
32	1.0143E-4	-	1.2516E-3	-	0.0941
64	3.4188E-5	1.57	4.0373E-4	1.63	0.1194
128	1.0775E-5	1.67	9.9661E-5	2.02	0.1273
256	4.4516E-6	1.28	2.5171E-5	1.99	0.1247

Table 3: Grid refinement analysis in Example 6.2 with iteration method, $\varphi=\frac{x^2}{0.7^2}+\frac{y^2}{0.6^2}+\frac{z^2}{0.5^2}-1$.

Mesh size	$\|u_{iter}-u_h\|_\infty$	Order	$\|\mathbf{X}^*_{iter}-\mathbf{X}^*_{exact}\|_\infty$	Oder	$\|\mathcal{C}_{iter}-\mathcal{C}_{exact}\|_\infty$
32	1.3493E-5	-	1.5707E-13	-	7.2183E-3
64	3.4131E-6	1.98	1.5853E-13	-	3.6083E-3
128	8.2942E-7	2.04	1.6744E-13	-	1.7877E-3
256	2.0349E-7	2.03	1.6720E-13	-	8.9370E-4

The mesh refinement analyses in maximum norm with the approximate orthogonal projection method and present method are shown in Table 2 and Table 3, respectively. In these tables, the variables with subscript $appr$ represent the corresponding quantities computed by the approximate orthogonal projection method. It can be seen from Table 2, the accuracy of solution is decreased for the case of not a spherical interface. This is because that the approximate orthogonal projections referring to the exact orthogonal projections have only second-order accuracy not third-order accuracy for this case, which results in a larger error of correction term, therefore a loss of accuracy of the computed correction term affects the final overall accuracy of the solution.

Table 3 indicates that the present IIM with the geometric iteration method can achieve second order accuracy of the solution. The last column of the table lists the error between the computed correction term and $\mathcal{C}_{exact}$, which demonstrates that the local truncation error $\mathcal{O}(h)$ at an irregular point is obtained as expected. Again the error of the orthogonal projection point is roughly equal to the machine precision due to the discrete partial derivatives approximated exactly for this level set function, which is shown in the fourth column of the table.

For further comparisons of the choice of level set function, the computation is also done for the case that the level set function is taken as $\varphi=\sqrt{\frac{x^2}{0.7^2}+\frac{y^2}{0.6^2}+\frac{z^2}{0.5^2}}-1$. The results with the approximate orthogonal projection method and present method are obtained, which are shown in Table 4 and Table 5, respectively. These results can be compared to those in Table 2 and Table 3, respectively, and the similar results of second order accuracy of the solution with iteration method and reduced accuracy of the solution with approximate orthogonal projection method are obtained. It also can be seen from the fourth column of this table that the present iteration scheme for finding the orthogonal

Table 4: Grid refinement analysis in Example 6.2 with approximate projection method, $\varphi=\sqrt{\frac{x^2}{0.7^2}+\frac{y^2}{0.6^2}+\frac{z^2}{0.5^2}}-1$.

Mesh size	$\|u_{appr}-u_h\|_\infty$	Order	$\|\mathbf{X}^*_{appr}-\mathbf{X}^*_{exact}\|_\infty$	Order	$\|\mathcal{C}_{appr}-\mathcal{C}_{exact}\|_\infty$
32	1.2075E-4	-	1.3732E-3	-	0.0953
64	4.0865E-5	1.56	4.2297E-4	1.70	0.1124
128	1.3310E-5	1.62	1.0195E-4	2.05	0.1240
256	5.0332E-6	1.40	2.5454E-5	2.00	0.1231

Table 5: Grid refinement analysis in Example 6.2 with iteration method, $\varphi=\sqrt{\frac{x^2}{0.7^2}+\frac{y^2}{0.6^2}+\frac{z^2}{0.5^2}}-1$.

Mesh size	$\|u_{iter}-u_h\|_\infty$	Order	$\|\mathbf{X}^*_{iter}-\mathbf{X}^*_{exact}\|_\infty$	Order	$\|\mathcal{C}_{iter}-\mathcal{C}_{exact}\|_\infty$
32	7.5874E-5	-	1.9630E-4	-	2.0792E-2
64	2.0834E-5	1.86	2.7843E-5	2.82	1.4594E-2
128	4.9408E-6	2.08	3.2978E-6	3.08	8.0419E-2
256	1.1919E-6	2.05	4.2968E-7	2.94	3.9504E-3

projection point is indeed third-order convergent. Since the discrete partial derivatives are not approximated exactly for this choice of level set function, the error of the orthogonal projection point referring to the exact orthogonal projection point doesn't reach the machine precision yet as expected.

Example 6.3. The first example of Stokes solver with fixed interface.

In this example, the Stokes equations with fixed interface are considered. The interface is set as a ellipsoid with semi-axes $a=0.68$, $b=0.58$, and $c=0.48$. So the level-set function can be given as $\varphi=\frac{x^2}{0.68^2}+\frac{y^2}{0.58^2}+\frac{z^2}{0.48^2}-1$. The exact velocity is given as

$$\begin{cases} u=\cos(x)\sin(y)\sin(z), & v=\sin(x)\cos(y)\sin(z), & w=-2\sin(x)\sin(y)\cos(z), & \boldsymbol{x}\in\Omega^-, \\ u=0, & v=0, & w=0, & \boldsymbol{x}\in\Omega^+. \end{cases}$$

And the exact pressure is given as

$$p=\begin{cases} -3\sin(x)\sin(y)\sin(z), & \boldsymbol{x}\in\Omega^-, \\ 0, & \boldsymbol{x}\in\Omega^+. \end{cases}$$

The external force $\boldsymbol{G}$ is given as

$$\boldsymbol{G}=\begin{cases} (0,0,-9\sin(x)\sin(y)\cos(z)), & \boldsymbol{x}\in\Omega^-, \\ \mathbf{0}, & \boldsymbol{x}\in\Omega^+. \end{cases}$$

The interface force $\boldsymbol{f}$ and the jumps in solution and its normal derivative for the velocity and pressure can be derived from the exact solution. The homogeneous boundary condition for the velocity is taken for this example.

Table 6: A mesh refinement study for the 3D Stokes solver in Example 6.3.

Mesh size	$\|u-u_h\|_\infty$	Order	$\|v-v_h\|_\infty$	Order	$\|w-w_h\|_\infty$	Order	$\|p-p_h\|_\infty$	Order	N_{iter}
32	2.7480E-5	-	3.2613E-5	-	5.3369E-5	-	2.9694E-4	-	9
64	6.4257E-6	2.10	6.6792E-6	2.29	1.2908E-5	2.05	4.2235E-5	2.82	11
128	1.5390E-6	2.06	1.6040E-6	2.06	3.0546E-6	2.08	8.6126E-6	2.30	13
256	3.7149E-7	2.05	4.0101E-7	2.00	7.4811E-7	2.03	1.8803E-6	2.20	14

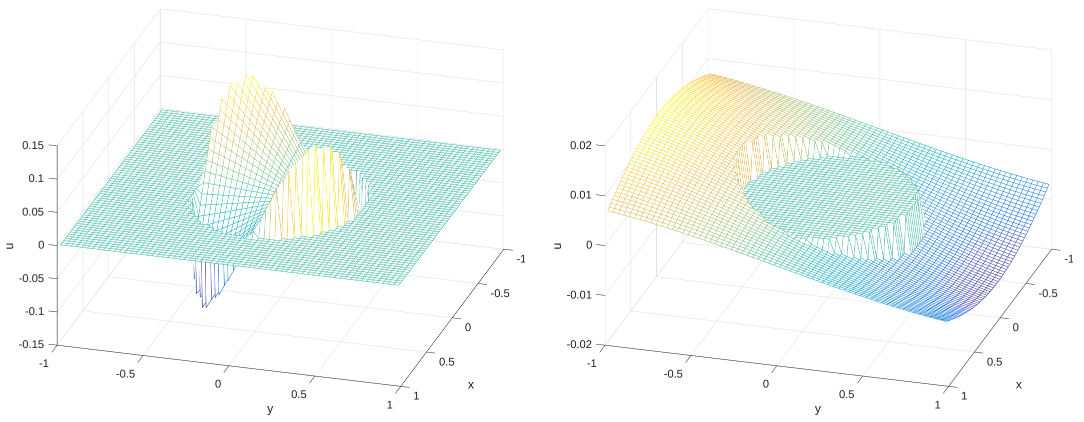


Figure 5: The solution of u at $z=\frac{h}{2}$ in Example 6.3 (left); solution of u at $z=\frac{h}{2}$ in Example 6.4 (right).

The result of the convergence rate analysis is shown in Table 6, which indicates that the velocity is second order accurate, and the pressure is also second order accurate. Table 6 also shows the number of the CG iterations in the Stokes solver, which indicates the number of iterations is almost independent of the mesh size. Fig. 5(left) shows the slice of the computed u on the cross section of the plane (i.e., $z=\frac{h}{2}$) on a $64\times64\times64$ grid. A clear jump in the velocity across the interface is observed.

Example 6.4. The second example of Stokes solver with fixed interface.

Compared to Example 6.3, the solution is set to be zero inside the interface for this example. The level-set function is written as $\varphi=\frac{x^2}{0.6^2}+\frac{y^2}{0.5^2}+\frac{z^2}{0.4^2}-1$. The exact expressions of the velocity and pressure are the same as Example 6.3 except for the value inside and outside exchanged. Then the surface force f and the jump conditions for solutions and their normal derivatives are reversed.

Fig. 5(right) shows the slice of the computed u on the cross section of the plane (i.e.,$z=\frac{h}{2}$) on a $64\times64\times64$ grid. It is observed that the discontinuity and the jump in the velocity are very sharply captured across the interface. The mesh refinement result is shown in Table 7, which also shows the number of the CG iterations in the Stokes solver. It indicates the number of iterations is almost independent of the mesh size again.

Table 7: A mesh refinement study for the 3D Stokes solver in Example 6.4.

Mesh size	$\|u-u_h\|_\infty$	Order	$\|v-v_h\|_\infty$	Order	$\|w-w_h\|_\infty$	Order	$\|p-p_h\|_\infty$	Order	N_{iter}
32	1.8664E-5	-	1.8642E-5	-	3.6901E-5	-	3.8501E-4	-	12
64	4.3675E-6	2.10	4.2952E-6	2.11	8.6202E-6	2.10	8.3160E-5	2.21	14
128	1.0452E-6	2.06	1.0229E-6	2.08	2.0860E-6	2.04	2.1392E-5	1.96	16
256	2.5794E-7	2.02	2.5206E-7	2.02	5.1300E-7	2.02	5.4489E-6	1.97	18

Example 6.5. Stokes equations with moving interface.

In this example, a moving interface problem which involves a pressurized bubble immersed in a quiescent Stokes flow is considered. The initial shape of the bubble is an ellipsoid with the semi-axes $a=0.75$, $b=0.5$, and $c=0.4$. The initial velocity and pressure are set zero, and the deformation of the bubble in the fluid is driven only by interface tension. The bubble is located at the center of the fluid domain. When the process starts, the bubble will relax to its spherical equilibrium shape with radius $r=\sqrt[3]{abc}\approx 0.5313$. The surface tension $\sigma=10$ is taken. The time step $\Delta t=h/5$ and a uniform grid of $64\times 64\times 64$ are employed. In the simulations, the velocity and pressure at time $t=0$ based on the initial ellipsoidal interface are first computed before the interface has moved.

Fig. 6(left) and Fig. 6(right) show a slice of the pressure and the u-component of the velocity field on the plane of $z=\frac{h}{2}$, respectively. As expected, it can be observed from this figure that the pressure is discontinuous and the pressure profile is very sharp across the interface, whereas the velocity is continuous across the interface. Fig. 7 shows this more clearly with the plots of cross section along the line $x=0$ or $y=0$ on the plane of $z=\frac{h}{2}$ for the pressure and the u-component of velocity field. The sharp jump in the pressure is well resolved as seen from these figures. Fig. 7 also shows that the pressure value along the major axis direction (x-axis direction) is larger than that along the short axis direction (y-axis direction), which is caused by the difference of the curvature.

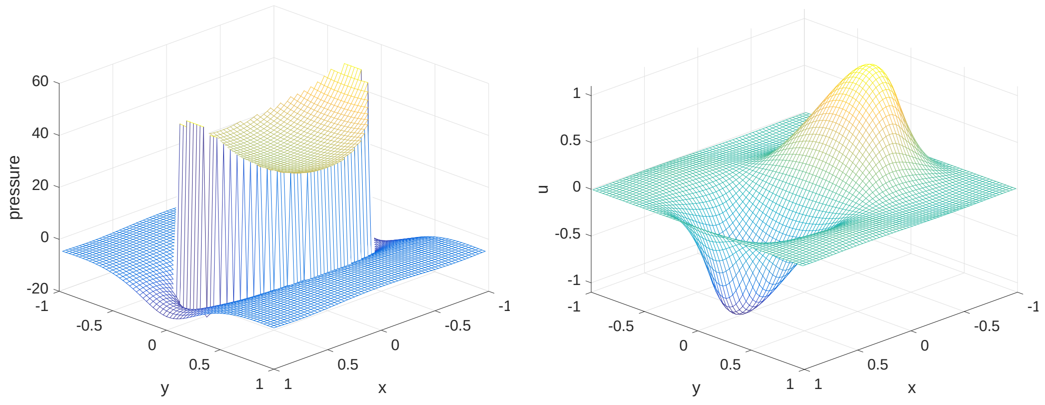


Figure 6: The pressure profile (left) and the component of velocity u (right) at the section of $z=\frac{h}{2}$.

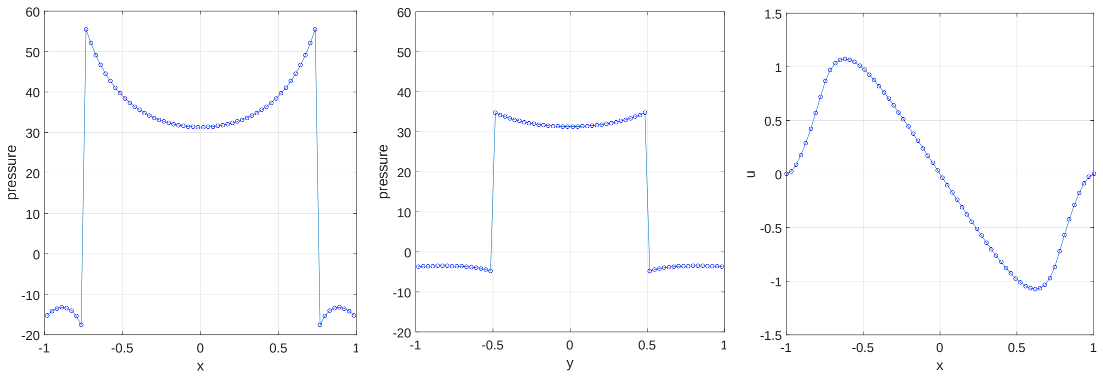


Figure 7: The pressure distributions along x-axis (left) and the y-axis (middle), respectively. The u-component of velocity field alone the x-axis (right).

Table 8: A mesh refinement study for moving interface in Example 6.5.

$L\times M\times N$	$\|u-u_h\|_\infty$	Order	$\|v-v_h\|_\infty$	Order	$\|w-w_h\|_\infty$	Order	$\|p-p_h\|_\infty$	Order
32	1.7774E-2	-	4.8890E-3	-	8.3265E-3	-	4.8234E-1	-
64	5.0507E-3	1.82	1.3127E-3	1.90	2.3483E-3	1.83	1.3452E-1	1.84
128	1.2926E-3	1.97	3.2698E-4	2.01	5.9147E-4	1.99	3.5392E-2	1.92

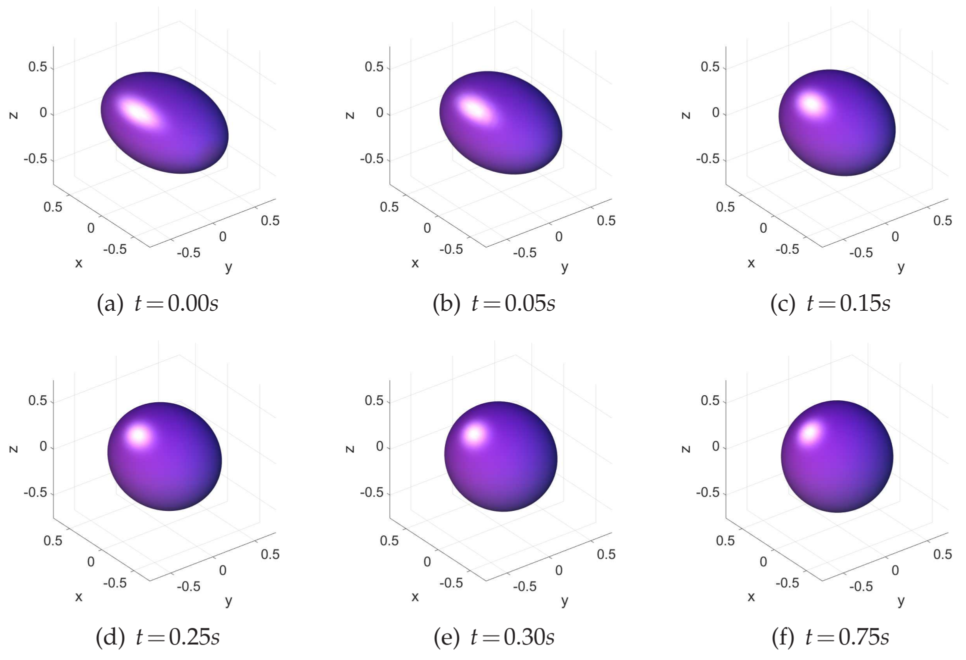
In Table 8, a mesh refinement analysis is shown, and the expected second order accuracy for the velocity is achieved. Since the analytic solution is not available, the errors in the velocity are measured via comparing with a fine grid solution. The computed solution on a $512\times512\times512$ grid is taken as the reference solution for comparison.

Then the simulation of moving interface is performed. Fig. 8 shows the deformed shape of the bubble at different times, which shows that the bubble is relaxing gradually to the spherical equilibrium shape. The corresponding snapshots of the bubble shapes in cross-sectional visualization of $z=0$ are presented in Fig. 9. Obviously, the long axis is changing faster than the short axis.

In Fig. 10(left), the volume conservation of the interface in the relaxation process is checked and found to be well conserved. In the figure, the maximum volume error is $0.91\ e-3$, which indicates only a small volume loss of 0.14 % in percentage.

Fig. 10(right) gives the maximum error of discrete divergence of velocity field versus time, which shows that the error is very small and reaches an accuracy of $9.98\ e-9$. It indicates that the discrete divergence-free condition is kept very well for the present method.

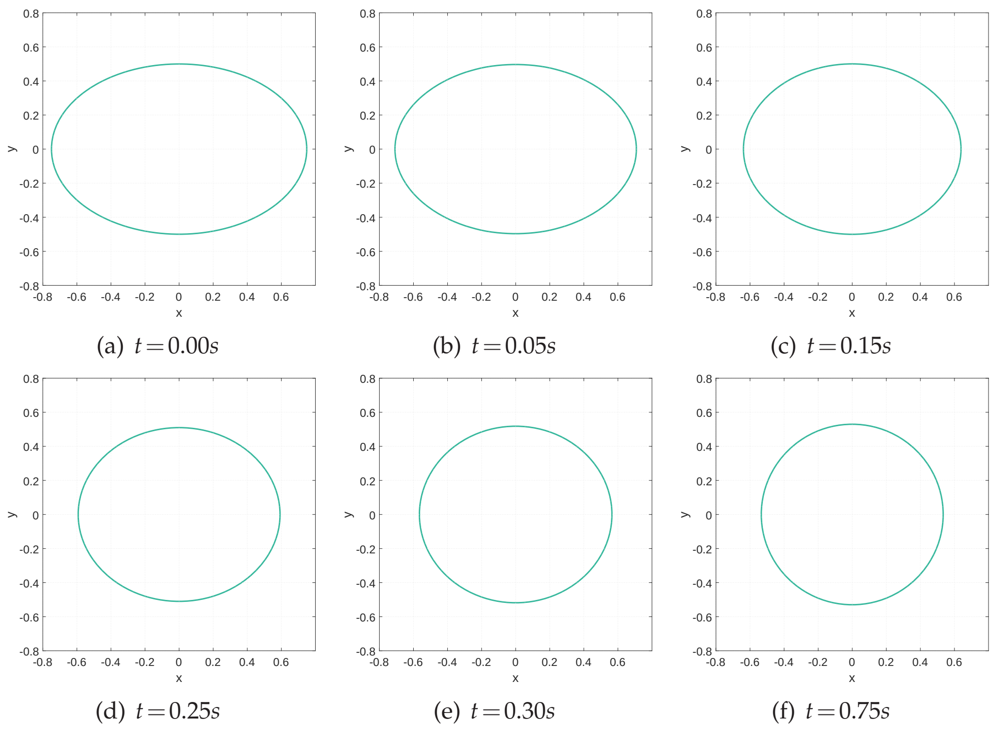
The evolution of three semi-axes versus time is shown in Fig. 11(left). It can been observed from this figure that the ellipsoidal interface relaxes gradually to its spherical equilibrium shape without almost no oscillations during the relaxation process. For the purpose of the comparison of surface tension effects on interface motion over longer times, the evolution of semi-major axis with different surface tension coefficient σ is pre-



(a) $t=0.00s$ (b) $t=0.05s$ (c) $t=0.15s$

(d) $t=0.25s$ (e) $t=0.30s$ (f) $t=0.75s$

Figure 8: The evolution of the bubble shapes at different time.



(a) $t=0.00s$ (b) $t=0.05s$ (c) $t=0.15s$

(d) $t=0.25s$ (e) $t=0.30s$ (f) $t=0.75s$

Figure 9: Snapshots of the bubble shapes in cross-sectional visualization of $z=0$.

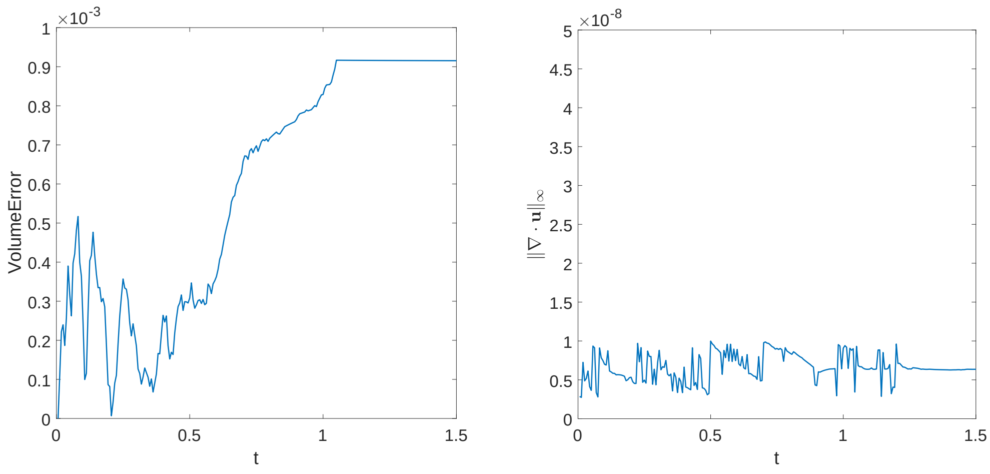


Figure 10: The absolute error in volume versus time (left), and the maximum absolute value of the numerical divergence of velocity field versus time (right).

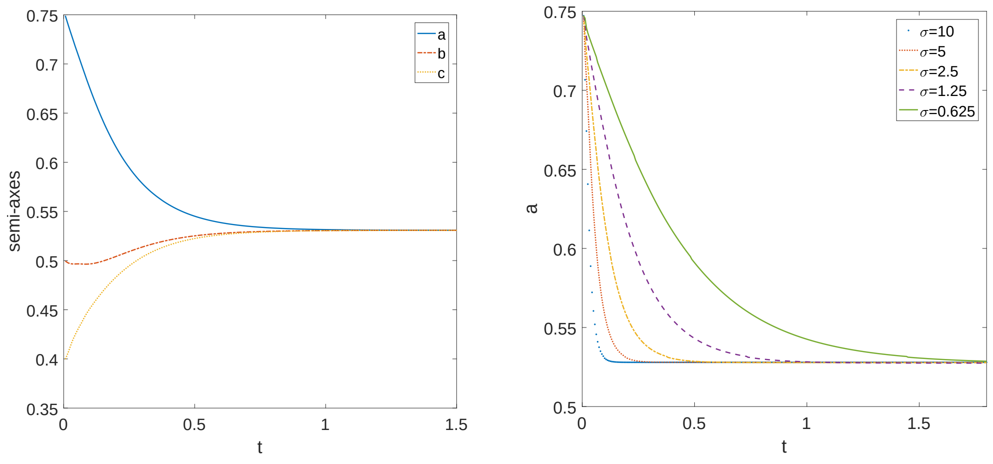


Figure 11: The evolutions of three semi-axes versus time (left) and the evolution of semi-major axis under different surface tension coefficients σ (right).

sented in Fig. 11(right). It indicates, with larger surface tension coefficient σ, the interface takes a relatively fewer time to relax to the equilibrium state.

Example 6.6. Stokes equations with complicated moving interface.

This example shows that the present method can handle more complicated interface. Let us consider a curve written as $r=0.5+0.1\sin 7\theta$, $\theta\in[0,\pi]$ in polar coordinates. By rotating it around the y-axis, a special surface is obtained. Fig. 12 shows the initial shape of the surface. In the simulation, a $64\times 64\times 64$ uniform grid is employed. The taken time step is $\Delta t=0.2h$. As expected, the interface will gradually relax into a sphere. This example mainly focuses on the deformation of the interface and not presents the more

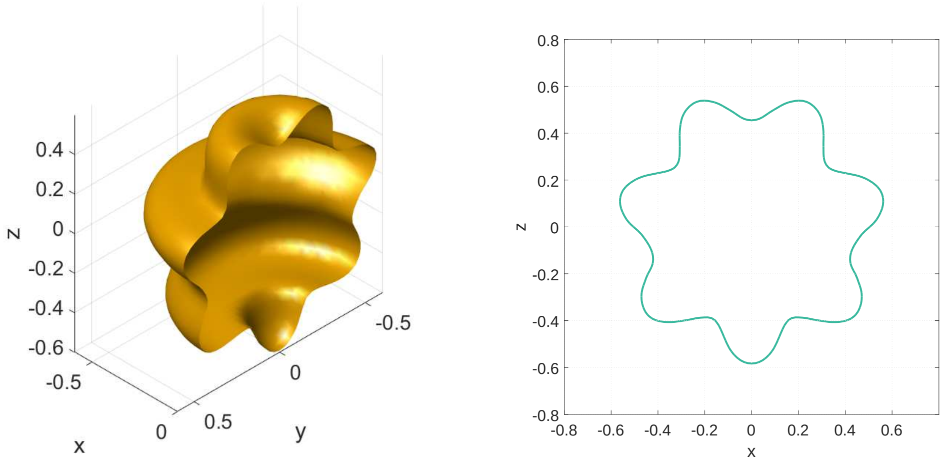
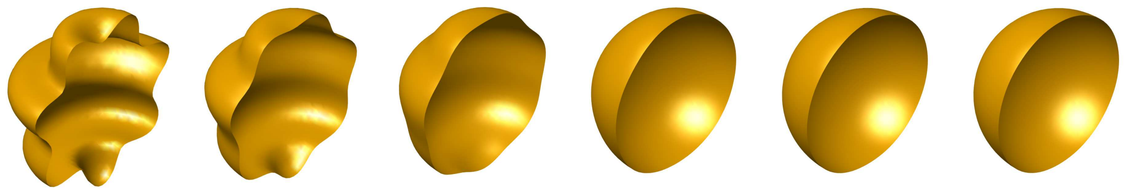
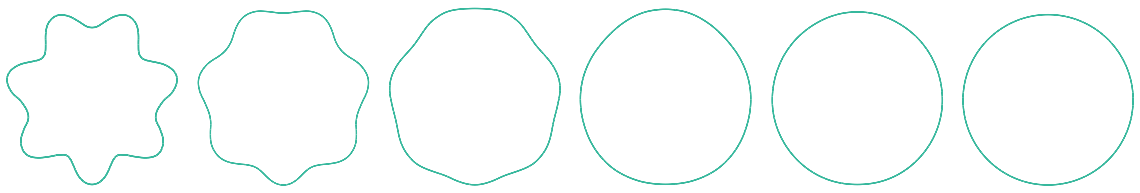


Figure 12: The initial shape of the complicated interface in Example 6.5.



(a) $t=0.00s$ (b) $t=0.02s$ (c) $t=0.05s$ (d) $t=0.10s$ (e) $t=0.20s$ (f) $t=0.50s$

Figure 13: The evolution of the interface at different time.



(a) $t=0.00s$ (b) $t=0.02s$ (c) $t=0.05s$ (d) $t=0.10s$ (e) $t=0.20s$ (f) $t=0.50s$

Figure 14: Snapshots of the interface in cross-sectional visualization of $y=0$.

detail numerical results yet. Fig. 13 presents the evolution of the interface at different time. The corresponding snapshots of the interface shapes in cross-sectional visualization of $y=0$ are presented in Fig. 14.

In Fig. 15(left), the volume conservation of the complicated interface in the relaxation process is checked and found to be well conserved. In the figure, the maximum volume error is 0.81 $e-3$, which indicates only a small volume loss of 0.15 % in percentage. And the Table 9 shows the mesh refinement analysis, and the expected second order accuracy for the velocity are achieved. It is noted that the order for both velocity and pressure

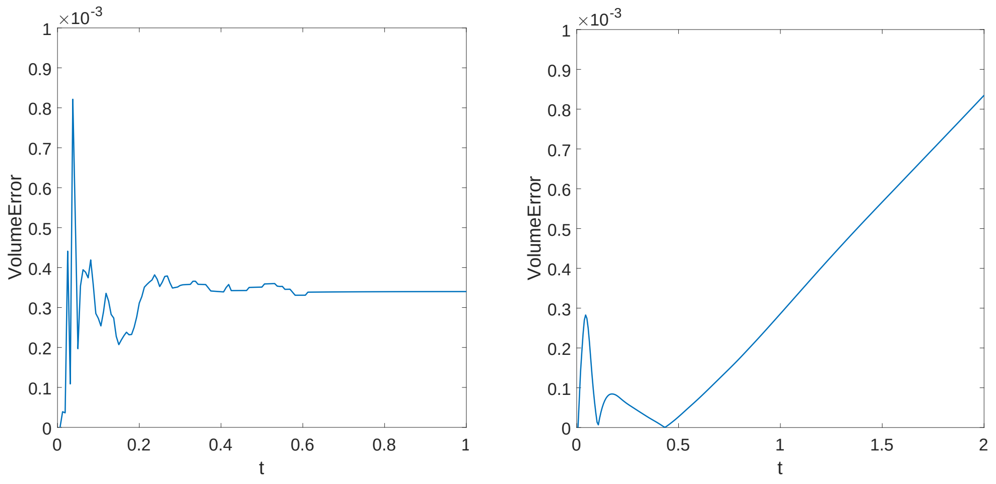


Figure 15: The absolute error in volume versus time in Example 6.6 (left), and in Example 6.7 (right) with $\sigma=5$, $\gamma=1$.

Table 9: A mesh refinement study for moving interface in Example 6.6.

$L\times M\times N$	$\|u-u_h\|_\infty$	Order	$\|v-v_h\|_\infty$	Order	$\|w-w_h\|_\infty$	Order	$\|p-p_h\|_\infty$	Order
32	1.2052E-0	-	1.2052E-0	-	1.6028E-0	-	2.7852E+1	-
64	4.8082E-1	1.33	4.8082E-1	1.33	4.6113E-1	1.80	2.9414E+1	-
128	1.1992E-1	2.00	1.1992E-1	2.00	1.0506E-1	2.13	9.1448E-0	1.69

field in the first stage of the refinement analysis are reduced for this example, which is probably caused due to the accuracy of the curvature. Because the curvature at some points on the complicated interface may become very large, the mesh size $32\times32\times32$ is too rough for the complicated interface. The computed solution on a $512\times512\times512$ grid is taken as the reference solution for comparison again.

Example 6.7. Stokes equations with moving interface in shear flow.

As a last example, the deformation of a single drop in a shear flow is considered. The velocity of shear flow without drop is given by the velocity $\boldsymbol{u}_\infty=(\gamma y,0,0)$, where γ is the shear rate. The different shear rate γ and the surface tension coefficient σ are taken in the simulations. The computational domain is $\Omega=[-2,2]\times[-1,1]\times[-1,1]$, and the initial shape of the drop is represented by a level-set function $\varphi=\sqrt{x^2+y^2+z^2}-0.5$. The deformation of the drop is described by the Taylor shape parameter $D(t)=(L(t)-B(t))/(L(t)+B(t))$, where $L(t)$ is the longest axis of the body and $B(t)$ is the shortest axis of the body.

In Fig. 15(right), the volume conservation of the shear flow in the deformation process is checked and found to be well conserved. In the figure, the maximum volume error is $0.91\,e{-3}$, which indicates only a small volume loss of 0.17 % in percentage. And the Table

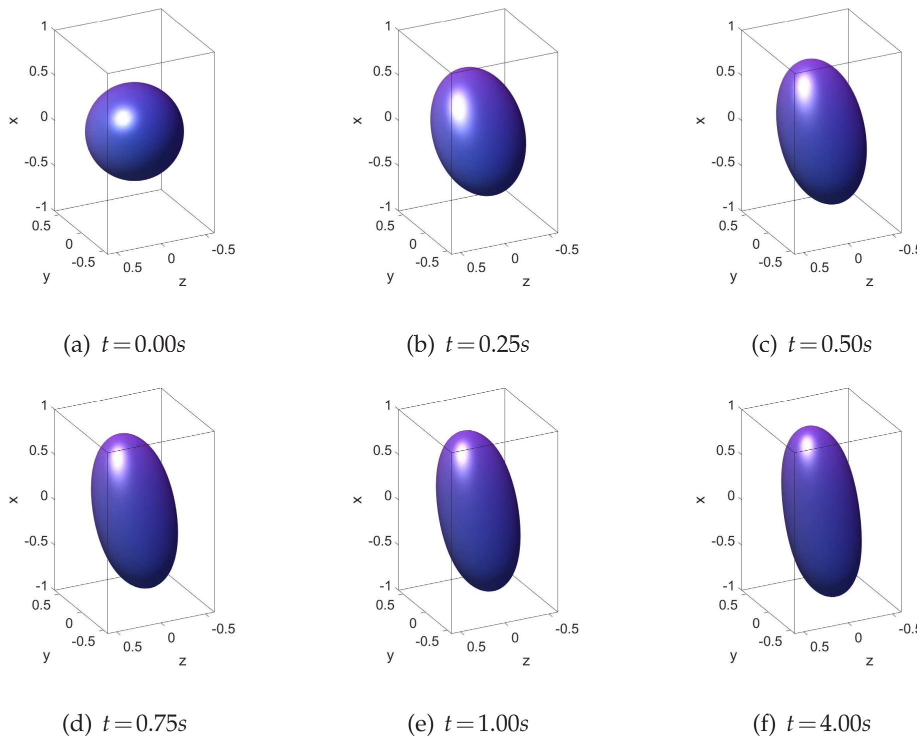

(a) $t=0.00s$ (b) $t=0.25s$ (c) $t=0.50s$ (d) $t=0.75s$ (e) $t=1.00s$ (f) $t=4.00s$

Figure 16: The deformation of the drop under shear flow at different time with $\sigma=7$ and $\gamma=2$.

Table 10: A mesh refinement study for moving interface in Example 6.7.

$L\times M\times N$	$\|u-u_h\|_\infty$	Order	$\|v-v_h\|_\infty$	Order	$\|w-w_h\|_\infty$	Order	$\|p-p_h\|_\infty$	Order
32	6.2938E-3	-	6.2938E-3	-	6.2938E-3	-	4.2421E-1	-
64	1.9058E-3	1.72	1.9058E-3	1.72	1.9058E-3	1.72	1.7992E-1	1.24
128	4.9177E-4	1.95	4.9183E-4	1.95	4.9183E-4	1.95	3.9615E-2	2.13

10 shows a mesh refinement analysis with $\sigma=5$ and $\gamma=1$, and the expected second order accuracy for the velocity are achieved. The computed solution on a $1024\times512\times512$ grid is taken as the reference solution for comparison.

The next simulation is performed on a $128\times64\times64$ grid. Fig. 16 shows the deformations of the drop at different time in shear flow with $\sigma=7$ and $\gamma=2$. As expected, the initial interface is a sphere and then stretched in shear flow. The effect of different shear rates γ on the deformation of the drop is studied. The time evolution of the Taylor shape parameter $D(t)$ with different shear rates is shown in Fig. 17, which shows that for the larger shear rate, the larger deformation of the drop is observed. But lager surface tension may have the opposite effect.

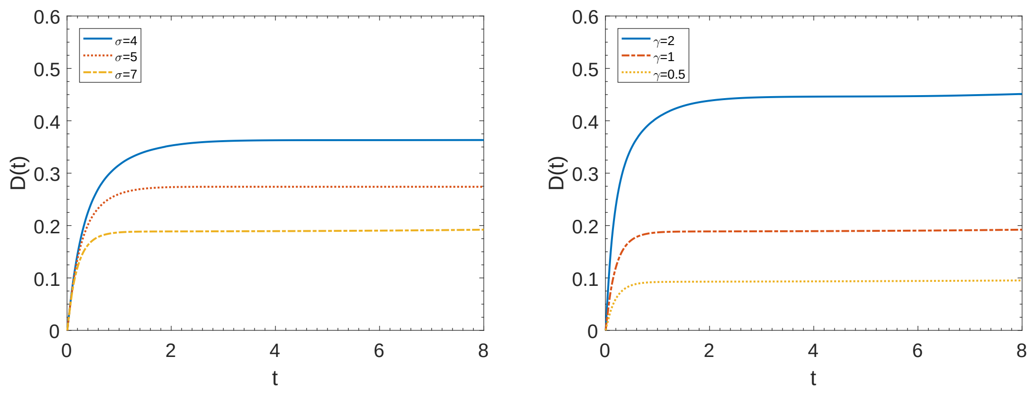


Figure 17: The evolution of the Taylor shape parameter $D(t)$ versus time with fixed $\gamma=1$ (left) and fixed $\sigma=7$ (right).

7 Conclusions

In this paper, a simple 3D IIM Stokes solver is employed to solve three-dimensional Stokes equations with singular forces, which is combined with the level set method. The method provides a fairly simple way to add the correction terms to the finite difference schemes, by applying the Taylor's expansions only along the normal direction and incorporating the jumps only in solution and up to its second normal derivative for the velocity and pressure, which makes the application of the method more general. A second order geometric iteration algorithm for computing orthogonal projections on the surface with third-order accuracy is employed. The fluid variables are computed on a staggered grid, which avoids the need of the pressure boundary condition. The CG-Uzawa type method is employed to solve discretized Stokes equations, and the number of iterations in the CG method is independent of the mesh size. The method can also make the volume conservation and the discrete divergence free condition kept very well. The proposed method is very efficient and flexible for different Stokes flow and Dirichlet boundary conditions. It should be pointed that the coefficient matrix gets larger when the grid is refined in the actual 3D numerical simulations, which will lead to more higher computational cost. Numerical results show that the present 3D IIM Stokes solver is second order accurate for both velocity and pressure. In the future work, the method will be extended to solve incompressible two-phase flow with interfaces in 3D.

Acknowledgments

Tan's research is supported by the Special Project on High-performance Computing under the National Key R&D Program (No. 2016YFB0200604), National Natural Science Foundation of China (11971502, 11571385), and Guangdong Natural Science Foundation (2017A030313017).

References

[1] A. S. Almgren, J. B. Bell, P. Colella, and T. Marthaler. A Cartesian grid projection method for the incompressible Euler equations in complex geometries. *SIAM Journal on Scientific Computing*, 18(5):1289–1309, 1997.

[2] M. J. Berger and R. J. LeVeque. A rotated difference scheme for Cartesian grids in complex geometries. *AIAA Paper CP-91-1602*, pages 1–9, 1991.

[3] R. P. Beyer Jr. A computational model of the cochlea using the immersed boundary method. *Journal of Computational Physics*, 98(1):145–162, 1992.

[4] D. K. Clarke, H. A. Hassan, and M. D. Salas. Euler calculations for multielement airfoils using Cartesian grids. *AIAA Journal*, 24(3):353–358, 1986.

[5] D. De Zeeuw and K. Powell. An adaptively-refined Cartesian Mesh solver for the Euler equations. Technical report, AIAA, 1991.

[6] S. Deng, K. Ito, and Z. Li. Three-dimensional elliptic solvers for interface problems and applications. *Journal of Computational Physics*, 184(1):215–243, 2003.

[7] R. Glowinski, T.-W. Pan, and J. Periaux. A fictitious domain method for external incompressible viscous flow modeled by Navier-Stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 112(1-4):133–148, 1994.

[8] S.-M. Hu and J. Wallner. A second order algorithm for orthogonal projection onto curves and surfaces. *Computer Aided Geometric Design*, 22(3):251–260, 2005.

[9] W. F. Hu, M. C. Lai, Y. Seol, and Y. N. Young. Vesicle electrohydrodynamic simulations by coupling immersed boundary and immersed interface method. *Journal of Computational Physics*, 317:66–81, 2016.

[10] W. X. Huang, C. B. Chang, and H. J. Sung. Three-dimensional simulation of elastic capsules in shear flow by the penalty immersed boundary method. *Journal of Computational Physics*, 231(8):3340–3364, 2012.

[11] L. J. Fauci. Interaction of oscillating filaments: A computational study. *Journal of Computational Physics*, 86(2):294–313, 1990.

[12] M.-C. Lai and Z. Li. A remark on jump conditions for the three-dimensional Navier-Stokes equations involving an immersed moving membrane. *Applied Mathematics Letters*, 14(2):149–154, 2001.

[13] M.-C. Lai and H.-C. Tseng. A simple implementation of the immersed interface methods for Stokes flows with singular forces. *Computers & Fluids*, 37(2):99–106, 2008.

[14] D.-V. Le, J. White, J. Peraire, K. M. Lim, and B. C. Khoo. An implicit immersed boundary method for three-dimensional fluid-membrane interactions. *Journal of Computational Physics*, 228(22):8427–8445, 2009.

[15] R. J. LeVeque. High resolution finite volume methods on arbitrary grids via wave propagation. In *Upwind and High-Resolution Schemes*, pages 491–518. Springer, 1988.

[16] R. J. Leveque and Z. Li. The immersed interface method for elliptic equations with discontinuous coefficients and singular sources. *SIAM Journal on Numerical Analysis*, 31(4):1019–1044, 1994.

[17] R. J. LeVeque and Z. Li. Immersed interface methods for Stokes flow with elastic boundaries or surface tension. *SIAM Journal on Scientific Computing*, 18(3):709–735, 1997.

[18] Z. Li. A fast iterative algorithm for elliptic interface problems. *SIAM Journal on Numerical Analysis*, 35(1):230–254, 1998.

[19] Z. Li and K. Ito. Maximum principle preserving schemes for interface problems with discontinuous coefficients. *SIAM Journal on Scientific Computing*, 23(1):339–361, 2001.

[20] Z. Li and K. Ito. *The Immersed Interface Method: Numerical Solutions of PDEs Involving Interfaces and Irregular Domains*, volume 33. SIAM, 2006.

[21] Z. Li, K. Ito, and M. C. Lai. An augmented approach for Stokes equations with a discontinuous viscosity and singular forces. *Computers & Fluids*, 36(3):622–635, 2007.

[22] Z. Li and M.-C. Lai. The immersed interface method for the Navier-Stokes equations with singular forces. *Journal of Computational Physics*, 171(2):822–842, 2001.

[23] J. Mohd-Yusof. Combined immersed boundary/B-spline method for simulations of flows in complex geometries. In *CTR Annual Research Briefs*, NASA Ames/Stanford University, 1997.

[24] C. S Peskin. Numerical analysis of blood flow in the heart. *Journal of Computational Physics*, 25(3):220–252, 1977.

[25] C. S Peskin. The immersed boundary method. *Acta Numerica*, 11:479–517, 2002.

[26] R. Dillon, L. Fauci, A. Fogelson, and D. Gaver III. Modeling biofilm processes using the immersed boundary method. *Journal of Computational Physics*, 129(1):57–73, 1996.

[27] J. C. Strikwerda. Finite difference methods for the Stokes and Navier-Stokes equations. *SIAM Journal on Scientific and Statistical Computing*, 5(1):56–68, 1984.

[28] P. Swarztrauber, R. Sweet, and J. C. Adams. FISHPACK: Efficient FORTRAN Subprograms for the Solution of Elliptic Partial Differential Equations. *UCAR Publication, July*, 1999.

[29] Z. Tan, K. M. Lim, and B. C. Khoo. A fast immersed interface method for solving Stokes flows on irregular domains. *Computers & Fluids*, 38(10):1973–1983, 2009.

[30] Z. Tan, K. M. Lim, and B. C. Khoo. An immersed interface method for Stokes flows with fixed/moving interfaces and rigid boundaries. *Journal of Computational Physics*, 228(18):6855–6881, 2009.

[31] Z. Tan, K. M. Lim, and B. C. Khoo. An implementation of MAC grid-based IIM-Stokes solver for incompressible two-phase flows. *Communications in Computational Physics*, 10(5):1333–1362, 2011.

[32] E. Y. Tau. Numerical solution of the steady Stokes equations. *Journal of Computational Physics*, 99(2):190–195, 1992.

[33] W. L. Wang and I. D. Boyd. Continuum breakdown in hypersonic viscous flows. In *40th AIAA Aerospace Sciences Meeting and Exhibit 2002*, 2002.

[34] X. Chen, X. Feng, and Z. Li. A direct method for accurate solution and gradient computations for elliptic interface problems. *Numerical Algorithms*, 80(3):709–740, 2019.

[35] X. Liu, F. Song, and C. Xu. An efficient spectral method for the inextensible immersed interface in incompressible flows. *Communications in Computational Physics*, 25(4):1071–1096, 2019.

[36] H. Xu, X. Fang, and L. Hu. Computing point orthogonal projections onto implicit surfaces. *Journal of Computer-Aided Design and Computer Graphics*, 20(12):1641–1646, 2008.

[37] J.-J. Xu, Y. Huang, M.-C. Lai, and Z. Li. A coupled immersed interface and level set method for three-dimensional interfacial flows with insoluble surfactant. *Communications in Computational Physics*, 15(2):451–469, 2014.

[38] J. J. Xu, W. Shi, W. F. Hu, and J. J. Huang. A level-set immersed interface method for simulating the electrohydrodynamics. *Journal of Computational Physics*, 400:108956, 2020.

[39] J.-J. Xu and H.-K. Zhao. An Eulerian formulation for solving partial differential equations along a moving interface. *Journal of Scientific Computing*, 19(1-3):573–594, 2003.

[40] S. Xu and G. D. Pearson. Computing jump conditions for the immersed interface method using triangular meshes. *Journal of Computational Physics*, 302(302):59–67, 2015.

[41] S. Xu and Z. J. Wang. Systematic derivation of jump conditions for the immersed interface method in three-dimensional flow simulation. *SIAM Journal on Scientific Computing*,

27(6):1948–1980, 2006.

[42] Y. Mori. Convergence proof of the velocity field for a Stokes flow immersed boundary method. *Communications on Pure and Applied Mathematics*, 61(9):1213–1263, 2008.

[43] Z. Li. On convergence of the immersed boundary method for elliptic interface problems. *Mathematics of Computation*, 84(293):1169–1188, 2014.