A fictitious domain method with a hybrid cell model for simulating motion of cells in fluid flow

Wenrui Hao, Zhiliang Xu, Chun Liu, Guang Lin

Department of Applied and Computational Mathematics and Statistics, University of Notre Dame, Notre Dame, IN 46556, United States
Department of Mathematics, The Penn State University, University Park, PA 16802, United States
Department of Mathematics, School of Mechanical Engineering, Purdue University, West Lafayette, IN 47907, United States

A R T I C L E   I N F O

Article history:
Received 27 March 2014
Received in revised form 11 August 2014
Accepted 18 September 2014
Available online 28 September 2014

Keywords:
Fictitious domain method
Navier–Stokes flow
Fluid–structure interaction
Level set method
ENO reconstruction
Elastic energy

A B S T R A C T

In this study, we develop a hybrid model to represent membranes of biological cells and use the distributed-Lagrange-multiplier/fictitious-domain (DLM/FD) formulation for simulating the fluid/cell interactions. The hybrid model representing the cellular structure consists of a continuum representation of the lipid bilayer, from which the bending force is calculated through energetic variational approach, a discrete cytoskeleton model utilizing the worm-like chain to represent network filament, and area/volume constraints. For our computational scheme, a formally second-order accurate fractional step scheme is employed to decouple the entire system into three sub-systems: a fluid problem, a solid problem and a Lagrange multiplier problem. The flow problem is solved by the projection method; the solid problem based on the cell model is solved by a combination of level set method, ENO reconstruction, and the Newton method; and the Lagrange multiplier problem is solved by immersed boundary interpolation. The incompressibility of the material is implemented with the penalty function method. Numerical results compare favorably with previously reported numerical and experimental results, and show that our method is suited to the simulation of the cell motion in flow.

© 2014 Elsevier Inc. All rights reserved.

1. Introduction

The study of dynamics of biological cells in fluids is essential for understanding many biological and bio-mechanical processes such as blood leukocytes binding to sites of inflammation or tissue injury [15], flowing red blood cells enhancing platelet transport to blood vessel wall in thrombus formation [69], tumor cell metastasis [66], and bacteria swimming near surfaces to promote biofilm formation in early stages [7,45,68]. With the rapid development of scientific computing, computer simulation has become a practical and important approach for studying these processes. In these processes, cell structures interacting with an internal or surrounding fluid flow is a common phenomenon, which is usually modeled by coupled incompressible Navier–Stokes equations and solid equations. Therefore, simulations of fluid–cell/structure interaction require both an accurate numerical procedure to solve the coupled system of equations, and a reliable cell structure (or solid) model to reproduce cell mechanics correctly.

* Corresponding authors.

E-mail addresses: hao.50@nd.edu (W. Hao), zxiu2@nd.edu (Z. Xu), liu@math.psu.edu (C. Liu), guanglin@purdue.edu (G. Lin).

URL: http://people.mbi.ohio-state.edu/hao.50 (W. Hao).

http://dx.doi.org/10.1016/j.jcp.2014.09.020
0021-9991/© 2014 Elsevier Inc. All rights reserved.
To solve the coupled system of fluid and cell equations, in most situations, one usually tracks the interface representing the membrane of a moving cell and employs some numerical techniques to resolve the interface dynamics properly; while neglecting the internal cell structure by treating interior of a cell as a single fluid [52]. Depending on how an interface is treated, numerical methods for solving moving interface problem can largely be classified into two types:

- The first type is to use a Lagrangian mesh (or markers) to represent the interface and to track the interface movement explicitly. On the interface, the coupled system of equations have to satisfy the explicitly defined boundary condition. Numerical methods, such as the boundary element method (BEM) [4,43,63], immersed boundary method (IBM) [40–42], immersed interface method (IBM) [31–33], fictitious domain method (FDM) [36,46,48,49,74], front tracking method (FTM) [16,27,50,73], and the grid based particle method [54] belong to this category.
- The second type is to capture the interface dynamics by evolving a scalar function defined on the whole domain. Numerical methods, including the level set method [39] and phase field method [10], belong to this category. There are also many other methods [12–14,51,60,75]. See also a recent review paper for these methods [24].

Many discrete or continuum models have been developed to model cell membrane structures. For example, [30] proposed a model of the spectrin network of the red blood cell (RBC). In a series of papers by G. Karniadakis et al. [20,21,44,59], a coarse-grained RBC membrane model was developed and coupled with fluid flow by using the dissipative particle dynamics (DPD) approach to study dynamics of healthy or pathological RBCs. A continuum approach [73] has also been used to model RBC membrane to yield resistance against shear deformation. In [73], the area dilatation is modeled using a strain energy function, and the bending is modeled using Helfrich’s formulation. Several continuum approaches [19,62,63] based on energy variational methods to describe exchange between kinetic energy of fluid and interfacial energy of the vesicle cell membrane have also been developed. [19] utilizes the phase field and incompressible Navier–Stokes equation coupling; while [62,63] employs BEM to couple the interface and the Stokes fluid flow. See also [63] for an overview of methods modeling vesicles.

To the best of our knowledge, there are only a few attempts [28,63,67,76] to utilize the interface tracking approach to simulate dynamics of cells in fluids. In this work, we develop a hybrid cell membrane model for simulating RBCs as well as other types of eukaryotic cells whose membrane structures are similar to that of RBCs, and a fictitious domain approach for solving fluid–cell interaction problems.

The RBC membrane primarily consists of a lipid bilayer and a two-dimensional (2D) cytoskeleton network, which attaches in a sparse manner to the lipid bilayer, and is usually stiffer than the bilayer [56,58]. As pointed out in [58], a consistent description of shape fluctuation spectrum must include the confining effects of the cytoskeleton. Additionally, compared with lipid bending modulus, the curvature bending modulus of the cytoskeleton is negligible. Based on these membrane structure properties, our hybrid cell membrane model uses the continuum model developed by [19] to represent the mechanics of the lipid bilayer, and the discrete approach developed by [17,30,44] to simulate mechanics of cytoskeleton network, respectively. Our simulation results show that this membrane model can predict cell mechanics, rheology, and dynamics in agreement with experiments.

Among the numerical methods for solving coupled fluid–structure system, FDM has an advantage that it allows the use of fairly structured meshes on a simple auxiliary fluid domain (called a fictitious domain) containing the actual one. This enables to use fast solvers for solving the auxiliary fluid problems on complicated geometries. In the FDM, the fluid and solid domains are coupled by introducing a Lagrange multiplier over the solid domain. This multiplier imposes the kinematic constraint, and can be interpreted as the body force to enforce this constraint. The FDM was used for simulating motions of embedded rigid bodies in fluid [36,37], [74] generalized this approach for the general solid material by introducing the continuum equations instead of Newton’s motion equations. In [46] a multiscale fluid–cell interaction and cell–substrate adhesion model was introduced to model the cell coating of artificial surfaces of cardiovascular implants. The cells were modeled as rigid particles and were coupled with flow by the distributed Lagrange multiplier-based FDM.

We utilize the FDM to couple the cell model and the fluid equation together. Instead of using weak formulation, we introduce the strong form due to the new hybrid cell model developed in this work. We also adopt the idea of using the delta function in IBM to interpolate the velocity from the nearby fluid point and to distribute the Lagrange multiplier at each solid point to the nearby fluid point. Our numerical methods are verified by various examples.

The paper is organized as follows: Section 2 describes our cell model and coupled cell–fluid system. The numerical methods are presented in Section 3. In Section 4, several numerical tests are performed to validate the cell model and numerical methods. Used formulas are given in Appendix A.

2. Mathematical model

In this section, we first present our hybrid cell model for modeling RBCs. Then we detail the DLM/FDM for coupling the cell model and fluid equations.
2.1. A hybrid RBC membrane model

Motivated by the works described in [17,19,30,44], we introduce a hybrid cell model to represent the RBC membrane. Specifically, we adopt the molecular based cell cytoskeleton model to describe the RBC cytoskeleton, in which a network of interconnected viscoelastic springs is utilized to provide cell visco-elasticity analogously to that of the spectrin network.

To mimic bending resistance of the lipid bilayer, a continuum approach, i.e., the elastic bending energetic variational approach [19] is employed to simulate the bending resistance of the lipid bilayer. In addition, local and global area constraints are introduced in our model to ensure the membrane incompressibility of real cells, while the volume constraint is employed to ensures the incompressibility of the inner solvent [44].

The combination of the cytoskeleton network and the lipid bilayer model together with the constraints leads to the following total coarse-grained Helmholtz free energy of the cell

\[ V = V_{\text{in-plane}} + \mathcal{S}_j + V_{\text{area}} + V_{\text{volume}}. \]  

(2.1)

Here \( V_{\text{in-plane}} \) is the in-plane energy of the cytoskeleton network; \( \mathcal{S}_j \) is the bending energy representing the bending resistance of the lipid bilayer; and \( V_{\text{area}} \), \( V_{\text{volume}} \) are area, volume conservation constraints, respectively. Required parameters of the RBC model Eq. (2.1) can be analytically related to the membrane macroscopic properties and will be explained in Subsection 2.1.5.

Many electron microscopy results suggest that normal RBC cytoskeleton network tends to have 6 filaments linked to a junction complex (or actin vertex) on the lipid bilayer [5,30,34–36]. To model the RBC membrane cytoskeleton structure and attached lipid bilayer, we use a triangular mesh consisting of \( N \) vertices denoted by \( \{x_i\}_{i=1}^{N} \), each of which corresponds to a collection of atoms or molecules, following the idea introduced in [17,30,44]. These vertices are connected by \( N_e \) edges, denoted by \( \{l_{ij}\}_{j=1}^{N_e} \), forming \( N_t \) triangles \( \bigcup_{i=1}^{N_t} \). Each edge \( l_{ij} \) represents a coarse-grained filament, and the triangulized mesh surface

\[ \Gamma = \bigcup_{i=1}^{N_t} \Gamma_i, \]

approximates the lipid bilayer of the cell membrane, where the cytoskeleton attaches to.

2.1.1. The cytoskeleton model

A coarse-grained filament \( l_{ij} \) is modeled by a worm-like chain (WLC), for which the associated in-plane free energy \( V_{\text{WLC},j} \) is defined as the sum of the WLC potential and a repulsive potential:

\[ V_{\text{WLC},j} = \frac{k_B T l_{\text{max}}}{4p} \left( \frac{3x_j^2}{1 - x_j} - 2x_j \right) - \frac{k_p}{(m - 1)l_{ij}^{m-1}}, \]

(2.2)

where \( x_j = l_j/l_{\text{max}} \), \( l_j \) is the length of the current spring, \( l_{\text{max}} \) is the maximum spring extension, \( p \) is the persistence length, \( k_B T \) is the energy unit, \( k_p \) is the spring constant, and \( m \) is a power coefficient. The first term in Eq. (2.2) is the attractive WLC potential, and the second term defines a repulsive potential for \( m > 0 \) [20], whose force is called the power force (POW), and is introduced to mimic the ability of the network to sustain weak compression. We abbreviate the spring model as WLC-POW. The in-plane energy \( V_{\text{in-plane}} \) is the sum of energies of individual filaments and is given by

\[ V_{\text{in-plane}} = \sum_{j=1}^{N_t} V_{\text{WLC},j}. \]

(2.3)

2.1.2. The lipid bilayer model

We assume that the lipid bilayer of the membrane is homogeneous, and use the bending elasticity (Willmore, mean curvature square) energy [19] to represent the bending resistance of the lipid bilayer.

Let \( \Sigma \subset \mathbb{R}^3 \) be a smooth, closed surface representing the lipid bilayer of the cell membrane. The quasi-static deformation and equilibrium configuration of a bilayer is mainly characterized by its interfacial energy [19]:

\[ \mathcal{S}_j(\Sigma) = k_0 \int_{\Sigma} \frac{1}{2} (H(x) - \kappa(x))^2 dS(x) + k_1 \int_{\Sigma} K(x) dS(x), \]

(2.4)

where \( \mathcal{S}_j(\Sigma) \) is called the Helfrich bending elasticity energy of the surface \( \Sigma \), and \( \kappa(x) \) denotes the spontaneous curvature at a point \( x \in \Sigma \). \( H(x) \) is the mean curvature defined by \( H(x) = \frac{1}{2}(\kappa_1(x) + \kappa_2(x)) \); and the \( K(x) \) is the Gaussian curvature defined by \( K(x) = \kappa_1(x)\kappa_2(x) \). \( \kappa_1(x) \) and \( \kappa_2(x) \) are the principle curvatures at a point \( x \in \Sigma \). \( k_0 \) and \( k_1 \) are two bending rigidities.
2.1.3. Area and volume constraints

The area and volume conservation constraints, which account for area incompressibility of the lipid bilayer and incompressibility of the inner cytosol, respectively, are expressed as

\[ V_{\text{area}} = \frac{k_s(S^{\text{total}} - S_0^{\text{total}})^2}{2S_0^{\text{total}}} + \frac{\sum_{j=1}^{N_t} k_l(S_j - S_{0,j})^2}{2S_{0,j}}, \quad (2.5) \]

\[ V_{\text{volume}} = \frac{k_v(V_c - V_0)^2}{2V_0}, \quad (2.6) \]

where \( S \) and \( V_c \) denote the surface area and volume, respectively. \( k_s, k_l \) and \( k_v \) are the global area, local area, and volume constraint coefficients, respectively. The terms \( S \) and \( V_c \) denote the surface area and volume, while \( S_{0,j} \) and \( S^{\text{total}} \) are the desired area of \( j \)th triangle and the desired area of the whole membrane, respectively.

2.1.4. Membrane force calculation

A cell deforms under the flow field, thereby changing the energy (2.1). Since Eq. (2.1) consists of terms derived from discrete and continuous descriptions of components of the membrane respectively, forces at mesh points \( \{x_i\}_{i=1}^N \) are derived correspondingly.

The normal component of the elastic force associated with Helfrich energy (2.4) can be obtained by taking variational derivative and is as follows [61,19]:

\[ \mathbf{F}_{\text{bend}}(\mathbf{x}) = k_{bend}(\Delta \Sigma(H - \kappa) + 2(H - \kappa)(H^2 - K + H\kappa))\mathbf{n}, \quad (2.7) \]

where \( \mathbf{n} \) is the outward pointing unit normal of \( \Sigma \) at the point \( \mathbf{x} \in \Sigma \), and \( \Delta \Sigma \) is the surface Laplacian.

Nodal forces associated with rest of energy terms are derived analytically as follows:

\[ \mathbf{F}_d(\mathbf{x}_i) = \frac{\partial (V_{\text{in-plane}} + V_{\text{area}} + V_{\text{volume}})}{\partial \mathbf{x}} \bigg|_{\mathbf{x}_i}. \quad (2.8) \]

Therefore the total force exerting on node \( \mathbf{x}_i \) is defined as

\[ \mathbf{F}_{\text{total}}(\mathbf{x}_i) = \mathbf{F}_d(\mathbf{x}_i) + \mathbf{F}_{\text{bend}}(\mathbf{x}_i). \quad (2.9) \]

2.1.5. Determination of parameter values of the model

The cell membrane model consists of parameters such as \( K_B, k_s, k_l, k_v, k_0 \) and \( k_1 \). Values of these parameters are initially taken from [19] and [44] for using this model to simulate dynamics of RBCs. Then numerical experiments of cell stretching are performed (see Subsection 4.3), and are compared with data of the optical tweezers experiments in [38]. By performing this numerical experiments, the values of parameters are slightly adjusted to archive a good agreement with the experimental data.

2.2. The governing equations

We briefly explain the equations governing fluid–structure interaction and their boundary conditions in this section. The fictitious-domain DLM method coupling these equations will be described in detail in Subsection 2.3.

Let \( \Omega(t), \Gamma_{s1}(t) \) and \( \Gamma_{s2}(t) \) represent the solid domain, its interior and exterior boundaries at time \( t \). \( \Omega \) is the entire domain, and \( \Gamma^c \) denotes the boundary of \( \Omega \), respectively. A 2D schematic diagram is shown in Fig. 1.
2.2.1. Fluid and solid motion equations

Let $\mathbf{u}_f$, $p$, $\rho_f$, $\sigma_f$, $\mathbf{D}$ and $\mathbf{I}$ denote the fluid velocity, pressure, density, stress tensor, rate of deformation tensor and unit tensor, respectively. The fluid field is governed by the incompressible Navier–Stokes equations. The unsteady momentum equation is

$$
\rho_f \frac{d\mathbf{u}_f}{dt} = \nabla \cdot \mathbf{f} \quad \text{in } \Omega \setminus \Omega_s(t),
$$

(2.10)

$$
\sigma_f = -p\mathbf{I} + \eta\mathbf{D} \quad \text{in } \Omega \setminus \Omega_s(t).
$$

(2.11)

The continuity equation is

$$
\nabla \cdot \mathbf{u}_f = 0 \quad \text{in } \Omega \setminus \Omega_s(t).
$$

(2.12)

For convenience of exposition, the Dirichlet boundary condition is imposed on the outer boundary $\Gamma$,

$$
\mathbf{u}_f = \mathbf{u}_f(t) \quad \text{on } \Gamma, \quad t \geq 0,
$$

(2.13)

and the initial condition for fluid is given by

$$
\mathbf{u}_f(t = 0) = \mathbf{u}_f(0) \quad \text{in } \Omega \setminus \Omega_s(0).
$$

(2.14)

The flexible body is suspended in the fluid so that the no-slip and the equal normal stress boundary conditions are imposed on the entire solid surface, namely,

$$
\mathbf{u}_f = \mathbf{u}_s \quad \text{on } \Gamma_{s,i}, \quad i = 1, 2,
$$

$$
\mathbf{b}_i + \sigma_f \cdot \mathbf{n} = \sigma_s \cdot \mathbf{n} \quad \text{on } \Gamma_{s,i}, \quad i = 1, 2,
$$

(2.15)

where $\mathbf{b}_i$ denotes the nonhydrodynamic force (density) on the solid surface such as external surface forces or contact forces between either two different bodies or between two different parts of the same (flexible) body (see [25,72]). $\mathbf{n}$ is the normal unit vector on the solid surface pointing towards the fluid.

The equation of motion for a solid is expressed as follows

$$
\rho_s \frac{d\mathbf{u}_s}{dt} = \nabla \cdot \mathbf{f}_s \quad \text{in } \Omega_s(t),
$$

(2.16)

where $\mathbf{u}_s$ denotes the solid velocity, $\rho_s$ the solid density, and $\mathbf{f}_s$ Cauchy stress tensor at $\mathbf{x}$.

Finally, the kinematic equations of the solid are given by

$$
\frac{d\mathbf{x}}{dt} = \mathbf{u}_s \quad \text{in } \Omega_s(t),
$$

$$
\mathbf{x}(t = 0) = \mathbf{x}_0 \quad \text{in } \Omega_s(0).
$$

(2.17)

2.3. Fictitious domain method

We will derive the FD/DLM formulation for a general solid material, and then present the numerical scheme for solving the fluid–cell interaction system governed by Eqs. (2.10)–(2.17).

Let’s define the following combined velocity space:

$$
S_u = \{ (\mathbf{u}_f, \mathbf{u}_s) \mid \mathbf{u}_f \in H^1(\Omega \setminus \Omega_s)^2, \mathbf{u}_s = \mathbf{u}_f \text{ on } \Gamma ; \mathbf{u}_s \in H^1(\Omega_s)^2 \},
$$

and corresponding test function space:

$$
S_v = \{ (\mathbf{v}_f, \mathbf{v}_s) \mid \mathbf{v}_f \in H^1(\Omega \setminus \Omega_s)^2, \mathbf{v}_f = 0 \text{ on } \Gamma ; \mathbf{v}_s \in H^1(\Omega_s)^2 \}.
$$

By adding weak forms of (2.10) and (2.16), we obtain

$$
\left( \rho_f \frac{d\mathbf{u}_f}{dt} - \nabla \cdot \mathbf{f}_f, \mathbf{v}_f \right) + \left( \rho_s \frac{d\mathbf{u}_s}{dt} - \nabla \cdot \mathbf{f}_s, \mathbf{v}_s \right) = 0 \quad \forall (\mathbf{v}_f, \mathbf{v}_s) \in S_v,
$$

(2.18)

where $(\cdot, \cdot)$ denotes the $L^2$ inner product. Applying integration by parts, we rewrite (2.18) as follows

$$
\int_{\Omega \setminus \Omega_s} \rho_f \frac{d\mathbf{u}_f}{dt} \cdot \mathbf{v}_f \, d\mathbf{x} + \int_{\Omega \setminus \Omega_s} \mathbf{f}_f : \nabla \mathbf{v}_f \, d\mathbf{x} + \int_{\Omega_s} \rho_s \frac{d\mathbf{u}_s}{dt} \cdot \mathbf{v}_s \, d\mathbf{x} + \int_{\Omega_s} \mathbf{f}_s : \nabla \mathbf{v}_s \, d\mathbf{x} = \int_{\Gamma_{s1}} \mathbf{b}_1 \cdot \mathbf{v}_s \, d\mathbf{x} - \int_{\Gamma_{s2}} \mathbf{b}_2 \cdot \mathbf{v}_s \, d\mathbf{x}.
$$

(2.19)

In order to construct a fictitious domain formulation, we extend the fluid computational domain from $\Omega \setminus \Omega_s$ to $\Omega$, and couple the velocities $\mathbf{u}_f$ and $\mathbf{u}_s$ on the entire solid domain $\Omega_s$. The following spaces are defined by modifying the trial and test spaces $S_u$ and $S_v$:
\[ \tilde{S}_u = \{ (u_f, u_s) \mid u_f \in H^1(\Omega)^2, u_s \in H^1(\Omega_s)^2, u_f = u_s in \Omega_s, u_f = u_f^r on \Gamma \}, \]

\[ \tilde{S}_v = \{ (v_f, v_s) \mid v_f \in H^1(\Omega)^2, v_s \in H^1(\Omega_s)^2, v_f = v_s in \Omega_s, v_f = 0 on \Gamma \}. \]

Since

\[ \int_{\Omega_s} \rho_f \frac{du_f}{dt} \cdot (v_f - v_s) \, dx + \int_{\Omega_s} \sigma_f : \nabla(v_f - v_s) \, dx = 0 \quad \forall (v_f, v_s) \in \tilde{S}_v, \quad (2.20) \]

we obtain the following equation by adding (2.19) and (2.20):

\[ \int_{\Omega} \rho_f \frac{du_f}{dt} \cdot v_f \, dx + \int_\Omega \sigma_f : \nabla v_f \, dx + \int_{\Omega_s} (\rho_s - \rho_f) \frac{du_s}{dt} \cdot v_s \, dx + \int_{\Omega_s} (\sigma_s - \sigma_f) : \nabla v_s \, dx \]

\[ = \int_{\Gamma_{12}} b_2 \cdot v_s \, dx - \int_{\Gamma_{11}} b_1 \cdot v_s \, dx \quad \forall (v_f, v_s) \in \tilde{S}_v. \quad (2.21) \]

The weak form of the fluid continuity equation (2.12) in \( \Omega \) is

\[ \int_{\Omega} q \nabla \cdot u_f \, dx = 0, \quad \forall q \in L^2_0(\Omega), \quad (2.22) \]

where

\[ L^2_0(\Omega) = \{ q \in L^2(\Omega) : \int_{\Omega} q \, dx = 0 \}. \]

These weak form equations (2.21)-(2.22) form a complete set of FD based governing equations.

In this study, we decouple Eq. (2.21) into a fluid equation and a solid equation by using a distributed Lagrange multiplier (DLM) \( \lambda \in H^1(\Omega_s)^2 \).

Let’s define

\[ \tilde{S}_u = \{ u_f \mid u_f \in H^1(\Omega)^2, u_f = u_f^r on \Gamma \}, \]

\[ \tilde{S}_v = \{ v_f \mid v_f \in H^1(\Omega)^2, v_f = 0 on \Gamma \}. \]

Then Eq. (2.21) results in the following DLM/FD based momentum equations for \( u_f \in \tilde{S}_u, u_s \in H^1(\Omega_s)^2, \) and \( \lambda \in H^1(\Omega_s)^2 \):

\[ \int_{\Omega} \rho_f \frac{du_f}{dt} \cdot v_f \, dx + \int_{\Omega} \sigma_f : \nabla v_f \, dx = (\lambda, v_f), \quad \forall v_f \in \tilde{S}_v, \quad (2.23) \]

\[ \int_{\Omega} (\rho_s - \rho_f) \frac{du_s}{dt} \cdot v_s \, dx + \int_{\Omega} (\sigma_s - \sigma_f) : \nabla v_s \, dx = \int_{\Gamma_{12}} b_2 \cdot v_s \, dx - \int_{\Gamma_{11}} b_1 \cdot v_s \, dx - (\lambda, v_s), \quad \forall v_s \in H^1(\Omega_s)^2, \quad (2.24) \]

\[ (u_f - u_s, \xi) = 0, \quad \forall \xi \in H^1(\Omega_s)^2, \quad (2.25) \]

where \( \xi \) is the variance of the DLM \( \lambda \), and \((\cdot, \cdot)\) is defined as the \( L^2 \) inner product.

2.4. Dimensionless governing equations for Newtonian fluid and cellular solid

Now we consider cases in which gravity and the non-hydrodynamic force acting on the solid surface are absent. For convenience, we write the dimensionless formulation for the incompressible solid immersed in a Newtonian fluid by introducing the following scales: \( L_c \) for length, \( U_c \) for velocity, \( L_c/U_c \) for time, \( \rho_f U_c^2 \) for the pressure \( p \), and \( \rho_f U_c^2/L_c^2 \) for the Lagrange multiplier.

Eqs. (2.23)-(2.25) then become

\[ \int_{\Omega} \left( \frac{du_f}{dt} + u_f \cdot \nabla u_f \right) \cdot v_f \, dx + \int_{\Omega} \left( -p I + \frac{1}{Re} \left( \nabla u_f \right)^T \right) : \nabla v_f \, dx = (\lambda, v_f), \]

\[ \int_{\Omega} q \nabla \cdot u_f = 0, \quad (2.26) \]
\[ \int_{\Omega_3} (\rho_f - 1) \frac{d\mathbf{u}_f}{dt} \cdot \mathbf{v}_s d\mathbf{x} + \int_{\Omega_2} (\nabla \mathbf{u}_s)^T : \mathbf{\sigma}_s d\mathbf{x} - \int_{\Omega_3} (\nabla \mathbf{v}_s)^T \left[ -p \mathbf{I} + \frac{1}{\text{Re}} (\nabla \mathbf{u}_f + (\nabla \mathbf{u}_f)^T) \right] d\mathbf{x} \]

\[ = \int_{\Gamma_{12}} \mathbf{b}_2 \cdot \mathbf{v}_s d\mathbf{x} - \int_{\Gamma_{11}} \mathbf{b}_1 \cdot \mathbf{v}_s d\mathbf{x} - (\mathbf{\lambda}, \mathbf{v}_s), \]

\[ \int_{\Omega_3} (\mathbf{u}_f - \mathbf{u}_s) \cdot \mathbf{\xi} d\mathbf{x} = 0, \]

\[ (\rho_f - 1) \frac{d\mathbf{u}_f}{dt} - \nabla \cdot \mathbf{\sigma}_s - \nabla p + \frac{1}{\text{Re}} \nabla^2 \mathbf{u}_f = -\mathbf{\lambda}, \quad \text{in } \Omega_3. \]

\[ \mathbf{u}_f - \mathbf{u}_s = 0, \quad \text{in } \Omega_3. \]

where dimensionless parameters are introduced: density ratio: \( \rho_r = \rho_s / \rho_f \), and Reynolds number \( \text{Re} \).

Since finite difference method will be employed to solve the whole system (2.26)–(2.28), we deduce their strong forms as follows

\[ \left\{ \begin{align*}
\frac{d\mathbf{u}_f}{dt} + \mathbf{u}_f \cdot \nabla \mathbf{u}_f + \nabla p - \frac{1}{\text{Re}} \nabla^2 \mathbf{u}_f &= \mathbf{\lambda}, \quad \text{in } \Omega, \\
\nabla \cdot \mathbf{u}_f &= 0, \quad \text{in } \Omega,
\end{align*} \right. \]

\[ (\rho_f - 1) \frac{d\mathbf{u}_f}{dt} - \nabla \cdot \mathbf{\sigma}_s - \nabla p + \frac{1}{\text{Re}} \nabla^2 \mathbf{u}_f = -\mathbf{\lambda}, \quad \text{in } \Omega_3. \]

\[ \mathbf{u}_f - \mathbf{u}_s = 0, \quad \text{in } \Omega_3. \]

Generally, the thickness of the cell membrane, which is on the order of nanometers, is much smaller than the spacing size of the grid used to solve Eq. (2.29). In this work, we treat the cell membrane \( \Omega_3 \) as a zero thickness shell and use the Kirchhoff beam theory [29] to convert the Cauchy stress tensor to scalar force components.

To this end, instead of solving Eq. (2.30), we solve

\[ (\rho_f - 1) \frac{d\mathbf{u}_f}{dt} - \mathbf{F}_{\text{total}} - \nabla p + \frac{1}{\text{Re}} \nabla^2 \mathbf{u}_f = -\mathbf{\lambda}, \quad \text{on } \Gamma_3, \]

(2.32)

together with Eqs. (2.29) and (2.31). Here \( \mathbf{F}_{\text{total}} \) is described by Eq. (2.9), \( \Gamma_3 \) is the cell membrane, or the fluid–solid interface. Since the density of cell membrane is 1.18 g/ml [71], the density of fluid is assumed to be 1 g/ml, we take \( \rho_r = 1.18 \) in our simulation.

3. Numerical method

In this section, we devise a simple and efficient computational scheme to solve the coupled system defined by Eqs. (2.29), (2.31) and (2.32). The level set method and essentially non-oscillatory (ENO) reconstruction are employed to compute curvature of the membrane, the surface Laplacian of the curvature for calculating elastic bending force defined by Eq. (2.7).

3.1. Splitting scheme

We employ a second-order accurate fractional step scheme to solve the fluid system. Eq. (2.29) is discretized as follows:

\[ \left\{ \begin{align*}
\frac{\mathbf{u}^n_j - \mathbf{u}^n_f + \mathbf{u}^n_s - \mathbf{u}^{n-1}_j}{\Delta t} + \mathbf{u}^n_f \cdot \nabla \mathbf{u}^n_f + \nabla p^n - \frac{1}{2 \text{Re}} \nabla^2 (\mathbf{u}^n_f + \mathbf{u}^n_s) &= \lambda^{n+1} - \lambda^n + \lambda^n, \\
\nabla \cdot \mathbf{u}^n_s &= 0
\end{align*} \right. \]

in which the momentum equation is split into the following two equations:

\[ \frac{\mathbf{u}^n_f - \mathbf{u}^n_s}{\Delta t} + \mathbf{u}^n_f \cdot \nabla \mathbf{u}^n_f + \nabla p^n - \frac{1}{2 \text{Re}} \nabla^2 (\mathbf{u}^n_f + \mathbf{u}^n_s) = \lambda^n, \]

\[ \frac{\mathbf{u}^{n+1}_f - \mathbf{u}^n_f}{\Delta t} = \lambda^{n+1} - \lambda^n. \]

Thus system (Eqs. (2.29), (2.31) and (2.32)) is decoupled into the following three sub-problems and discretized as follows:

- Fluid problem for updating \( \mathbf{u}^n_f \) and \( p \):

\[ \frac{\mathbf{u}^n_f - \mathbf{u}^n_f}{\Delta t} + \mathbf{u}^n_f \cdot \nabla \mathbf{u}^n_f + \nabla p^n - \frac{1}{2 \text{Re}} \nabla^2 (\mathbf{u}^n_f + \mathbf{u}^n_s) = \lambda^n, \]

\[ \nabla \cdot \mathbf{u}^n_f = 0, \]

(3.33)
Solid problem for solving $\mathbf{x}^{n+1}$:

\[
(\rho_t - 1) \frac{\mathbf{x}^{n+1} - \mathbf{x}^n - 2\mathbf{x}^n}{\Delta t^2} - \mathbf{F}_{\text{total}}(\mathbf{x}^{n+1}) - \nabla p^n + \frac{1}{\text{Re}} \nabla^2 \mathbf{u}_f^n = -\lambda^n. \tag{3.34}
\]

- Lagrange multiplier problem for computing $\mathbf{u}_f^{n+1}$ and $\lambda^{n+1}$:

\[
\lambda^{n+1} = \frac{\mathbf{u}_f^{n+1} - \mathbf{u}_f^n}{\Delta t} + \lambda^n, \quad \text{on } I_s^{n+1},
\]

\[
\mathbf{u}_f^{n+1} = \frac{\mathbf{x}^{n+1} - \mathbf{x}^n}{\Delta t}, \quad \text{on } I_s^{n+1}. \tag{3.36}
\]

In the following Sections 3.2, 3.3 and 3.4, we detail numerical methods for solving these three sub-problems.

3.2. Solving the fluid problem

We employ a projection scheme using staggered grid to solve Eq. (3.33) (see [2] for more detail about the projection scheme). Specifically, Eq. (3.33) is solved by the following steps:

- Solving Navier–Stokes momentum equation for $\mathbf{u}_f^p$:

\[
\frac{\mathbf{u}_f^p - \mathbf{u}_f^n}{\Delta t} + (\mathbf{u}_f \cdot \nabla \mathbf{u}_f)^n = \frac{1}{2\text{Re}} \left( \nabla^2 \mathbf{u}_f^p + \nabla^2 \mathbf{u}_f^n \right) - \nabla p^n + \lambda^n, \quad \text{with } \mathbf{u}_f^n = \mathbf{u}_f^0 \quad \text{on } \Gamma. \tag{3.37}
\]

- Solving the Poisson equation for $\phi$:

\[
\nabla^2 \phi = \frac{\nabla \cdot \mathbf{u}_f^p}{\Delta t}, \quad \text{with } \frac{\partial \phi}{\partial n} = 0 \quad \text{on } \Gamma, \tag{3.38}
\]

- Computing $\mathbf{u}_f^p$ and the fluid pressure $p^{n+1}$:

\[
\mathbf{u}_f^p = \mathbf{u}_f^n - \Delta t \nabla \phi, \quad \text{with } \mathbf{u}_f^n = \mathbf{u}_f^0 \quad \text{on } \Gamma
\]

\[
p^{n+1} = p^n + \phi \tag{3.39}
\]

In order to speed up our computation, we use algebraic multi-grid method in PETSc [3] to solve the Poisson equation (3.38) (see Appendix A for more details).

3.3. Solving the solid problem

We use Newton–Krylov method in KINSol [23] to solve the nonlinear solid equation (3.34) for $\mathbf{x}^{t+1}$. Nodal forces are explicitly computed by Eq. (2.9). We will explore calculation of the bending force in this section. The main difficulty of evaluating $\mathbf{F}_{\text{bend}}(\mathbf{x})$ defined by Eq. (2.7) lies in computing surface Laplacian of cell membrane curvature, which involves approximating high-order derivatives at each nodes on the membrane. Curvature can be computed numerically by locally fitting a polynomial function [50]. However, it introduces strong oscillations when computing surface Laplacian of curvature by taking the 4th order derivatives of the polynomial function. To overcome this difficulty, we employ a combination of the level set method [39,64] and ENO reconstruction [65] for computing both curvature and its surface Laplacian.

To summarize our approach, we first construct a level set function on the whole domain $\Omega$ with the membrane surface being the zeroth level of this level set function; and compute first and second derivatives of the level set function by using ENO reconstruction to interpolate the Gaussian and mean curvatures of the cell membrane on finite difference grid points in vicinity of the cell membrane. We then calculate the surface Laplacian by computing the second derivatives of curvatures on grid points and interpolate back to the nodes of the membrane by quadratic interpolation.

The level set function, describing the fluid–solid interface, $\Gamma_s$, is defined implicitly as the zero level set of a distance function which we will denote as $d$; therefore we write

\[
\Gamma_s = \{ \mathbf{x} \mid d(\mathbf{x}, t) = 0 \}.
\]

3.3.1. Constructing the level sets $d(\mathbf{x}, t)$

Let $T_i$ be a triangle of $\Gamma_s$ and $A$ is a grid point of the domain $\Omega$. As shown in Fig. 2, we first construct a rotated principal coordinate frame with the origin at one vertex of $T_i$. The $z$ axis is aligned with the surface normal and $x$ axis is chosen as one edge. Then after mapping the point $A$ into the new coordinate system, the distance of $A$ to the triangle can be defined as
Algorithm 1: Algorithm for computing the bending force.

\begin{itemize}
  \item Generate the distance function \( d \) for each grid point on the domain \( \Omega \);
  \item Compute the first and second derivatives of the distance function by using ENO reconstruction, and interpolate the Gaussian curvature \( K \), mean curvature \( H \) and normal vector \( \mathbf{n} \) on the interface \( \Gamma_i \) using bilinear formula;
  \item Compute the surface Laplacian of the mean curvature by using ENO reconstruction, and interpolate the surface Laplacian on the interface \( \Gamma_i \) by using bilinear formula.
\end{itemize}

Fig. 2. The rotated principal coordinate frame: (a) the projection of a point \( A \) to \( x-y \) plane, \( B \), is inside \( \Gamma_i \); (b) the projection of a point \( A \) to \( x-y \) plane, \( B \), is outside \( \Gamma_i \).

\[
\text{dist}(A \text{ to } \Gamma_i) = \begin{cases} 
  z_A & \text{if } B \text{ is inside the triangle (Fig. 2(a))} \\
  \sqrt{z_C^2 + |BC|^2} & \text{otherwise (Fig. 2(b)).}
\end{cases}
\]

where \( z_A \) is the \( z \) coordinate of \( A \) in the new coordinate system, \( B \) is the projection of point \( A \) to \( x-y \) plane, \( C \) is the point with shortest distance to \( B \) in \( \Gamma_i \). Then the distance of \( A \) to the interface \( \Gamma_5 \) is defined as shown in Fig. 2.

The level set function, namely, distance function of a point on the domain \( \Omega \), is the minimum distance of this point to all the triangles on the \( \Gamma_i \).

\[
\text{dist}(A \text{ to } \Gamma_5) = \min_i \text{dist}(A \text{ to } \Gamma_i)
\]  \hspace{1cm} (3.40)

3.3.2. Computing mean curvature of interface and its surface Laplacian

Due to the need to evaluate the fourth derivative of \( d \) required by computing the surface Laplacian of curvature (see Appendix A for more detail), we employ the ENO reconstruction to compute the needed numerical derivatives. ENO reconstruction is motivated by the idea of “adaptive stencil” to use the smoothest piece of numerical solution if possible. Fifth order and third order ENO reconstructions are employed to computed curvatures of interface, and surface Laplacian respectively. In this case, we obtain the first order accuracy for the whole computation. Briefly, the adaptive selection of the stencil is as follows: assume \( V(x) \) is a grid function and the \( j \)-th divided difference can be computed as

\[
V[x_i, \cdots, x_{i+j}] = \frac{V[x_{i+1}, \cdots, x_{i+j}] - V[x_i, \cdots, x_{i+j-1}]}{x_{i+j} - x_i}.
\]

If \( |V[x_i, \cdots, x_{i+j}]| < |V[x_{i-1}, \cdots, x_{i+j-1}]| \), we will take the stencil as \( \{x_i, \cdots, x_{i+j}\} \); otherwise, we will take \( \{x_{i-1}, \cdots, x_{i+j-1}\} \). This procedure can be continued, with one point added to the stencil at each step, according to the smaller of the absolute values of the two relevant divided differences, until the desired number of points in the stencil is reached.

Finally, we state Algorithm 1 based on level sets and ENO reconstruction for computing the bending force.

3.4. Solving the Lagrange multiplier problem

The DLM \( \lambda_n^{i+1} \) on the interface \( \Gamma_i \) is computed using Eq. (3.35), and subsequently is distributed onto the fluid grid by using the immersed boundary interpolation technique, namely:

\[
\lambda_{\text{fluid}}^{i+1} = \int_{\Omega_i} \lambda_n^{i+1} \delta(x_f - x_s) \, dx_s,
\]

where \( x_f - x_s \) is the distance from any point \( x_f \) in \( \Omega \) to the solid grid point \( x_s \), and the delta function \( \delta(x) \) is approximated as
Algorithm 2: Numerical algorithms.

- Assuming that \( u_i^n \), \( \lambda^n \) at the \( n \)-th time step are given, we calculate \( u_i^{n+1} \) from (3.37);
- Solve the Poisson equation (3.38) by using multi-grid method and get \( \phi \);
- Update \( u_i^n \) and compute \( p^{n+1} \) from (3.39);
- Solve the solid equation (3.34) by using Newton–Krylov method, and get \( x_n \) and \( u_i^{n+1} \). Compute the WLC, volume, surface and bending forces for each iteration.
- Compute \( \lambda^{n+1} \) from (3.35)–(3.36), and interpolate to the fluid mesh using the interpolation technique of the immersed boundary method.
- Compute \( u_i^{n+1} \) from (3.36).

### Table 1
Accuracy of fluid solver.

<table>
<thead>
<tr>
<th>( M )</th>
<th>( \text{Re} = 10 )</th>
<th>( \text{Re} = 20 )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( L_2 ) error</td>
<td>order</td>
</tr>
<tr>
<td>20</td>
<td>1.09e−1</td>
<td>1.21e−1</td>
</tr>
<tr>
<td>40</td>
<td>3.39e−2</td>
<td>3.70e−2</td>
</tr>
<tr>
<td>80</td>
<td>1.01e−2</td>
<td>1.05e−2</td>
</tr>
<tr>
<td>160</td>
<td>2.75e−3</td>
<td>2.83e−3</td>
</tr>
<tr>
<td>320</td>
<td>7.11e−4</td>
<td>7.26e−4</td>
</tr>
</tbody>
</table>

\[ \delta(\mathbf{x}) = \prod_{l=1}^{3} d_h(x_l), \quad \mathbf{x} = (x_1, x_2, x_3)^T, \]

and

\[ d_h(r) = \begin{cases} \frac{1}{60} (1 + \cos(\frac{\pi r}{h})) & \text{for } |r| \leq 2h \\ 0 & \text{otherwise} \end{cases} \]

where \( h \) is the step size. Similarly, the solid velocity \( \mathbf{u}_i \) is also distributed onto the fluid grid by using the delta function.

3.5. Numerical procedure

We summarize our numerical approach to solve the whole system (3.33)–(3.36) as shown in Algorithm 2. This algorithm is based on the implicit calculation of the displacement of the solid part, and appears to be robust and accurate by our numerical experiments.

4. Numerical results

4.1. Numerical accuracy test for fluid solver

In this section, we consider numerically solving the incompressible Navier–Stokes equation only, i.e., the solid equation will be ignored. By using the projection method, \( u_i^{n+1} = u_i^n \) since the DLM \( \lambda \) is 0 on the domain \( \Omega \). A two-dimensional benchmark example is used to check the accuracy of our numerical scheme: Denote \( \mathbf{u}_f = (u_1, u_2)^T \), the NS equation is defined on the box \([0, 2\pi] \times [0, 2\pi]\). We first take the initial condition as

\[ u_1(x, y, 0) = -\cos(x) \sin(y), \quad u_2(x, y, 0) = \sin(x) \cos(y), \]

which is used in [8]. The exact solution for this case is known:

\[ u_1(x, y, t) = -\cos(x) \sin(y)e^{2t/\text{Re}}, \quad u_2(x, y, 0) = \sin(x) \cos(y)e^{-2t/\text{Re}}. \]

We take \( h_x = h_y = \frac{\pi}{M} \) with \( M = 20, 40, 80, 160 \) and 320. The solution is computed up to \( t = 2 \) and the \( L_2 \) error of the velocity and numerical order of accuracy are listed in Table 1. We can clearly observe the second-order accuracy in this table.

4.2. Numerical accuracy test for computing curvatures and surface Laplacian

In this section, we test the accuracy of computing surface Laplacian by employing level set method and ENO reconstruction shown in Algorithm 1. The formulas of mean and Gaussian curvatures of an ellipsoid is given in Appendix A. The analytical surface Laplacian is computed by using symbolic tool box in Matlab. We choose \( a = 1, b = 2, c = 3 \). The \( L_2 \) error of the curvatures and surface Laplacian, as well as numerical order of accuracy are listed in Table 2. We can clearly observe the first-order accuracy in this table.
We note that even for $N = 100$, the model still gives a very good approximation to the experimental result. Fig. 4 shows the deformation of the RBC surface with $N = 1000$.
4.4. RBC tank treading movement

In this section, we investigate the motion of a three-dimensional RBC in shear flows. The simulation setup is shown in Fig. 5, where the cell is placed in the center of the computational domain. The shear plane is parallel to the x–z plane. Two parallel flows moving in opposite directions are used to generate the linear shear flow. The boundaries in y direction are set to be wall boundaries. The boundaries in x direction are set to be periodic. The resolution in the computational domain is 80 × 80 × 80. The RBC membrane consists of 1000 nodes. The Reynolds number is chosen as 0.01.

Using the cone-plate rheoscope, it is possible to record the motion and deformation of individual RBCs suspended in simple shear flow. The stationary state of motion called “tank-treading” has been documented in [6,22,53,55,70,73]. As seen in the rheoscope, the tank-treading RBC assumes a flattened, quasi-ellipsoidal form while its membrane rotates steadily. In addition to the cell’s elliptical periphery projected on the plane of shear, the translational speed of the membrane, made visible by the rheoscope, and thus its frequency, \( f \), are directly measurable.

The tank-treading dynamic was theoretically predicted by the KS theory [53]: the cell is modeled as a non-deformable ellipsoid of a Newtonian fluid with a linear shear external flow. The cell dynamic is described in terms of an ODE for the instantaneous inclination angle \( \theta \) of the ellipsoid major axis with respect to the flow direction

\[
\frac{d\theta}{dt} = -(\frac{1}{2} \gamma + \frac{2S_1 S_2}{S_1^2 + S_2^2} \nu) + \frac{1}{2} \gamma \frac{S_1^2 - S_2^2}{S_1^2 + S_2^2} \cos(\theta),
\]

where \( \gamma \) is the shear rate, \( S_1 \) and \( S_2 \) are the semi-major and minor axes of the ellipsoid shown in Fig. 5, and \( \nu \) is the instantaneous membrane tank-treading frequency, which was obtained in [53,73] as:

\[
\nu = -\frac{\gamma f_3}{f_2 - \lambda f_1} \cos(\theta),
\]

where \( f_i, i = 1, 2, 3 \) is the dimensionless function of the three major axes of the ellipsoid (see [53,73] for more detail). The tank-treading frequency, \( \nu \), depends on the cell shape, shear rate, and viscosity ratio.

There are several experimental studies in the past which have been devoted to measuring the tank-treading frequency [6, 22,70]. We compared our numerical simulation result with the experimental data from [6]. Fig. 6 shows this agreement of the tank treading frequency on shear rate for human RBC. Figs. 7 and 8 show the agreement of the tank treading inclination angle on the analytical solutions shown in [1,67], and the comparison of Swinging average angle and amplitude for different shear rates.
Fig. 7. Comparison of TT inclination angle on shear rate with analytical solutions from [1,67].

Fig. 8. Swinging average angle and amplitude for different shear rates.

Fig. 9. The snapshots of the tank treading motion at various times within a period. The shear rates is $30 \text{ s}^{-1}$, the number of grid points is 1136.

Fig. 9 illustrates how the position of the membrane material marker moves around the cell, indicating that the trajectory is an ellipsoidal-like shape. Moreover, the trajectory of the marker is the same for every tank-treading period, while the RBC shape varies.

4.5. Platelet flipping motion in shear flow

The platelet membrane, which is similar to red blood cell, is also composed of a lipid bilayer and an attached cytoskeleton. We investigate the three-dimensional motion of an oblate spheroid-shaped particle or platelet in shear flow, and compare our simulation results with the theoretical solutions of motion of rigid spheroid-shaped cell obtained using Jeffery’s orbit theory [26]. In this simulation, we choose $K_{\text{bend}} = 1000K_{\theta}T$ in order to increase the stiffness of cell. (See Fig 10.) The values of the other parameters are same as Subsection 4.2. The angle of platelet orientation, namely, the instantaneous inclination angle $\theta$ of the ellipsoid major axis with respect to the flow direction, was obtained as [26]:

$$\theta(t) = \arccos \left( \frac{1}{2} \left( 1 - \frac{\sin(2\pi \gamma t)}{2\pi \gamma t} \right) \right)$$
where

\[ \tan(\theta) = \frac{S_2}{S_1} \tan \left( \frac{\gamma}{S_1 S_2} \gamma t \right), \]

where \( \dot{\gamma} \) is the shear rate, \( S_1 \) and \( S_2 \) are the semi-major and minor axes of the ellipsoid shown in Fig. 5, and \( t \) is the time.

Fig. 10 shows the simulated platelet flipping rate is in close agreement with the Jeffery orbit. Fig. 11 shows the snapshots of the platelet during the flipping process in our simulations. The platelet or circular disk was oriented with its major axis parallel to the surface so that its axis of symmetry lay at right angles to the surface. These results are verified using the analytical solutions of [26] for the rotational trajectory of an oblate spheroid owing in the linear shear flow.

4.6. Bacterial swimming

Bacteria swimming is an important research field, particularly in molecular biology, genetics, biochemistry. Individual cells of these bacteria are rod-shaped, and their surface is peritrichously flagellated. We focus on flagellated bacteria that are able to swim in a viscous fluid by means of self-propulsion, for example Escherichia coli (E. coli) and Bacillus subtilis (B. subtilis). These are very common bacteria that can be found in digestive tract of animals, or in soil and water. They are so small that, on their scale, the fluid is essentially viscous and inertial effects can be ignored when describing their locomotion. In fact, these microorganisms live in the low Reynolds regime (\( \text{Re} \sim 10^{-2} \)) (see [9,18] and references therein).

We denote the bacteria region as \( B \), and the flagellar bundle as \( P \) in Fig. 12. Following [18], we model bacteria region \( B \) as ellipse, with an associated point force representing the action of flagella. \( P \) is a very elongated ellipse. Here we only take into account the force they exert on the particles and on the fluid. The force exerted by the flagellar bundle on the flagellar bundle as a volume force density \( \mathbf{f}_p \) supported in \( P \). The propulsion force is directed outward from the center of \( B \), parallel to the major semiaxes of \( B \) and \( P \), and has the orientation angle \( \theta \). The force exerted on the fluid by the flagella \( \mathbf{f}_p \) can be defined as [18]:

\[ \mathbf{f}_p = -r \tau, \quad \mathbf{f}_b = r \tau, \]

where \( r \) is a constant magnitude [18,47], \( \tau \) is given by the orientation angle \( \theta \), \( \mathbf{f}_b \) is the force to the bacteria.
We investigate the bacterial swimming by using our simulation model. Here we neglect the differences of the membrane structure between bacterial cells and RBCs, and simply assume that the mechanics of the bacterial cell membrane can also be produced by our cell model. In Fig. 13 we show the centroid trajectories of our bacteria cell. The centroid trajectory forms a helix with total run time of 0.5 seconds, and agrees with the result of [47].

5. Conclusion

In this paper, we present a hybrid model for modeling cell membrane of RBCs or other types of cells with membrane structures similar to that of RBCs. The model incorporates bending mechanics of the lipid bilayer, and viscoelasticity of the cytoskeleton network. Several numerical techniques, including level set method, ENO reconstruction and immerse boundary interpolation were employed to numerically compute stiff terms in the governing equations. DLM/FD formulation was extended to model fluid–flexible cell interactions. A second-order accurate fractional step scheme was developed to solve the fluid problem. The solid problem was solved by the Newton–Krylov iterative method. The proposed model and numerical methods were applied to several typical fluid–cell interaction problems: RBC movement, platelet flipping, and bacteria swimming. All the simulation results are in good agreement with the previous numerical and experimental work, which validates our cell membrane model and numerical methods.

Acknowledgements

We would like to express our thanks to Professor Shuwang Li for helpful discussions related to computing elastic bending force.

WH has been supported by the Mathematical Biosciences Institute and the National Science Foundation under Grant DMS 0931642. ZX’s research was partially supported by NSF Grants DMS-1115887, DMS-0800612 and NIH Grants 1 R01 GM100470-01, 1 R01 GM095959-01A1 and IU01HL116330-01A1. The work of CL has been partially supported by NSF Grants DMS-109107, DMS-1216938, and DMS-1159937. GL would like to acknowledge support from the Applied Mathematics Program within the Department of Energy’s (DOE) Office of Advanced Scientific Computing Research (ASCR) as part of the Collaboratory on Mathematics for Mesoscopic Modeling of Materials (CM4). PNNL is operated by Battelle for the DOE under Contract DE-AC05-76RL01830.

Appendix A

A.1. Formula for the surface Laplacian and the mean curvature

The surface Laplacian is defined as

$$\Delta_{\Sigma} = \Delta - 2H n \cdot \nabla - n \otimes n : \nabla^2,$$

and the mean curvature $H$ may be defined as follows:

$$H = -\frac{1}{2} \nabla \cdot n.$$
Denoting by \( d \) the level set function, the Mean Curvature \( H \) can be expressed explicitly in terms of first and second order partial derivatives of \( d \) as
\[
H = \frac{d_{xx} + d_{yy} + d_{zz} - N}{\sqrt{d_x^2 + d_y^2 + d_z^2}}.
\]
where
\[
N = \frac{d_x^2 d_{xx} + d_y^2 d_{yy} + d_z^2 d_{zz} + 2d_x d_y d_{xy} + 2d_x d_z d_{xz} + 2d_y d_z d_{yz}}{d_x^2 + d_y^2 + d_z^2}.
\]
And Gaussian Curvature can be written as
\[
K = \frac{[d_z(d_{xx}d_z - 2d_x d_{xz}) + d_x^2 d_{zz}] [d_z(d_{yy}d_z - 2d_y d_{yz}) + d_y^2 d_{zz}] - [d_z(-d_x d_{yz} + d_y d_x - d_y d_z) + d_x d_y d_{zz}]^2}{d_x^2(d_x^2 + d_y^2 + d_z^2)^2}.
\]
And the surface Laplacian can be written as
\[
\Delta_S = \sum_{i=1}^{3} \frac{\partial^2}{\partial x_i^2} - 2H \sum_{i=1}^{3} n_i \frac{\partial}{\partial x_i} + \sum_{i=1}^{3} \sum_{j=1}^{3} n_i n_j \frac{\partial^2}{\partial x_i \partial x_j},
\]
where the normal vector \( n = (n_1, n_2, n_3)^T \), and \( \partial_i, \partial_{ij} \) are the differential operators with respect to \( x = (x, y, z)^T \).

A.1.1. Multi-grid method

One effective numerical way of solving the Poisson equation (3.38) is to solve it using multi-grid method [4]. As the grid size increases, the multi-grid solution will approach the exact solution very quickly. For example, the equation is discretized on a grid with \( N + 2 \) lattice points, i.e., \( N \) interior points and 2 boundary points, in the \( x \), \( y \) and \( z \) directions. The algorithm uses a succession of lattices or grids. The number of different grids is called the number of multi-grid levels \( l \). The number of interior lattice points in the \( x \), \( y \) and \( z \) directions is then taken to be \( 2^l \), so that \( N = 2^l + 2 \). \( N \) is chosen in this manner so that the downward multi-grid iteration can construct a sequence of coarser lattices with
\[
2^{l-1} \rightarrow 2^{l-2} \rightarrow \ldots \rightarrow 2^0 = 1,
\]
iinterior points in each direction. The multi-grid method starts with the coarsest grid on which the equation can be solved exactly. It then proceeds to finer grids, performing one or more V-cycles [11] at each level along the way, and gets the solutions on the finer grids. In order to implement our finite difference scheme to solve the Poisson equation (3.38), we needed data structures that implemented multi-grid. We chose to use the Portable, Extensible Toolkit for Scientific Computing (PETSc) [3] created by Argonne National Laboratory. PETSc is a set of data structures that allow for parallel processing of solutions to partial differential equations.

A.1.2. Newton–Krylov method

Newton–Krylov method is used to solve the nonlinear solid equation (3.39), and employs a Krylov method inside of Newton’s method in a Jacobian free manner through directional differentiation. It also derive a pre-conditioner for the Krylov accelerator that relies primarily on local information, for data-parallel concurrency. It seems particularly well suited for solving nonlinear equation for large scale systems in high-accuracy, distributed memory environments. We used a Newton–Krylov algorithm with an emphasis on domain decomposition iterative aspects. We chose to use Newton–Krylov algorithm in KINSol [23], which is a Newton–Krylov solver package written for solution of systems arising in the solution of partial differential equations. This package makes use of Newton’s method to solve the discrete nonlinear systems and applies pre-conditioned Krylov linear solvers for solution of the Jacobian system at each nonlinear iteration, and approximates Jacobian matrix by a finite difference of the nonlinear system-defining function, avoiding a requirement for the formation of the actual Jacobian matrix. Significantly less memory is required for the solver as a result.

A.1.3. Curvatures of ellipsoid

The general ellipsoid is a quadratic surface which is given in Cartesian coordinates by
\[
\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1,
\]
where \( a, b \) and \( c \) are the lengths of semi-axes. The parametric equations of an ellipsoid can be written as
\[
x = a \cos(u) \sin(v), \quad y = b \sin(u) \sin(v), \quad z = c \cos(v),
\]
where \( u \in [0, 2\pi) \) and \( v \in [0, \pi] \).
Then the Gaussian curvature is
\[ K = \frac{a^2 b^2 c^2}{(a^2 b^2 \cos^2 \varphi + c^2 (b^2 \cos^2 \varphi + a^2 \sin^2 \varphi) \sin^2 \varphi)^2}, \]
and the mean curvature is
\[ H = \frac{abc(3(a^2 + b^2) + 2c^2 + (a^2 + b^2 + c^2) \cos(2\varphi) - 2(a^2 - b^2) \cos(2u) \sin^2 \varphi)}{8(a^2 b^2 \cos^2 \varphi + c^2 (b^2 \cos^2 \varphi + a^2 \sin^2 \varphi) \sin^2 \varphi)^{3/2}}. \]

References
