Journal of Computational Physics 395 (2019) 747–764

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

A coupled grid based particle and implicit boundary integral method for two-phase flows with insoluble surfactant

Shih-Hsuan Hsu [a], Jay Chu [b,*], Ming-Chih Lai [a], Richard Tsai [c]

[a] Department of Applied Mathematics, National Chiao Tung University, 1001, Ta Hsueh Road, Hsinchu 30010, Taiwan
[b] Department of Mathematics, National Tsing Hua University, 101 Section 2 Kuang-Fu Road, Hsinchu, Taiwan
[c] Department of Mathematics and Institute for Computational Engineering and Sciences (ICES), The University of Texas at Austin, TX 78712, USA

ARTICLE INFO

Article history:
Received 17 September 2018
Received in revised form 15 June 2019
Accepted 18 June 2019
Available online 25 June 2019

Keywords:
Interfacial flow
Insoluble surfactant
Closest point extensions
Grid based particle method

ABSTRACT

We develop a coupled grid based particle and implicit boundary integral method for simulation of three-dimensional interfacial flows with the presence of insoluble surfactant. The grid based particle method (GBPM, Leung and Zhao [20]) tracks the interface by the projection of the neighboring Eulerian grid points and does not require stitching of parameterizations nor body fitted moving meshes. Using this GBPM to represent the interface, the surfactant equation defined on the interface is discretized naturally following a new volumetric constant-along-surface-normal extension approach (Chu and Tsai [4]). We first examine the proposed scheme to solve the convection-diffusion equation for the problems with available analytical solutions. The numerical results demonstrate second-order accuracy of the scheme. We then perform a series of simulations for interfacial flows with insoluble surfactant. The numerical results agree well with the theory, and are comparable with other numerical works in literature.

1. Introduction

Surfactant is a chemical compound consisting of molecules with hydrophilic heads and hydrophobic tails. Surfactant molecules adhere to the two-phase fluid interface and reduce surface tension. They have important roles in many industrial applications such as pharmaceutical, cosmetic, and oil industries. The two-phase flow problem with surfactant draws a lot of attention, not only for the sake of its applications but also for the numerical points of interest. In order to predict the surfactant concentration on the fluid interface, it is necessary to solve a convection-diffusion equation defined on an evolving interface. It is a challenging task to accurately solve such type of problems that involve non-trivial moving interfaces embedded in three dimensions. In this work, we propose a numerical method to solve this three-dimensional problem in an efficient and relatively simple fashion.

Solving PDEs on surfaces has been explored by many researchers for decades. Dziuk, Elliott et al. developed a series of works to solve PDEs on surfaces by finite element approaches [5–7]. Their idea is to approximate both surfaces and solutions on suitable finite element spaces and correspondingly formulate the weak forms of the PDEs in those spaces. Error analysis and the optimal convergence rates for elliptic PDEs have been studied. We refer the readers to the review

* Corresponding author.
E-mail addresses: narisawa.am02g@g2.nctu.edu.tw (S.-H. Hsu), ccchu@math.nthu.edu.tw (J. Chu), mclai@math.nctu.edu.tw (M.-C. Lai), ytsai@ices.utexas.edu (R. Tsai).

https://doi.org/10.1016/j.jcp.2019.06.046
0021-9991/© 2019 Elsevier Inc. All rights reserved.

paper [8] for more details about the evolving surface finite element method (ESFEM). Barrett et al. used a parametric finite element approximation coupled with the ESFEM to develop a stable numerical scheme for solving two-phase flow with insoluble surfactant [1,2]. The method preserves good mesh properties and is proved to be stable and conserved. Finite difference methods which employ front-tracking and triangular meshes for solving the surfactant equation in 3D have been developed in past years. Muradoglu and Tryggvason [17] used a front-tracking method: they first wrote the equation in the integral form on each front element, and then converted the area integral of surface diffusion term into a line integral. That scheme is in spirit an explicit finite volume method for the surfactant convection-diffusion equation. De Jesus et al. [18] used the implicit finite volume method developed in [19] to solve the surfactant equation and applied it to simulate 3D two-phase flows with insoluble surfactant. There are other front-tracking methods that use the spherical harmonics expansion to represent the surface, together with some re-parametrization techniques for surface meshes and the corresponding surfactant concentration. Sorgentone and Tornberg [30] combined such an approach with a boundary integral method for Stokes flows to study surfactant-laden drop dynamics in 3D. While these front-tracking methods have the advantage of locally conserving the surfactant concentration, they suffer from the additional computational overhead due to the need to restructure and optimize the Lagrangian grid frequently. Recently, Seol et al. [26] have developed an equi-arclength parametrization technique to automatically control the Lagrangian meshes so that the total surfactant mass is conserved numerically. However, their scheme is limited to planar surfaces.

To avoid the drawback of re-meshing the surface, other embedding methods based on solving the equation on Eulerian domain have been developed. The main advantage of Eulerian embedding methods for solving surface PDEs is the elimination of the need for laying out Lagrangian meshes that move with and conform to the moving surface. Consequently, there is no need to remesh nor discretize the surface PDE on different patches and "glue" them together. A common approach for Eulerian methods is to extend the surface quantity (surfactant) and the equation defined on the interface to a narrowband around the surface and solve the extended PDE in that narrowband by some simple and robust finite difference methods. The level set method proposed by Xu and Zhao [33] and the closest point method proposed by Ruuth and Merriman [25] belong to such category. Xu and his collaborators also applied their method to simulate the interfacial flows with insoluble surfactant [35] and soluble surfactant [36]. Of course, such methods may need to provide additional conditions at the boundary of the narrowband in order to close the discretized systems. Naturally, the additional boundary conditions may influence the consistency of the solution computed in the narrowband to the surface PDE solution that one intends to solve.

There are methods that track the surface using implicit representation such as the level set method. Those methods typically require computing the solution of some PDEs (Hamilton-Jacobi equation in the case of the level set method) in a Eulerian domain around the surface. Instead, we employ the grid based particle method (GBPM) proposed by Leung and Zhao [20] for surface representation. GBPM is a numerical method to track an evolving surface such that the surface is represented by meshless and non-parametrized Lagrangian markers, defined explicitly using underlying reference Eulerian grid points. The aim is to avoid the common re-meshing process in a typical Lagrangian tracking method. As in a conventional Lagrangian method, the surface evolution is computed efficiently by solving an ordinary differential equation on each marker. In this way, GBPM combines the advantages of the Lagrangian and Eulerian approaches. Additionally, with the required local polynomial approximations in GBPM, one obtains conveniently the closest points to the surface and other geometric quantities. These informations are particularly useful for the present implicit boundary integral method [4] which we shall present later, and for the closest point method (CPM) [25,22]. Petras and Ruuth [23] coupled a modified grid based particle method and the CPM to solve the convection-diffusion equation on moving surfaces. As with other versions of the closest point method, their approach requires an extra "reinitialization" procedure at every time step. This procedure involves the closest point projection and interpolation to enforce the computed quantities being locally constant along surface normals. For systems with strong reaction terms and interfaces with high curvature relative to the underlying grid, the "non-surface intrinsic" reactions computed by the method could influence the values computed on the interface via the underlying diffusion. However, such nuances can be hard to observe if the width of the narrowband is thin and the "reinitialization" is done sufficiently frequently.

Recently Chu and Tsai [4] have proposed a framework to extend surface PDEs into a tubular neighborhood in the embedding space. For a wide class of surface PDEs, the properly extended solutions are naturally constant along the surface normals, without the need to enforce additional constraints. This property not only provides a natural way to close the discretized system but also allows the reduction of computational cost on interpolation. Furthermore, the resulting system enjoys better stability compared to other Eulerian embedding methods. In this paper, we shall refer to this framework as the IBIM framework. The term IBIM stands for Implicit Boundary Integral Method, which is a general approach to formulate volumetric extensions of boundary integrals, see [14,15] in detail. For the reasons listed above, we develop a method by combining GBPM and IBIM to solve the convection-diffusion equation on an evolving surface and apply to simulate the three-dimensional interfacial flows with insoluble surfactant. The proposed scheme enjoys the advantages of those two methods and is competitively accurate and efficient. Furthermore, this framework can be applied easily to electrohydrodynamic applications which require coupling with boundary integral solvers for electric potential computations.

The rest of the paper is organized as follows. In Section 2, we present the governing equations that describe an immiscible two-phase fluid in which the interface between those two phases is dynamic and under the influence of surfactant. We then introduce the ideas underlying the GBPM and IBIM, and couple those methods with the Navier-Stokes solver for interfacial flows with surfactant in Section 3. A more detailed GBPM-IBIM scheme for solving the convection-diffusion equation on an evolving surface and a series of numerical tests for accuracy are given in Section 4. Some simulation results for the three-dimensional interfacial flows with surfactant are presented in Section 5. Conclusion and future work are given in Section 6.

2. Interfacial flows with insoluble surfactant

In this section, we consider an incompressible two-phase flow in a fixed three-dimensional domain $\Omega = \Omega_1 \cup \Omega_2$ where the interface Σ separates Ω_1 (interior) from Ω_2 (exterior) and is a closed evolving surface. The interface is contaminated by an insoluble surfactant, which changes the surface tension accordingly. Since we mainly focus on the effect due to the existence of surfactant in this paper, for the sake of simplicity, we further assume that the two fluids have matched density and viscosity. Using the non-dimensionalization process described in [13], the governing equations in dimensionless form lead to the following single-fluid formulation as

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u} + \nabla p = \frac{1}{Re}\Delta\mathbf{u} + \frac{1}{ReCa}\mathbf{f}, \quad \text{in } \Omega \tag{1}$$

$$\nabla\cdot\mathbf{u} = 0, \quad \text{in } \Omega \qquad \mathbf{u} = \mathbf{u}_b, \quad \text{on } \partial\Omega \tag{2}$$

$$\mathbf{f} = \left(\nabla_s\sigma - 2\kappa\sigma\mathbf{n}\right)\delta(d), \quad \text{in } \Omega \tag{3}$$

$$\sigma = 1 - \beta\Gamma, \quad \text{on } \Sigma \tag{4}$$

$$\frac{D\Gamma}{Dt} + (\nabla_s\cdot\mathbf{u})\,\Gamma = \frac{1}{Pe_s}\Delta_s\Gamma, \quad \text{on } \Sigma \tag{5}$$

$$\frac{\partial \mathbf{X}}{\partial t} = \mathbf{u}(\mathbf{X}, t), \quad \text{on } \Sigma. \tag{6}$$

Eqs. (1)-(2) are the Navier-Stokes equations in which $\mathbf{u}$ is the fluid velocity and p is the pressure. The force $\mathbf{f}$ in Eq. (1) is the interfacial force that arises from the surface tension σ, which consists of Marangoni force $\nabla_s\sigma$ and capillary force $2\kappa\sigma\mathbf{n}$ as shown in Eq. (3). Here, $\mathbf{n}$ is the unit outward normal vector, δ is the Dirac delta function, d is the signed distance function to the interface, and κ is the mean curvature of Σ with the convention of positive sign when the surface is convex. The surface tension relating to the surfactant concentration is described by the Langmuir equation of state [27]. We adopt the linear approximation Eq. (4) as in [16]. Equation (5) is the convection-diffusion equation that governs the surfactant concentration along the interface [28]. Equation (6) shows that the interface moves along with the local fluid velocity determined by the Navier-Stokes equations. To complete the system, the above equations (1)-(6) should be accompanied with suitable initial conditions. For fluids with non-identical viscosity, only the governing equation (1) has to be modified. The modified equation and the corresponding numerical discretization can be found in Appendix A.

There are four dimensionless numbers: the Reynolds number Re describing the ratio between the inertial force and the viscous force, the capillary number Ca describing the strength of the surface tension, the parameter β, $0 \le \beta < 1$, measuring the sensitivity of surface tension change to the surfactant concentration, and the surface Peclet number Pe_s measuring the ratio of surfactant diffusion and convection effects along the interface.

3. Components of the proposed method

We first introduce some notations used in the paper. Assume Σ to be a C^2-closed surface in $\mathbb{R}^3$. For any small $\epsilon > 0$, we define an ϵ-narrowband of Σ as

$$T_\epsilon := \{\mathbf{x} \in \mathbb{R}^3 : \min_{\mathbf{y}\in\Sigma} |\mathbf{x} - \mathbf{y}| < \epsilon\}. \tag{7}$$

The closest point mapping $P_\Sigma : T_\epsilon \mapsto \Sigma$ is then given by

$$P_\Sigma(\mathbf{x}) = \arg\min_{\mathbf{y}\in\Sigma} |\mathbf{x} - \mathbf{y}|. \tag{8}$$

The closest point map P_Σ is well-defined on T_ϵ, provided that different parts of the surfaces are not close within the distance of 2ϵ, and that $\epsilon \in (0, \kappa_\infty^{-1})$, where κ_∞ is an upper bound of the curvatures of Σ.

3.1. The grid based particle method for tracking the interface

The Grid Based Particle Method (GBPM) [20] is a numerical method to track an evolving surface using Cartesian grids. The main idea is to represent the surface by meshless and non-parametrized Lagrangian particles and to compute the geometric information by local parametrization with suitable polynomial approximations. Every Lagrangian particle is in fact the closest point of some Cartesian grid node lying inside a thin narrowband around the surface. Given that the underlying Cartesian grid resolves the surface well, this approach provides a quasi-uniform point cloud sampling of the surface. The surface position is then updated by moving the particles according to some given velocity which this step is exactly as in the traditional Lagrangian method. After each time step, the surface is "resampled" in the sense that a new set of Lagrangian particles closest to the Cartesian grid nodes within the narrowband is constructed via interpolation of the old set of Lagrangian particles. In addition, GBPM provides a natural way to compute the geometric information accurately which is essential for IBIM. We will describe more details about IBIM in the following section.

Throughout this paper, we use the GBPM to track the surface motion and compute the corresponding geometric quantities, such as the principle curvatures, that are used in later computation. A demonstration of the different procedures that would take place at each time step is illustrated in Fig. 1. They are also outlined as follows.

1. **Initialization.** We shall denote the fluid interface by Σ. One of the fluids occupies the bounded region enclosed by Σ. We give the orientation of the surface by assigning the normals to the surface pointing outwards from the bounded region. Similar to the level set method, this information is equivalent to assigning "signs": points inside the bounded region are assigned to be negative, while points outside will take the positive sign. Collect all grid points $\mathbf{x}$ that are within T_ϵ as defined before. The surface Σ is represented by the closest points to surface of grid points in T_ϵ, that is, $\Sigma = \{P_\Sigma(\mathbf{x}) : \mathbf{x} \in T_\epsilon\}$. Record the sign on each grid nodes in T_ϵ, so that with each closest point to the grid node, one can recover the outward normal vector $\mathbf{n}$.
2. **Evolving surface.** Evolve the surface by solving the advection equation at each Lagrangian particles in Σ.
3. **Resampling surface.** After the evolution, those Lagrangian particles may no longer be the closest points of some Cartesian grids so we need to compute the new closest points to the surface for each grid points in T_ϵ by performing some local polynomial reconstruction.
4. **Activating new grid points.** Activate the grid points that are neighbors of the computational tube and find their corresponding closest points.
5. **Updating the computational narrowband.** Deactivate the grid points that have the distance to the surface larger than the given tube radius ϵ and obtain the new computational tube domain T_ϵ. As a result, this yields a new closest point representation of the surface Σ.

The original GBPM in [20] used the tube radius $\epsilon = 1.1\Delta x \sim 1.5\Delta x$ to track the surface motion. However, due to the bi-cubic interpolation used in the closest point based methods for solving surface PDEs (including the method used in this paper), one needs to use narrowband with larger radius. In [25], Ruuth and Merriman gave an estimated tube radius for interpolating the degree p of Lagrange polynomial in d-dimensional space as

$$\epsilon = \sqrt{(d-1)\left(\frac{p+1}{2}\right)^2 + \left(1+\frac{p+1}{2}\right)^2}\,\Delta x.$$

For instance, the interpolation of cubic polynomials in three-dimensional space leads to the tube radius of $\epsilon = 4.1\Delta x$ which is of the same order as the one used in the standard GBPM.

In the resampling step, an inaccurate polynomial fitting frequently occurs when the grid point has a larger distance from the surface. In the standard GBPM, these grid points are simply removed from T_ϵ in grid point deactivation step. Recently, Petras and Ruuth [23] proposed a modified GBPM using an osculating circle and sphere reconstruction to replace the local polynomial reconstruction when grid point deactivation occurs. However, this approach degrades the computing accuracy of the closest points and evaluations of geometric quantities.

In order to preserve the desired accuracy of the closest points and geometric quantities, we use the least squares polynomial fitting for local reconstruction of the surface throughout this work. Notice that, the inaccurate polynomial fitting occurs when the chosen particles in local reconstruction are insufficient. We can simply increase the number of candidates for choosing particles to resolve the issue. Fig. 2 illustrates the local reconstruction using two different regions of neighboring grid points. One can see that by simply enlarging the chosen region, the local reconstruction is more accurate. In our numerical experiments, we use the region of $\epsilon + \Delta x$ for choosing particles in local reconstruction.

The corresponding closest points on the surface and the geometric quantities can be obtained through the local reconstruction surface. We evaluate the closest point by minimizing the distance between the grid and the local polynomial surface, and compute the first and second fundamental forms to approximate the mean curvature κ, the normal vector $\mathbf{n}$ and the principal curvatures k_i at the closest point. To retain the sign information so that normal vector $\mathbf{n}$ is outward-pointing, and the signed distance function d takes positive values outside of interface and negative values inside of interface, we take the normal vector from the previous time step as a reference, see the details in [20]. The principle directions $\mathbf{t}_i$ can be obtained from the eigenvectors corresponding to the eigenvalue k_i of the shape operator which is given in terms of the components of the first and second fundamental forms by the Weingarten equations [10].

3.2. The implicit boundary integral method for solving surface PDEs

The Implicit Boundary Integral Method (IBIM) is a general framework for developing numerical methods for a class of integro-differential operators on non-parametrically defined surfaces or curves in $\mathbb{R}^n$, for instance implicit surfaces defined by level set method or closest point mapping, see e.g. [14,15,4]. The main idea is to extend the operators via the closest point map to the surface. By replacing surface gradient ∇_s and surface divergence $\nabla_s\cdot$ by Euclidean gradient and divergence

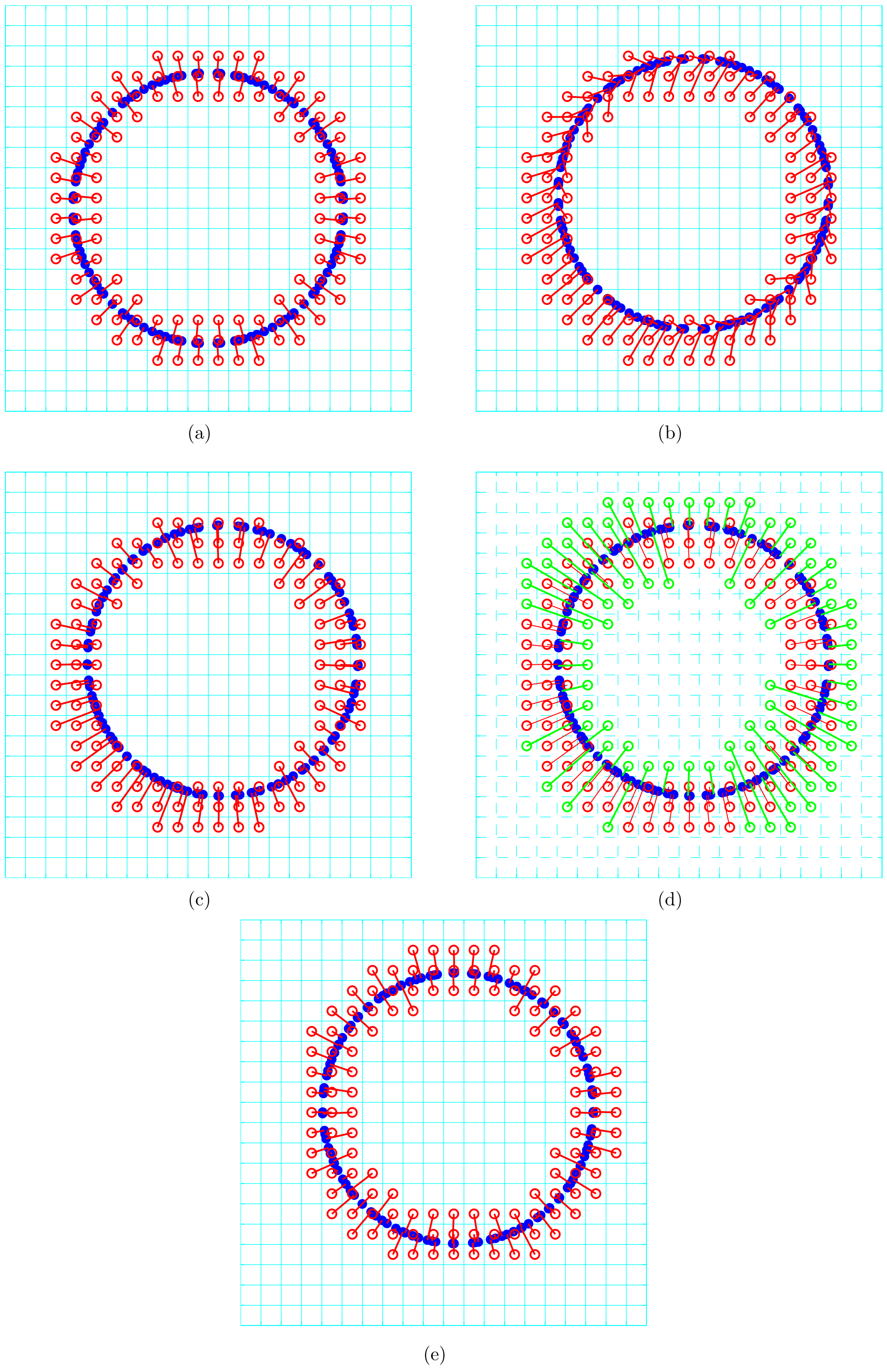

Fig. 1. The algorithm of grid based particle method: (a) Initialization (b) Evolving surface (c) Resampling surface (d) Activating new grid points (e) Updating the computational narrowband.

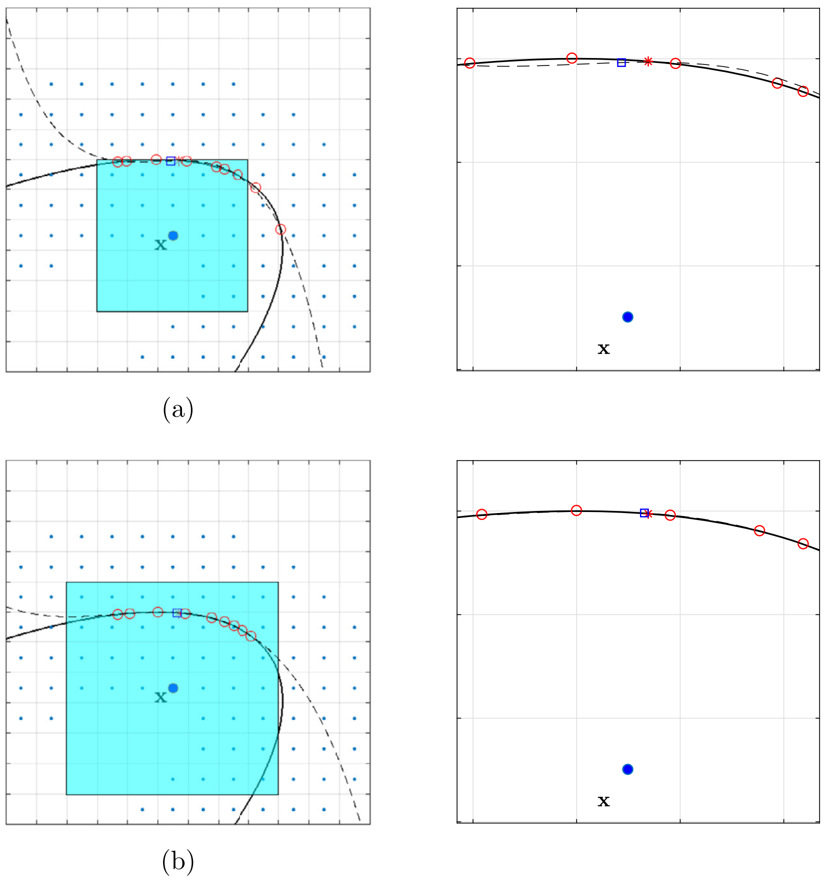

Fig. 2. The local cubic polynomial reconstruction (dashed curve $--$) for the surface (solid curve $-$) using Lagrangian markers (red circles $\circ$) inside the square region centered at $\mathbf{x}$. ($*$) and ($\square$) are the closest points to $\mathbf{x}$ on the surface and reconstruction surface, respectively. The right column is the zoom-ins of Figure (a) and (b), respectively. Figure (a) and (b) use different sizes of regions for reconstruction. This comparison shows that the accuracy of local reconstruction can be improved by enlarging the interpolation region. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

with suitable tensor coefficients, the method admits the solution which is constant along the surface normals, if the initial data has the same property. This property provides an equivalence between the solution computed in a thin narrowband around the surface and the solution to the intrinsic surface PDE. The discretization of the extended PDE can be done in Cartesian grids with finite difference or finite element schemes. We briefly illustrate the method in this subsection.

For any function f defined on Σ, we define its normal extension $\overline{f}$ on T_ϵ by $\overline{f}(\mathbf{x}) = f(P_\Sigma(\mathbf{x}))$ for any $\mathbf{x} \in T_\epsilon$. It can be shown that for any $f \in H^1(\Sigma)$, we have

$$\nabla_s f(P_\Sigma(\mathbf{x})) = A\nabla\overline{f}(\mathbf{x}), \quad \text{for all } \mathbf{x} \in T_\epsilon, \tag{9}$$

where $A(\mathbf{x};\mu) = A_0(\mathbf{x}) + \mu A_1(\mathbf{x})$. Here, $A_0(\mathbf{x})$ is the scaling tensor, $A_1(\mathbf{x})$ is an extra term to make the matrix tensor A non-degenerate, and μ is any real number. Two matrix tensors are calculated by

$$A_0(\mathbf{x}) := \sigma_1^{-1}\mathbf{t}_1 \otimes \mathbf{t}_1 + \sigma_2^{-1}\mathbf{t}_2 \otimes \mathbf{t}_2,$$
$$A_1(\mathbf{x}) := \mathbf{n} \otimes \mathbf{n},$$

where $\mathbf{t}_1$, $\mathbf{t}_2$ are the two orthonormal tangent vectors corresponding to the directions that yield the principle curvatures of Σ, $\mathbf{n}$ is the unit normal vector of Σ, σ_1 and σ_2 are two largest singular values of DP_Σ, where $DP_\Sigma(\mathbf{x})$ is the Jacobian matrix of $P_\Sigma(\mathbf{x})$ [15]. DP_Σ can be approximated by simple finite differencing.

Similarly, we can extend the surface divergence for a vector field $\mathbf{F}$ defined on Σ to T_ϵ by

$$(\nabla_s \cdot \mathbf{F})(P_\Sigma(\mathbf{x})) = J^{-1}\nabla \cdot (JA\overline{\mathbf{F}}(\mathbf{x})), \qquad \text{for all } \mathbf{x} \in T_\epsilon, \tag{10}$$

where J is the Jacobian that comes from the change of variables by the closest point mapping and can be computed by

$$J = \sigma_1\sigma_2. \tag{11}$$

In the present GBPM surface representation, we can simply compute those singular values by σ_i by

$$\sigma_i(\mathbf{x}) = 1/(1 + d(\mathbf{x})k_i(P_\Sigma(\mathbf{x}))) \tag{12}$$

directly, where $d(\mathbf{x})$ is the signed distance between $\mathbf{x}$ and $P_\Sigma(\mathbf{x})$, and $k_i(P_\Sigma(\mathbf{x}))$ are the corresponding principal curvatures at the surface. The curvatures used here are signed determined by the second fundamental form. In particular, the surface Laplacian of f can be extended to T_ϵ by simply substituting the surface gradient of f defined in Eq. (9) into

$$\Delta_s f(P_\Sigma(\mathbf{x})) = J^{-1}\nabla \cdot (JA^2\nabla \overline{f}(\mathbf{x})), \qquad \text{for all } \mathbf{x} \in T_\epsilon.$$

Analogously, when Σ is a curve in $\mathbb{R}^2$, the scaling tensor $A_0(x)$ is simply given by

$$A_0(x) = \sigma^{-1}\mathbf{t} \otimes \mathbf{t},$$

where $\mathbf{t}$ is the unit tangent vector of Σ and σ is the largest singular value of DP_Σ. To simplify the method, we usually choose the free variable $\mu = \sigma^{-1}$ and obtain the following simple relations

$$\nabla_s f(P_\Sigma(\mathbf{x})) = \sigma^{-1}\nabla \overline{f}(\mathbf{x}), \quad \text{for all } \mathbf{x} \in T_\epsilon,$$

and

$$(\nabla_s \cdot \mathbf{F})(P_\Sigma(\mathbf{x})) = \sigma^{-1}\nabla \cdot (\overline{\mathbf{F}}(\mathbf{x})), \qquad \text{for all } \mathbf{x} \in T_\epsilon,$$

for any smooth functions f and F defined on Σ.

To deal with Neumann boundary condition on ∂T_ϵ, a local interpolation strategy is used to close the system. We project the ghost grid points outside of T_ϵ into it along the normal direction and interpolate the projected position by nearby inner grid points. It can be shown the closure is second-order accurate and stable for parabolic equations, see [4] for detail.

3.3. The coupled scheme for interfacial flows

In the following, we describe our proposed numerical scheme for solving Eqs. (1)-(6). We assume the computational domain to be a rectangular domain $\Omega = [a, b] \times [c, d] \times [e, f]$. Within this domain, a uniform lattice grid with mesh width h is employed. The velocity components u, v and w are defined at usual staggered MAC grid [12] as

$$\begin{aligned}
(x_{i-1/2}, y_j, z_k) &= (a + (i-1)h, c + (j-1/2)h, e + (k-1/2)h),\\
(x_i, y_{j-1/2}, z_k) &= (a + (i-1/2)h, c + (j-1)h, e + (k-1/2)h),\\
(x_i, y_j, z_{k-1/2}) &= (a + (i-1/2)h, c + (j-1/2)h, e + (k-1)h).
\end{aligned}$$

The pressure p is defined at the cell center labeled as

$$(x_i, y_j, z_k) = (a + (i-1/2)h, c + (j-1/2)h, e + (k-1/2)h).$$

Notice that, the Cartesian grid points used by GBPM to represent the interface and by IBIM for solving surfactant concentration are all chosen at cell center points $\mathbf{x} = (x_i, y_j, z_k)$.

Let Δt be the time step size, and n be the time step index. At the beginning of each time step n, the fluid velocity $\mathbf{u}^n$, the interface position $\mathbf{X}^n$, and the surfactant concentration Γ^n are all given. More precisely speaking, in the GBPM framework, the interface position Σ^n are those projection points of the grid points $\mathbf{x} = (x_i, y_j, z_k)$ within the narrowband T_ϵ so that the signed distance function d and the mean curvature κ of the interface can be easily obtained. The time advancement of one time step can be summarized as follows.

1. Compute the surface tension and interfacial force

   $$\sigma^n = 1 - \beta\Gamma^n, \quad \mathbf{f}^n = \left(A\nabla\sigma^n - 2\kappa^n\sigma^n\mathbf{n}^n\right)\delta_h(d^n),$$

   directly at each grid point $\mathbf{x}$ in the neighborhood of interface Σ^n. Notice that, all the surface geometric quantities in the above equation are evaluated at corresponding closest point $\mathbf{X} = P_\Sigma(\mathbf{x})$. Since the force $\mathbf{f}^n$ is defined at the cell center point $\mathbf{x}$, we need to further interpolate the value at the corresponding staggered grid point used in the following Navier-Stokes solver.
2. Solve the Navier-Stokes equations using the second-order accurate projection method proposed in [11] to update the new velocity $\mathbf{u}^{n+1}$.

$$\begin{aligned}
&\frac{3\mathbf{u}^* - 4\mathbf{u}^n + \mathbf{u}^{n-1}}{2\Delta t} + 2\left(\mathbf{u}^n \cdot \nabla_h\right)\mathbf{u}^n - \left(\mathbf{u}^{n-1} \cdot \nabla_h\right)\mathbf{u}^{n-1} + \nabla_h p^n \\
&\qquad\qquad = \frac{1}{Re}\Delta_h \mathbf{u}^* + \frac{\mathbf{f}^n}{Re\,Ca}, \quad \mathbf{u}^*|_{\partial\Omega} = \mathbf{u}_b, \\
&\Delta_h p^* = \frac{3}{2\Delta t}\nabla_h \cdot \mathbf{u}^*, \quad \frac{\partial p^*}{\partial \mathbf{n}}|_{\partial\Omega} = 0, \\
&\mathbf{u}^{n+1} = \mathbf{u}^* - \frac{2\Delta t}{3}\nabla_h p^*, \\
&\nabla_h p^{n+1} = \nabla_h p^* + \nabla_h p^n - \frac{2\Delta t}{3Re}\Delta_h(\nabla_h p^*).
\end{aligned}$$

3. Move the interface by GBPM and solve the surfactant concentration equation by IBIM with Strang splitting technique [31] as

$$\Gamma^{n+1} = \mathcal{A}_{\Delta t/2}\mathcal{B}_{\Delta t}\mathcal{A}_{\Delta t/2}\Gamma^n, \tag{13}$$

where $\mathcal{A}$ represents the simple transport operator and $\mathcal{B}$ represents the discrete convection-diffusion operator that shall be defined in details later. The complete numerical details are given in the following Section 4.

The overall computational cost is dominated by Step 2 and 3 in the above scheme. In Step 2, the projection method for Navier-Stokes equations involves solving three modified Helmholtz equations for the velocity and one Poisson equation for the pressure. Both types of equations can be efficiently solved by applying standard fast direct solvers with $O(N^d \log(N))$ computational cost; here N is the number of grid points in each spatial dimension and d is the dimension of space. The complexity of Step 3 scales linearly with the total number of grid points in the narrow band T_ϵ so for $\epsilon \sim N^{-1}$; it is of order $O(N^{d-1})$. The total cost is comparable to other conventional methods such as the closest point method or front-tracking method.

4. Approximation of the convection-diffusion equation on an evolving surface

In this section, we elaborate the numerical details in Eq. (13) and introduce a coupled GBPM and IBIM to solve the convection-diffusion equation on an evolving surface. Here, we assume the closed surface $\Sigma(t)$ can be immersed in a fixed three-dimensional domain Ω. It is deformable and moving with a given velocity $\mathbf{u}$ (or obtained from the Navier-Stokes equations in previous section) as

$$\frac{\mathrm{d}\mathbf{X}}{\mathrm{d}t} = \mathbf{u}(\mathbf{X}, t) \quad \text{on } \Sigma(t). \tag{14}$$

Along this evolving surface $\Sigma(t)$, the concentration follows the convection-diffusion equation

$$\frac{D\Gamma}{Dt} + (\nabla_s \cdot \mathbf{u})\,\Gamma = \frac{1}{Pe_s}\Delta_s\Gamma + g \quad \text{on } \Sigma(t), \tag{15}$$

where $\frac{D}{Dt}$ is material derivative, ∇_s and $\Delta_s = \nabla_s \cdot \nabla_s$ are the surface gradient and surface Laplacian defined as before, and g is a source term.

Since the surface is tracked explicitly by the closest points in the GBPM setting, we denote Σ^n the set of closest points used for surface representation at time $n\Delta t$, and the function Γ defined at Σ^n denoted by Γ^n. Instead of solving the convection-diffusion equation (15) directly, we adopt the Strang splitting technique [31] to solve the equation. We first write the equation into the simple transport equation

$$\frac{D\Gamma}{Dt} = 0, \tag{16}$$

and the convection-diffusion equation

$$\frac{\partial\Gamma}{\partial t} + (\nabla_s \cdot \mathbf{u})\,\Gamma = \frac{1}{Pe_s}\Delta_s\Gamma + g. \tag{17}$$

The Strang splitting consists of the following three steps:

(1) Solve Eq. (16) by GBPM with one half of time step.
First, we update the new surface position $\Sigma^{n+1/2}$ from Σ^n by moving the markers according to the velocity field. Then we obtain the value of the concentration $\hat{\Gamma}^n$ at $\Sigma^{n+1/2}$ by a local approximation of Γ^n at the markers' location (recall that Γ^n are defined on the grid nodes in the narrowband around Σ^n). We write the approximation in the matrix form, $\hat{\Gamma}^n = \mathcal{A}_{\Delta t/2}\Gamma^n$, where $\mathcal{A}$ denotes the formal matrix operator whose entries correspond to the moving surface, re-sampling and interpolation presented in Subsection 3.1.

(2) Solve Eq. (17) by IBIM with one time step.
We decompose the velocity into the tangential and normal components as $\mathbf{u} = \mathbf{v} + v\mathbf{n}$. Then, the surface divergence of velocity can be rewritten as

$$\nabla_s \cdot \mathbf{u} = \nabla_s \cdot \mathbf{v} + 2v\kappa, \tag{18}$$

where $\kappa = \frac{1}{2}\nabla_s \cdot \mathbf{n}$ is the mean curvature of the surface. Applying the IBIM to Eq. (17), we obtain the embedding equation in T_ϵ

$$\frac{\partial \Gamma}{\partial t} + \left(J^{-1}\nabla \cdot (JA\mathbf{v}) + 2v\kappa\right)\Gamma = \frac{1}{Pe_s} J^{-1}\nabla \cdot (JA^2\nabla\Gamma) + g.$$

Then we use Crank-Nicolson scheme for the time integration and second-order central difference for the spatial discretization to update the surface function $\hat{\Gamma}^{n+1}$ as

$$\begin{aligned}\frac{\hat{\Gamma}^{n+1}_{ijk} - \hat{\Gamma}^{n}_{ijk}}{\Delta t} &+ \frac{1}{2}\left(J^{-1}_{ijk}\nabla_h \cdot (JA\mathbf{v})_{ijk} + 2v_{ijk}\kappa_{ijk}\right)(\hat{\Gamma}^{n+1}_{ijk} + \hat{\Gamma}^{n}_{ijk}) \\ &= \frac{1}{2Pe_s} J^{-1}_{ijk}\nabla_h \cdot (J_{ijk}A^2_{ijk}\nabla_h(\hat{\Gamma}^{n+1}_{ijk} + \hat{\Gamma}^{n}_{ijk})) + g^{n+1/2}_{ijk},\end{aligned} \tag{19}$$

for each $\mathbf{x}_{ijk}$ in $T^{n+1/2}_\epsilon$. In short, solving $\hat{\Gamma}^{n+1}$ by the above scheme can be simply written as $\hat{\Gamma}^{n+1} = \mathcal{B}_{\Delta t}\hat{\Gamma}^{n}$.

(3) Solve Eq. (16) by GBPM with one half of time step.
We repeat the procedure in step (1) by using $\Sigma^{n+1/2}$ and $\hat{\Gamma}^{n+1}$ as the initial conditions and update the new surface position Σ^{n+1} and function Γ^{n+1} by $\Gamma^{n+1} = \mathcal{A}_{\Delta t/2}\hat{\Gamma}^{n+1}$.

Fig. 3 shows the mechanism of the splitting algorithm for advancing the system within one time step. In the example, we put an unit circle with surface function $\Gamma(x, y, t^n) = x$ under a shear flow. The solution of the surface convection-diffusion equation is shown on the interface according to the colorbar on the side.

4.1. Numerical results

In this subsection, we perform a series of tests to check the accuracy of the above proposed algorithm for solving the convection-diffusion equation on an evolving surface. In these examples, the surface is moving along a given velocity. This way, the surface position in time can be determined in a priori, and we can design the analytical solution Γ that satisfies the convection-diffusion equation (15). The errors computed in each test are presented in Tables 1–4. Throughout this section, we choose a uniform Cartesian mesh $h = \Delta x = \Delta y = \Delta z$ and compute the corresponding L_∞, L_1 and L_2 errors using the following formulas

$$\|e_h\|_\infty = \max_{\mathbf{x}_i \in T_\epsilon} e_h(P_\Sigma(\mathbf{x}_i)), \tag{20}$$

$$\|e_h\|_1 = \int_{T_\epsilon} e_h\left(P_\Sigma(\mathbf{x})\right)\delta_h\left(d\left(\mathbf{x}\right)\right) J(\mathbf{x})d\mathbf{x}, \tag{21}$$

$$\|e_h\|_2 = \left(\int_{T_\epsilon} e^2_h\left(P_\Sigma(\mathbf{x})\right)\delta_h\left(d\left(\mathbf{x}\right)\right) J(\mathbf{x})d\mathbf{x}\right)^{\frac{1}{2}}, \tag{22}$$

where $e_h(P_\Sigma(\mathbf{x})) = |\Gamma(P_\Sigma(\mathbf{x})) - \Gamma_h(P_\Sigma(\mathbf{x}))|$ is the error of the solution Γ on those grid points within the narrowband T_ϵ. Here, $d(\mathbf{x})$ is the signed distance function obtained from GBPM, and $\delta_h(x)$ is the discrete delta function defined by

$$\delta_h(x) = \begin{cases} \frac{1}{4h} + \frac{1}{4h}\cos\frac{\pi x}{2h} & \text{if } -2h < x < 2h \\ 0 & \text{otherwise} \end{cases} \tag{23}$$

Note that, the integrals in Eq. (21)-(22) are approximated by the trapezoidal rule on Cartesian grid points within T_ϵ. The details for computing integrals over a curve or surface via closest point mapping can also be found in [15]. The linear system in Eq. (19) is solved by the GMRES iterative method. Since we use an explicit scheme for evolving the Lagrangian markers, the time step size Δt is constrained by the order $O(h)$ due to the CFL condition. Furthermore, as we do not wish to consider the situation in which the interface advances more than one grid point in each time step and evolves outside of the narrowband, we further apply the restriction $\|\mathbf{u}\|_\infty \Delta t \le h$.

In the following, we consider the convection-diffusion equation (15) on the cases of 2D curves and 3D surfaces where the motions of curves and surfaces are known in a priori. For simplicity, we set the surface Peclet number $Pe_s = 1$.

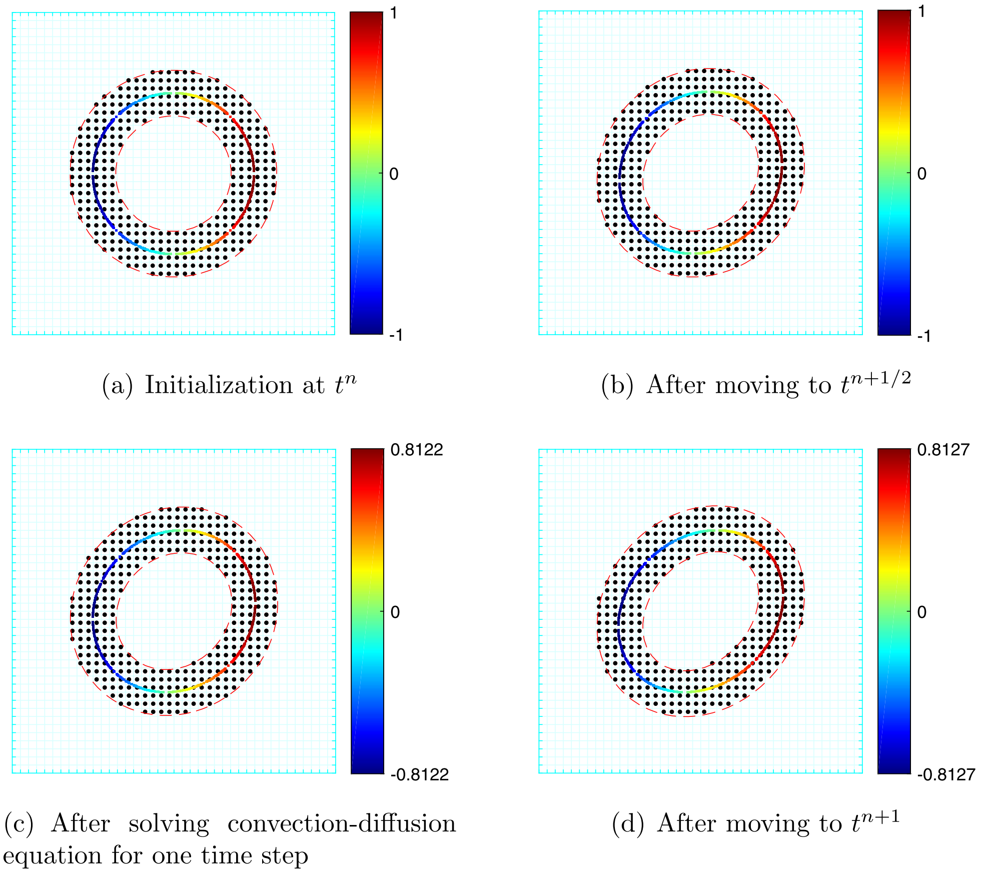

Fig. 3. The Strang splitting algorithm for solving the convection-diffusion equation defined on an evolving surface in one time step. The figure illustrates the change of interface Lagrangian markers and their corresponding grid nodes in the narrowband. The color along the interface indicates the magnitude of the solution Γ.

Table 1
The errors and their convergent rates for the designed analytical solution Γ of convection-diffusion equation on an expanding circle at $t = 0.5$.

h	$\|e_h\|_\infty$	rate	$\|e_h\|_1$	rate	$\|e_h\|_2$	rate
1/10	2.46×10^{-4}	–	3.42×10^{-3}	–	7.94×10^{-4}	–
1/20	5.98×10^{-5}	2.04	8.39×10^{-4}	2.03	1.95×10^{-4}	2.03
1/40	1.48×10^{-5}	2.02	2.06×10^{-4}	2.02	4.81×10^{-5}	2.02
1/80	3.69×10^{-6}	2.00	5.13×10^{-5}	2.01	1.19×10^{-5}	2.00
1/160	9.26×10^{-7}	2.00	1.28×10^{-5}	2.00	3.00×10^{-6}	2.00

Example 1: A 2D expanding circle

Although the description of the present scheme is based on 3D formulation, the method can be applied to 2D in a simpler manner when solving diffusion equation on Σ as explained in subsection 3.2. As the first test, we follow the example given in [9] by considering the interface as a unit circle with velocity field $\mathbf{u} = 5\mathbf{n}$. Therefore, the interface is an expanding circle and can be represented by $r(t) = 1 + 5t$. The function

$$\Gamma(x, y, t) = e^{\frac{4}{5r(t)}} \frac{xy}{r^3(t)}$$

satisfies the exact solution of Eq. (15) with initial condition $\Gamma(x, y, 0)$ on the unit circle. The solutions are computed up to time $t = 0.5$ with time step size $\Delta t = h/5$. The numerical errors and their convergence rates for different mesh widths are shown in Table 1. One can see that the numerical results show second-order convergence clearly.

Example 2: A 2D unit circle under simple shear flow

In this test, we consider a unit circle under simple shear flow $\mathbf{u} = (y, 0)$. Thus, the interface evolves to a slanted ellipse of the form

Table 2
The errors and their convergent rates for the designed analytical solution Γ of convection-diffusion equation on a 2D unit circle under the simple shear flow at $t = 1$.

h	$\|e_h\|_\infty$	rate	$\|e_h\|_1$	rate	$\|e_h\|_2$	rate
1/40	2.02×10^{-4}	–	5.14×10^{-4}	–	2.32×10^{-4}	–
1/80	4.61×10^{-5}	2.13	1.23×10^{-4}	2.06	5.51×10^{-5}	2.07
1/160	1.15×10^{-5}	2.01	3.05×10^{-5}	2.01	1.36×10^{-5}	2.02
1/320	2.86×10^{-6}	2.01	7.61×10^{-6}	2.00	3.40×10^{-6}	2.00
1/640	7.16×10^{-7}	2.00	1.90×10^{-6}	2.00	8.50×10^{-7}	2.00

Table 3
The errors and their convergent rates for the designed analytical solution Γ of convection-diffusion equation on a 3D expanding sphere at $t = 0.5$.

h	$\|e_h\|_\infty$	rate	$\|e_h\|_1$	rate	$\|e_h\|_2$	rate
1/10	6.25×10^{-4}	–	1.45×10^{-2}	–	2.30×10^{-3}	–
1/20	1.54×10^{-4}	2.02	3.56×10^{-3}	2.02	5.74×10^{-4}	2.00
1/40	3.83×10^{-5}	2.01	8.84×10^{-4}	2.01	1.42×10^{-4}	2.01
1/80	9.63×10^{-6}	1.99	2.21×10^{-4}	2.00	3.57×10^{-5}	2.00

Table 4
The errors and their convergent rates for the designed analytical solution Γ of convection-diffusion equation on a 3D oscillating ellipsoid at $t = 4$.

h	$\|e_h\|_\infty$	rate	$\|e_h\|_1$	rate	$\|e_h\|_2$	rate
1/10	7.15×10^{-3}	–	2.89×10^{-2}	–	8.80×10^{-3}	–
1/20	1.41×10^{-3}	2.34	7.03×10^{-3}	2.04	2.23×10^{-3}	1.98
1/40	3.29×10^{-4}	2.10	1.72×10^{-3}	2.03	5.53×10^{-4}	2.01
1/80	7.82×10^{-5}	2.07	4.25×10^{-4}	2.02	1.37×10^{-4}	2.01

$$\Sigma(t) = \left\{\mathbf{x}(\theta, t) = \big(\cos(\theta) + t\sin(\theta), \sin(\theta)\big),\ 0 \le \theta \le 2\pi\right\}.$$

Along the interface $\Sigma(t)$, we set the function $\Gamma(\theta, t) = e^{-t}\cos(\theta)$ as the exact solution of Eq. (15) with an extra source function g defined by $g = \frac{D\Gamma}{Dt} + (\nabla_s \cdot \mathbf{u})\,\Gamma - \Delta_s\Gamma$. Again, the initial interface is a unit circle with initial condition $\Gamma(\theta, 0)$. The solutions are computed up to time $t = 1$ with time step size $\Delta t = h/2$. We set the largest mesh width $h = 1/40$ to make sure that the closest point mapping is well-defined in T_ϵ during computations. The numerical errors and their convergence rates are shown in Table 2 which shows the clear second-order convergence as well.

Example 3: A 3D expanding sphere

In this example, we consider the case of an expanding sphere with the velocity $\mathbf{u} = 2\mathbf{n}$. Thus, the expanding sphere can be written in terms of radius function $r(t) = 1 + 2t$. Along the surface, we define the function

$$\Gamma(x, y, z, t) = e^{1/r(t)}\frac{z}{r^2(t)},$$

which is the exact solution of Eq. (15) with the initial condition $\Gamma(x, y, z, 0)$ on the unit sphere. The solutions are computed up to time $t = 0.5$ with time step size $\Delta t = h/2$. The numerical errors and their convergence rates are shown in Table 3 which again show clean second-order convergence for different norms too.

Example 4: A 3D oscillating ellipsoid

In this test, we consider the case of an oscillating ellipsoid developed in [6],

$$\Sigma(t) = \left\{\mathbf{x} = (x, y, z) : \frac{x^2}{a^2(t)} + y^2 + z^2 = 1\right\},$$

where $a(t) = \sqrt{1 + \sin(2t)/4}$, and associated velocity field $\mathbf{u} = \left(\frac{a'(t)}{a(t)}x, 0, 0\right)$. More generally, the oscillating ellipsoid has the parametric form

$$\mathbf{x}(\theta, \phi, t) = \Big(a(t)\sin(\theta)\cos(\phi), \sin(\theta)\sin(\phi), \cos(\theta)\Big),$$

where $0 \le \theta < \pi$ and $0 \le \phi < 2\pi$. The exact solution is defined by $\Gamma(\theta, \phi, t) = a(t)\sin^2(\theta)\cos(\phi)\sin(\phi)$ on the interface $\Sigma(t)$ and an extra source term $g = \frac{D\Gamma}{Dt} + (\nabla_s \cdot \mathbf{u})\,\Gamma - \Delta_s\Gamma$ is added so that the surface function Γ satisfies Eq. (15). The solutions are computed up to time $t = 4$ with time step size $\Delta t = h$. The numerical errors and their convergence rates are shown in Table 4 which confirms the desired second-order convergence again. We further show the snapshots for the numerical solution at different times in Fig. 4.

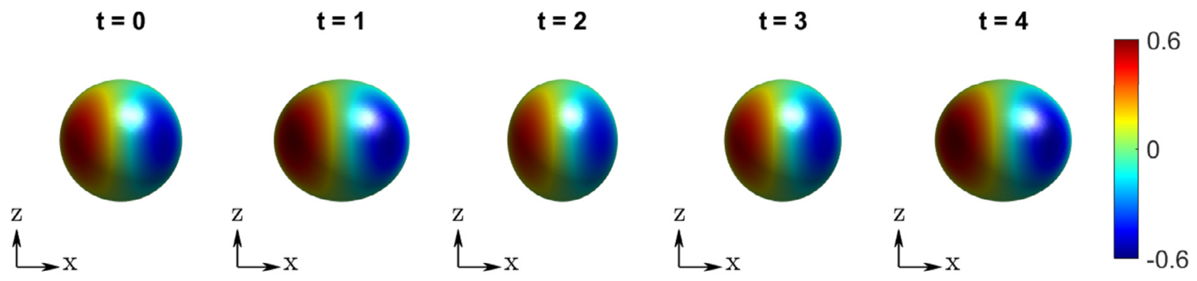

Fig. 4. The snapshots of numerical solution at $T = 0, 1, 2, 3$ and 4 on the oscillating ellipsoid.

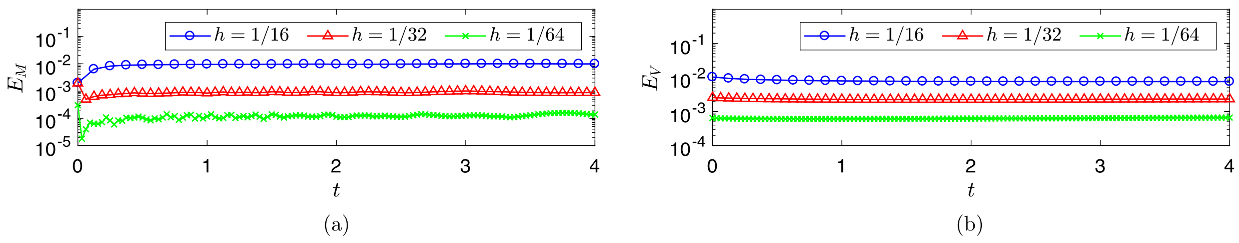

Fig. 5. Relative errors of (a) total surfactant mass and (b) droplet volume.

5. Numerical results on interfacial flows

In this section, we perform a series of numerical simulations for three-dimensional interfacial flow with insoluble surfactant. First, we simulate an ellipsoid droplet under a quiescent flow. We then validate the presented method by a numerical convergence study for velocity components, total surfactant mass, and droplet volume. Next, we consider a unit sphere droplet under shear flow and study the effect of dimensionless numbers. We further compare the numerical results of the deformation factor and the inclination angle with small deformation theory. Finally, we study the influence of surfactant for the interaction of two droplets under shear flow.

5.1. An ellipsoid droplet in quiescent flow

As a first test, we perform a numerical convergence study of the presented algorithm. We consider a droplet in the form of an ellipsoid at the initial time, with the interface $\Sigma(0) = \{x^2 + y^2/(1.6)^2 + z^2/(0.4)^2 = 1\}$ immersed in a quiescent flow. Simulations are computed on uniform Cartesian grids over the domain $\Omega = [-2, 2] \times [-2, 2] \times [-2, 2]$. Three uniform grid spacings $h = 1/16, 1/32, 1/64$ are considered, with the time step size $\Delta t = h/4$. The initial surfactant concentration on the interface is set to be $\Gamma_0 = 1$. The dimensionless numbers are chosen as $Re = 0.1, Ca = 0.1, \beta = 0.6$, and $Pe_s = 1$. The computations are performed up to time $t = 4$.

Since the surfactant is insoluble, the total surfactant mass must be conserved. Furthermore, since the fluid is incompressible, the total volume of the droplet is also a theoretically conserved quantity. The total surfactant mass $M(t)$ can be computed by

$$M(t) = \int_{\Omega} \Gamma\left(P_{\Sigma}(\mathbf{x})\right) \delta_h\left(d\left(\mathbf{x}\right)\right) J(\mathbf{x}) \mathrm{d}\mathbf{x},$$

with δ_h as defined in Eq. (23), J defined in (11), and the volume of the droplet $V(t)$ can be approximated by

$$V(t) = \int_{\Omega} \left(1 - H_h\left(d\left(\mathbf{x}\right)\right)\right) \mathrm{d}\mathbf{x},$$

where

$$H_h(x) = \begin{cases} 0, & \text{if } x \le -2h, \\ \frac{1}{2} + \frac{x}{4h} + \frac{1}{2\pi}\sin\frac{\pi x}{2h}, & \text{if } -2h < x < 2h, \\ 1, & \text{if } x \ge 2h, \end{cases}$$

is the corresponding regularized Heaviside function.

Fig. 5(a) shows the relative errors of total surfactant mass $E_M = \frac{|M(t)-M_0|}{M_0}$ with three different meshes widths $h = 1/16, 1/32, 1/64$ while Fig. 5(b) shows the relative errors of droplet volume $E_V = \frac{|V(t)-V_0|}{V_0}$. Here, M_0 and V_0 are the initial

Table 5
The mesh refinement results for the velocity components u, v and w at $t = 4$.

h	$\|e_h^u\|_\infty$	rate	$\|e_h^v\|_\infty$	rate	$\|e_h^w\|_\infty$	rate
1/16	1.78E-01	–	6.84E-02	–	1.29E-01	–
1/32	7.59E-02	1.23	2.82E-02	1.28	5.71E-02	1.17

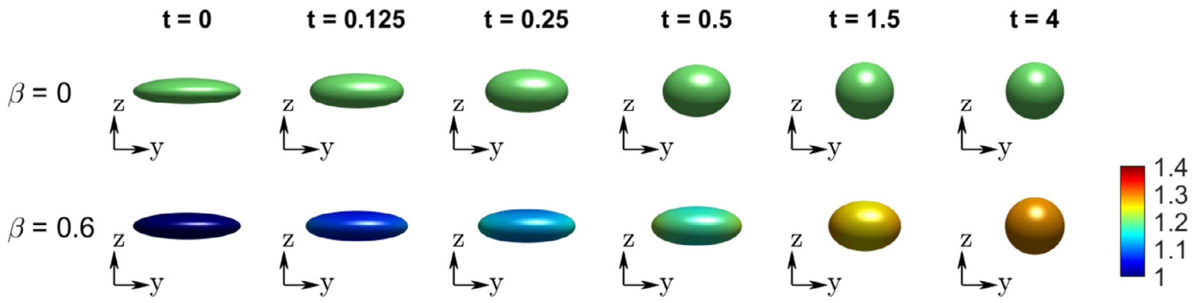

Fig. 6. The clean droplet morphology (top, $\beta = 0$) and contaminated droplet morphology (bottom, $\beta = 0.6$) in a quiescent flow. The color coding shows the distribution of surfactant concentration.

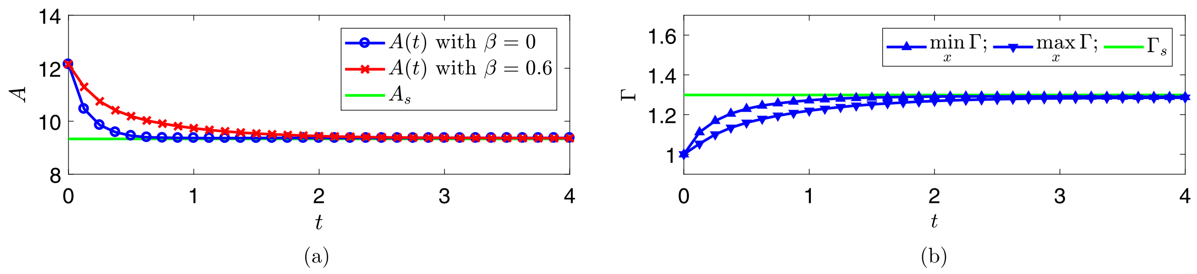

Fig. 7. (a) The time evolutionary surface area plots for both clean ($\beta = 0$) and contaminated ($\beta = 0.6$) interfaces. (b) The time evolutionary plots of minimum and maximum surfactant concentrations along the contaminated interface ($\beta = 0.6$).

total surfactant mass and droplet volume, respectively. One can see that both relative errors are less than 1% for three different mesh resolutions and the orders of magnitude decrease linearly as the mesh is refined. This confirms that the proposed GBPM-IBIM method preserves the total surfactant mass and droplet volume quite well without performing any re-scalings during computations. Since the exact solution is not available in this test, we use the result obtained from the finest mesh ($h = 1/64$) as a reference solution to evaluate the approximately convergent rate. Table 5 shows the maximum norm errors for the velocity components $\mathbf{u} = (u, v, w)$ and their convergence rates. One can see that the results converge slightly better than the first-order.

In order to see the influence of surfactant, it is natural to compare the dynamics between the clean droplet (no surfactant, $\beta = 0$) and the contaminated one (with surfactant, $\beta = 0.6$). Fig. 6 shows the droplet morphology at different times for the cases of clean interface (top, $\beta = 0$) and the one with surfactant (bottom, $\beta = 0.6$) using $h = 1/16$. Since the surfactant reduces the surface tension, the retreating force of the contaminated droplet is significantly weaker than the one without surfactant. As a result, the surfactant retards the droplet evolution to equilibrium shape; see the comparisons at $t = 0.25, 0.5, 1.5$. Nevertheless, at final time $t = 4$, both droplets evolve to the same spherical equilibrium shape eventually. This can also be confirmed by checking the time evolutionary surface area plots as shown in Fig. 7(a), where the steady surface area is $A_s = 4\pi r_s^2 = 4\pi(\frac{3V_0}{4\pi})^{2/3}$. One can also see in Fig. 7(b) that the minimum and maximum values of the surfactant concentration converge to the equilibrium value $\Gamma_s = A_0/A_s$, where A_0 is the initial surface area of the droplet.

5.2. Effect of capillary number

In this test, we study the effect of capillary number on the droplet dynamics under simple shear flow. We put a unit spherical droplet in the center of the computational domain $\Omega = [-4, 4] \times [-8, 8] \times [-4, 4]$ initially and apply the uniform boundary condition $\mathbf{u} = (0, z, 0)$ at $z = \pm 8$. The mesh width is $h = 1/16$, and the time step size is $\Delta t = h/4$. The initial surfactant concentration is chosen to be uniformly distributed $\Gamma_0 = 1$. The dimensionless numbers are chosen as $Re = 0.1$, $\beta = 0.1$, $Pe_s = 1$ but we vary the capillary number $Ca = 0.2, 0.3$ and 0.4. It is known from previous numerical studies such as [21,16] that as the capillary number increases, the droplet is elongated more so the deformation becomes larger. One can measure the deformation by the factor $D = \frac{L-B}{L+B}$, where L and B are the lengths of major and minor axis of the droplet, respectively. Fig. 8 shows the droplet morphology together with surfactant concentration at different times for

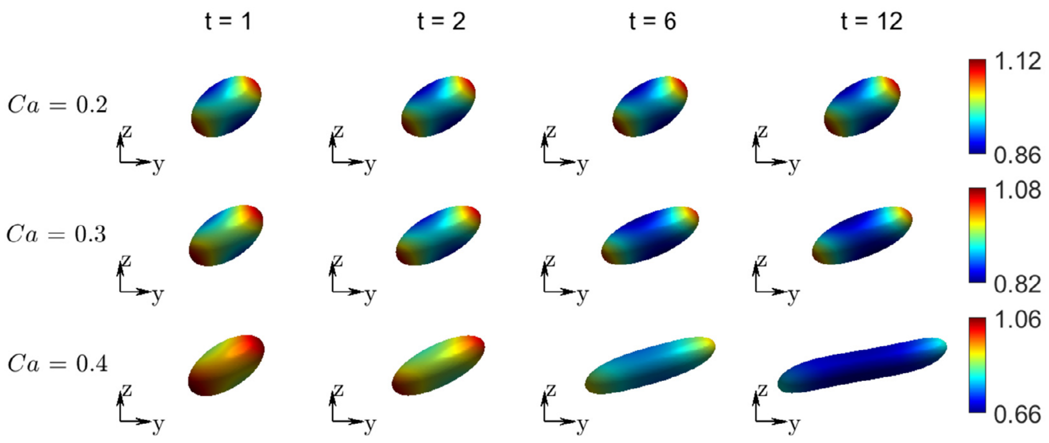

Fig. 8. The droplet morphology together with surfactant concentration at different times for $Ca = 0.2, 0.3, 0.4$.

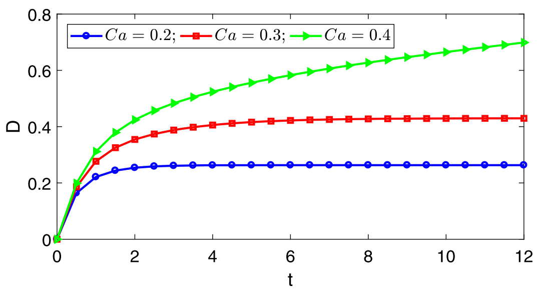

Fig. 9. The time evolutionary plots of the deformation factor for $Ca = 0.2, 0.3, 0.4$.

three different capillary numbers. One can see that for smaller capillary number $Ca = 0.2, 0.3$, the droplet tends to be in equilibrium shape after $t = 6$ while the one with $Ca = 0.4$ elongate more even after $t = 12$. These droplet behaviors have been confirmed by the time evolutionary plots of the deformation factor as shown in Fig. 9.

5.3. Comparison with small deformation theory

As shown in previous test, we know that the droplet under shear flow tends to become an equilibrium shape when the capillary number is small. Therefore, we can compare the present numerical results with the ones obtained from the small deformation theory [29,21] in which the droplet deformation is assumed to be small (smaller Ca) and the surfactant concentration is assumed to be nearly uniform. In such case, the deformation factor is given by

$$D = \frac{5}{8}\frac{35+4\alpha}{20+2\alpha}\frac{Ca}{1-\beta}, \tag{24}$$

where $\alpha = \frac{Pe_s\beta}{Ca}$.

In this simulation, the computational domain setup is the same as the previous test while we choose smaller Reynolds number $Re = 0.01$ and Peclet number $Pe_s = 0.1$. Fig. 10(a) shows the deformation factor D versus the capillary number for the present simulations and the theoretical results obtained from the relation (24). One can immediately see that the present results agree with the theoretical ones pretty well for the cases of $\beta = 0$ (clean interface) and $\beta = 0.5$ (contaminated interface with surfactant) when the capillary number is below 0.2. In addition, one can confirm again that given the same capillary number, the deformation for the case of with surfactant $\beta = 0.5$ is larger than the one of without surfactant.

When the droplet is in equilibrium, the shape does not change and the major axis will align with flow direction and forms an inclination angle θ (the angle between the major axis and the shear flow direction). The flow inside the droplet is rotational. In [3], for the case of the droplet without surfactant, the authors derived the relation between the angle and the capillary number as

$$\theta = \frac{\pi}{4} + \frac{35}{32}Ca. \tag{25}$$

Fig. 10(b) shows the inclination angle θ versus the capillary number Ca for the present simulations and the theoretical results obtained from Eq. (25). Again, one can see that our numerical results are in a good agreement with the theoretical

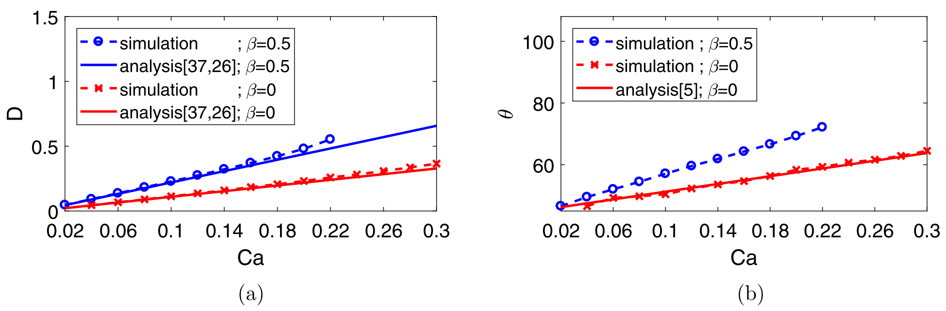

Fig. 10. (a) The deformation factor versus the capillary number for the present simulations and theoretical results in Eq. (24). (b) The inclination angle versus the capillary number for the present simulations and theoretical results in Eq. (25).

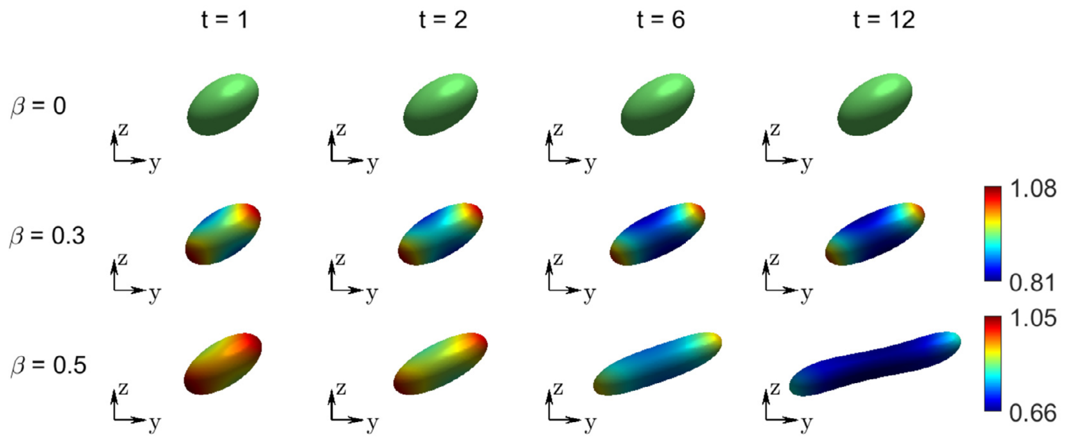

Fig. 11. The droplet morphology together with surfactant concentration at different times for $\beta = 0, 0.3, 0.5$.

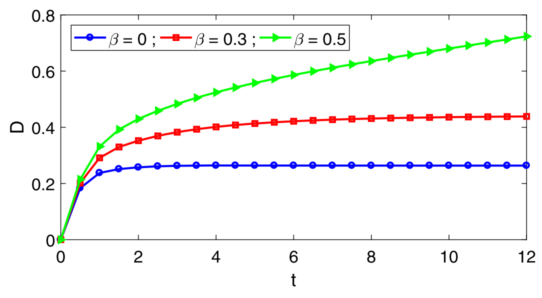

Fig. 12. The time evolutionary plots of the deformation factor for $\beta = 0, 0.3, 0.5$.

ones for the case of $\beta = 0$. Meanwhile, from this figure, one can also see that at given capillary number, the inclination angle for the case of with surfactant $\beta = 0.5$ is greater than the one of without surfactant $\beta = 0$.

5.4. Effect of elasticity number

In this test, we study the effect of elasticity number β on the droplet under shear flow. The computational setup is the same as the second test except we fix the dimensionless numbers $Re = 1, Ca = 0.2, Pe_s = 1$ and vary the elasticity number $\beta = 0, 0.3, 0.5$. From the equation of state Eq. (4), we can see that as β increases, the surface tension reduces more which results larger deformation. This is exactly the case as we can see from Fig. 11 in which the droplet morphology together with surfactant concentration at different times for $\beta = 0, 0.3, 0.5$ are plotted. One can see that for smaller elasticity number $\beta = 0, 0.3$, the droplet tends to be in equilibrium shape after sometime while the one with $\beta = 0.5$ still elongate even after $t = 12$. These droplet behaviors have been confirmed again for the time evolutionary plots of the deformation factor as shown in Fig. 12.

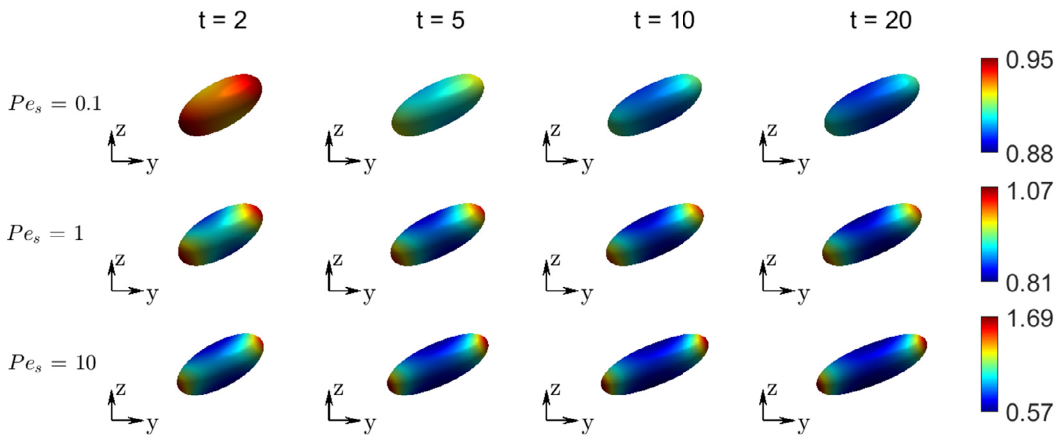

Fig. 13. The droplet morphology together with surfactant concentration at different times for $Pe_s = 0.1, 1, 10$.

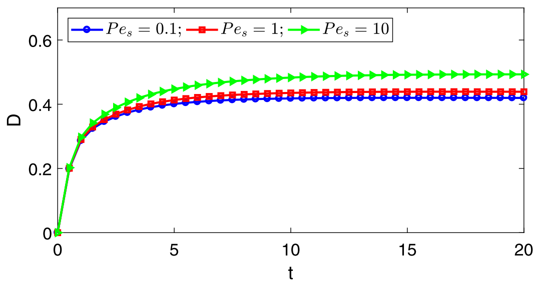

Fig. 14. The time evolutionary plots of the deformation factor for $Pe_s = 0.1, 1, 10$.

5.5. *Effect of Peclet number*

In this test, we study the effect of surface Peclet number Pe_s on the droplet under shear flow. The computational setup is same as the second test except we fix the dimensionless numbers $Re = 1$, $Ca = 0.2$, $\beta = 0.28$ and vary the Peclet number $Pe_s = 0.1, 1, 10$. As mentioned earlier, the Peclet number represents the relative importance between surfactant convection and diffusion in the droplet interface dynamics, so a larger Peclet number indicates the convection is more dominated. Since the initial surfactant concentration is uniformly $\Gamma_0 = 1$ along the interface, the larger Peclet number makes the surfactant distribution more profound as shown in Fig. 13 where the value range of surfactant concentration becomes larger as Pe_s increases. This results in larger droplet deformation as the Peclet number increases which is confirmed in Fig. 14.

5.6. *Interaction of two droplets*

As the last test, we study the interaction of two droplets in a shear flow, under the effect of surfactant concentration as in [34,36]. We consider two unit spherical droplets with the centers positioned at $(0, 1.4, -0.8)$ and $(0, -1.4, 0.8)$ at $t = 0$. The dimensionless parameters are $Re = 0.01$ and $Ca = 0.2$. In order to see the influence of surfactant, we vary the $\beta = 0,\ 0.3$ and $Pe_s = 0.1,\ 10$. The rest of the computation setup is the same as the second test in the previous subsection. The simulation is performed up to the terminal time $t = 4$. Fig. 15 illustrates the time evolution of two droplets and distribution of surfactant concentration for the case of $\beta = 0.3$ and $Pe_s = 10$. One can see that the two droplets deform while approaching each other. Meanwhile, as shown in [24,34,36], the surfactant enhances droplets bouncing and prevents coalescence so the minimum distance between those two contaminated droplets ($\beta = 0.3$) during the approaching process ($1 \le t \le 3$) should be larger compared to the droplets without surfactant ($\beta = 0$). Furthermore, larger surface Peclet number further increases the minimum distance due to the stronger Marangoni effect induced by more nonuniform surfactant distribution. These two droplets interaction behaviors can be confirmed in Fig. 16 in which the minimal distance between two droplets for different cases is plotted.

6. Conclusion

In this paper, we develop a coupled grid based particle and implicit boundary integral method for simulations of three-dimensional interfacial flows with the presence of insoluble surfactant. We use splitting scheme to handle the moving interface as well as the surfactant density that is being convected along. We compare our numerical simulations to some

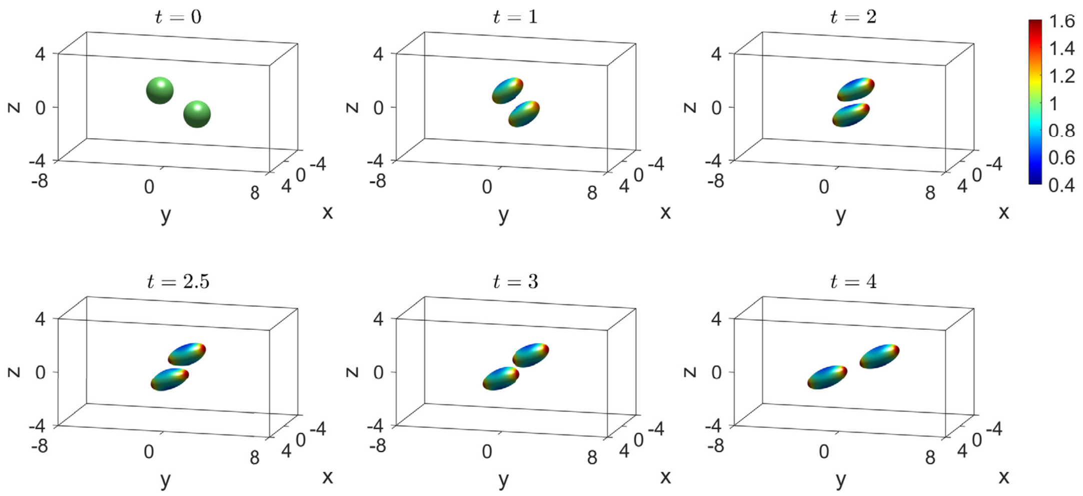

Fig. 15. The two droplets morphology together with surfactant concentration at different times for the case of $\beta = 0.3$ and $Pe_s = 10$.

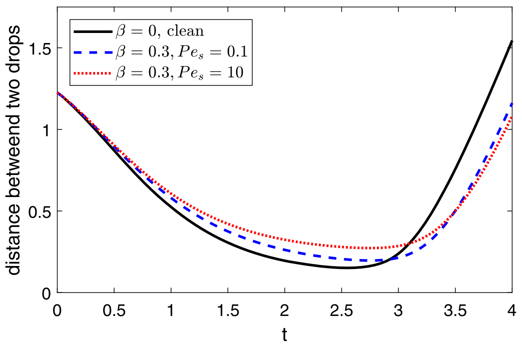

Fig. 16. The time evolutionary plots of the minimal distance between two droplets.

problems for which the analytical solutions are available. The comparison shows that the proposed method obtains the second-order convergence rate as grid refines. For complicated problems, such as a droplet in shear flow, our simulation results highly coincide with theoretical behavior. We plan to generalize the proposed method and code to study droplet dynamics in the presence of surfactant under a DC electric field. As we mentioned in the introduction, our proposed method can be used easily to compute electric potentials involving dielectric droplets in electrohydrodynamic applications.

Acknowledgements

The work of M.-C. Lai is supported in part by Ministry of Science and Technology of Taiwan under research grant MOST-107-2115-M-009-016-MY3 and NCTS. J. Chu's is partially supported by Ministry of Science and Technology of Taiwan under research grant MOST-106-2115-M-007 -002. R. Tsai's research is partially supported by National Science Foundation grant DMS-1720171. Tsai also thanks National Center for Theoretical Sciences (NCTS) for hosting this collaboration.

Appendix A

We briefly introduce a simple way to modify the proposed model and numerical scheme for two fluids with different viscosity. The momentum equation (1) with different viscosity in dimensionless form is modified as

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} + \nabla p = \frac{1}{Re} \nabla \cdot (\mu (\nabla \mathbf{u} + \nabla \mathbf{u}^T)) + \frac{1}{ReCa} \mathbf{f} \quad \text{in } \Omega,$$

where the viscosity

$$\mu(x) = \begin{cases} \mu_r & \text{if } x \in \Omega_1, \\ 1 & \text{if } x \in \Omega_2, \end{cases}$$

is a piecewise constant function.

We also present a simple numerical discretization for Navier-Stokes flows with variable viscosity as follows. A second-order projection method [11] for the Navier-Stokes equations has the form

$$\begin{aligned}
&\frac{3\mathbf{u}^* - 4\mathbf{u}^n + \mathbf{u}^{n-1}}{2\Delta t} + 2\left(\mathbf{u}^n \cdot \nabla_h\right)\mathbf{u}^n - \left(\mathbf{u}^{n-1} \cdot \nabla_h\right)\mathbf{u}^{n-1} + \nabla_h p^n \\
&= \frac{1}{Re}\left(\mu^* \Delta_h \mathbf{u}^* - \mu^* \Delta_h \mathbf{u}^n + \nabla_h \cdot (\mu(\nabla_h \mathbf{u}^n + (\nabla_h \mathbf{u}^n)^T))\right) + \frac{\mathbf{f}^n}{Re\, Ca}, \quad \mathbf{u}^*|_{\partial\Omega} = \mathbf{u}_b, \\
&\Delta_h p^* = \frac{3}{2\Delta t}\nabla_h \cdot \mathbf{u}^*, \quad \frac{\partial p^*}{\partial \mathbf{n}}|_{\partial\Omega} = 0, \\
&\mathbf{u}^{n+1} = \mathbf{u}^* - \frac{2\Delta t}{3}\nabla_h p^*, \quad \nabla_h p^{n+1} = \nabla_h p^* + \nabla_h p^n - \frac{2\mu^* \Delta t}{3Re}\Delta_h(\nabla_h p^*),
\end{aligned}$$

where $\mu^* = \max(1, \mu_r)$. In practice, the viscosity function $\mu(x)$ can be calculated by the harmonic average [32]

$$\frac{1}{\mu(\mathbf{x})} = \frac{H_h(d(\mathbf{x}))}{1} + \frac{1 - H_h(d(\mathbf{x}))}{\mu_r},$$

where H_h is the discrete Heaviside function.

References

[1] John W. Barrett, Harald Garcke, Robert Nürnberg, On the stable numerical approximation of two-phase flow with insoluble surfactant, ESAIM Math. Model. Numer. Anal. 49 (2) (2015) 421–458.
[2] John W. Barrett, Harald Garcke, Robert Nürnberg, Stable finite element approximations of two-phase flow with soluble surfactant, J. Comput. Phys. 297 (2015) 530–564.
[3] C.E. Chaffey, H. Brenner, A second-order theory for shear deformation of drops, J. Colloid Interface Sci. 24 (1967) 258–269.
[4] J. Chu, R. Tsai, Volumetric variational principles for a class of partial differential equations defined on surfaces and curves, Res. Math. Sci. 5 (2018).
[5] A. Demlow, G. Dziuk, An adaptive finite element method for the Laplace-Beltrami operator on implicitly defined surfaces, SIAM J. Numer. Anal. 45 (2007) 421–442.
[6] G. Dziuk, C.M. Elliott, Finite elements on evolving surfaces, IMA J. Numer. Anal. 27 (2007) 262–292.
[7] G. Dziuk, C.M. Elliott, A fully discrete evolving surface finite element method, SIAM J. Numer. Anal. 50 (2012) 2677–2694.
[8] G. Dziuk, C.M. Elliott, Finite element methods for surface PDEs, Acta Numer. 22 (2013) 289–396.
[9] C.M. Elliott, B. Stinner, V. Styles, R. Welford, Numerical computation of advection and diffusion on evolving diffuse interfaces, IMA J. Numer. Anal. 31 (2011) 786–812.
[10] A. Gray, E. Abbena, S. Salamon, Modern Differential Geometry of Curves And Surfaces With Mathematica, CRC Press, 2006.
[11] J.L. Guermond, P. Minev, J. Shen, An overview of projection methods for incompressible flows, Comput. Methods Appl. Mech. Eng. 195 (2006) 6011–6045.
[12] F.H. Harlow, J.E. Welch, Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface, Phys. Fluids. 8 (1965) 2182–2189.
[13] A.J. James, J.S. Lowengrub, A surfactant-conserving volume-of-fluid for interfacial flows with insoluble surfactant, J. Comput. Phys. 201 (2004) 685–722.
[14] C. Kublik, N.M. Tanushev, R. Tsai, An implicit interface boundary integral method for Poisson's equation on arbitrary domains, J. Comput. Phys. 247 (2013) 279–311.
[15] C. Kublik, R. Tsai, Integration over curves and surfaces defined by the closest point mapping, Res. Math. Sci. 3 (2016) 3.
[16] M.-C. Lai, Y.H. Tseng, H. Huang, An immersed boundary method for interfacial flows with insoluble surfactant, J. Comput. Phys. 227 (2008) 7279–7293.
[17] M. Muradoglu, G. Tryggvason, Simulations of soluble surfactants in 3D multiphase flow, J. Comput. Phys. 274 (2014) 737–757.
[18] W.C. de Jesus, A.M. Roma, M.R. Pivello, M.M. Villar, A. da Silveira-Neto, A 3D front-tracking approach for simulation of a two-phase fluid with insoluble surfactant, J. Comput. Phys. 281 (2015) 403–420.
[19] M. Lenz, S. Nemadjieu, M. Rumpf, A convergent finite volume scheme for diffusion on evolving surfaces, SIAM J. Numer. Anal. 49 (2011) 15–37.
[20] S. Leung, H. Zhao, A grid based particle method for moving interface problems, J. Comput. Phys. 228 (2009) 2993–3024.
[21] X. Li, C. Pozrikidis, The effect of surfactants on drop deformation and on the rheology of dilute emulsions in Stokes flow, J. Fluid Mech. 341 (1997) 165–194.
[22] C.B. Macdonald, S.J. Ruuth, The implicit closest point method for the numerical solution of partial differential equations on surfaces, SIAM J. Sci. Comput. 31 (2010) 4330–4350.
[23] A. Petras, S.J. Ruuth, PDEs on moving surfaces via the closest point method and a modified grid based particle method, J. Comput. Phys. 312 (2016) 139–156.
[24] K.-L. Pan, Y.-H. Tseng, J.-C. Chen, K.-L. Huang, C.-H. Wang, M.-C. Lai, Controlling droplet bouncing and coalescence with surfactant, J. Fluid Mech. 799 (2016) 603–636.
[25] S.J. Ruuth, B. Merriman, A simple embedding method for solving partial differential equations on surfaces, J. Comput. Phys. 227 (2008) 1943–1961.
[26] Y. Seol, S.-H. Hsu, M.-C. Lai, An immersed boundary method for simulating interfacial flows with insoluble surfactant in three dimensions, Commun. Comput. Phys. 23 (2018) 640–664.
[27] K.J. Stebe, R. Balasubramaniam, Marangoni effects on drop deformation in an extensional flow: the role of surfactant physical chemistry. I. Insoluble surfactants, Phys. Fluids. 8 (1996).
[28] H.A. Stone, A simple derivation of the time-dependent convective-diffusion equation for surfactant transport along a deforming interface, Phys. Fluids A Fluid Dyn. 2 (1990) 111–112.
[29] H.A. Stone, L.G. Leal, The effects of surfactants on drop deformation and breakup, J. Fluid Mech. 220 (1990) 161.
[30] C. Sorgentone, A.-K. Tornberg, A highly accurate boundary integral equation method for surfactant-laden drops in 3D, J. Comput. Phys. 360 (2018) 167–191.
[31] G. Strang, On the construction and comparison of difference schemes, SIAM J. Numer. Anal. 5 (1968) 506–517.
[32] K.E. Teigen, S.T. Munkejord, Influence of surfactant on drop deformation in an electricfield, Phys. Fluids. 22 (2010) 112104.
[33] J. Xu, H.K. Zhao, An eulerian formulation for solving partial differential equations along a moving interface, J. Sci. Comput. 19 (2003) 573–594.
[34] J. Xu, Z. Li, J. Lowengrub, H. Zhao, Numerical study of surfactant-laden drop-drop interactions, Commun. Comput. Phys. 10 (2011) 453–473.
[35] J. Xu, Y. Yang, J. Lowengrub, A level-set continuum method for two-phase flows with insoluble surfactant, J. Comput. Phys. 231 (2012) 5897–5909.
[36] J. Xu, W. Shi, M.-C. Lai, A level-set method for two-phase flows with soluble surfactant, J. Comput. Phys. 353 (2018) 336–355.