Journal of Computational Physics 228 (2009) 8427–8445

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

An implicit immersed boundary method for three-dimensional fluid–membrane interactions

D.V. Le [a,b,e,*], J. White [a,b], J. Peraire [a,c], K.M. Lim [a,d], B.C. Khoo [a,d]

[a] Singapore-MIT Alliance, 4 Engineering Drive 3, National University of Singapore, Singapore 117576, Singapore
[b] Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 77 Massachusetts Avenue, MA 02139, USA
[c] Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, 77 Massachusetts Avenue, MA 02139, USA
[d] Department of Mechanical Engineering, National University of Singapore, Kent Ridge Crescent, Singapore 119260, Singapore
[e] Institute of High Performance Computing, 1 Fusionopolis Way, #16-16 Connexis, Singapore 138632, Singapore

ARTICLE INFO

Article history:
Received 14 October 2008
Received in revised form 30 April 2009
Accepted 20 August 2009
Available online 2 September 2009

MSC:
65M06
76D05
74F10
92C10

Keywords:
Immersed boundary method
Newton–Krylov method
Navier–Stokes equations
Membrane capsules
Red blood cells

ABSTRACT

We present an implicit immersed boundary method for the incompressible Navier–Stokes equations capable of handling three-dimensional membrane–fluid flow interactions. The goal of our approach is to greatly improve the time step by using the Jacobian-free Newton–Krylov method (JFNK) to advance the location of the elastic membrane implicitly. The most attractive feature of this Jacobian-free approach is Newton-like nonlinear convergence without the cost of forming and storing the true Jacobian. The Generalized Minimal Residual method (GMRES), which is a widely used Krylov-subspace iterative method, is used to update the search direction required for each Newton iteration. Each GMRES iteration only requires the action of the Jacobian in the form of matrix–vector products and therefore avoids the need of forming and storing the Jacobian matrix explicitly. Once the location of the boundary is obtained, the elastic forces acting at the discrete nodes of the membrane are computed using a finite element model. We then use the immersed boundary method to calculate the hydrodynamic effects and fluid–structure interaction effects such as membrane deformation. The present scheme has been validated by several examples including an oscillatory membrane initially placed in a still fluid, capsule membranes in shear flows and large deformation of red blood cells subjected to stretching force.

1. Introduction

This paper considers an implicit immersed boundary method for simulating viscous incompressible flows with immersed elastic membranes. The immersed boundary (IB) method was originally developed by Peskin [36] to study the fluid dynamics of blood flow in the human heart. Peskin's immersed boundary method has proven to be a very useful method for modeling fluid–structure interaction involving large geometry variations. The original method has been developed further and applied to many biological problems including platelet aggregation [14,15,52], the deformation of red blood cells in a shear flow [11], the swimming of bacterial organisms and others [10,13]. More details on the immersed boundary method can be found in [37] and the references therein.

Typically, in the framework of the IB method, the elastic boundary is treated as a collection of elastic fibers. Alternatively, the elastic boundary is also modeled as an elastic membrane [11] or a thin shell [17]. These models are used to compute the

* Corresponding author. Address: Institute of High Performance Computing, 1 Fusionopolis Way, #16-16 Connexis, Singapore 138632, Singapore. Tel.: +65 64191219; fax: +65 64674350.
E-mail addresses: ledv@ihpc.a-star.edu.sg (D.V. Le), white@mit.edu (J. White), peraire@mit.edu (J. Peraire), mpelimkm@nus.edu.sg (K.M. Lim), mpekbc@nus.edu.sg (B.C. Khoo).

0021-9991/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2009.08.018

forces acting at the discrete nodes representing the immersed boundary. For most biological tissues, their membranes are stiff and therefore a small perturbation of the boundary can lead to large elastic forces. This causes a severe restriction in time step required to maintain stability of the immersed boundary method. Much effort has been made to analyze the stiffness of the IB method and remove this restriction [18,22,33,47,48]. Several semi-implicit and implicit methods have been developed to alleviate this problem [12,21,31,35,51]. Comparisons of the explicit method and the implicit methods in the context of moving immersed boundaries have been presented in [35,51]. In the context of immersed interface methods (IIM) a quasi-Newton method has been proposed [27–29] to improve time stability. However, for very stiff problems, small time steps are still required. In order to alleviate this problem, an unconditionally stable discretization of the immersed boundary equations has been proposed [34]. In this scheme, an approximate Newton solver is employed to advance the location of the boundary. The Newton method, however, requires a Jacobian matrix which is extremely expensive to compute explicitly for every time step. To avoid forming the Jacobian matrix explicitly, an iterative matrix-free method has been introduced in the context of the immersed continuum method [53]. Another strategy for solving the implicit IB equations involves deriving Schur complement equations by eliminating one or more of the unknown variables [33,35]. Several methods have been employed to solve the Schur complement equations such as fixed point methods, projection methods and Krylov-subspace methods.

Recently, an efficient semi-implicit IB method with arclength–tangent angle formulation has been proposed for two-dimensional Navier–Stokes equations [21]. In this formulation, an unconditionally stable semi-implicit discretization is derived and the small scale decomposition technique [20] is applied to the discretization. This semi-implicit scheme has much better stability property than the explicit scheme and therefore offers a substantial computational cost saving. We note that the stability of this scheme is weaker than the unconditionally stable scheme proposed in [34] because this scheme treats only the leading order term implicitly [21].

In the present paper, an implicit immersed boundary method is presented with vastly improved time step by using the Jacobian-free Newton–Krylov method (JFNK) [24] to advance the location of the elastic membrane implicitly. This matrix-free approach has many advantages. The most attractive is Newton-like nonlinear convergence without the cost of forming and storing the true Jacobian. In this Jacobian-free method, the search direction required for each Newton iteration is updated using the Generalized Minimal Residual method (GMRES) [44], which is a widely used Krylov-subspace iterative method. Each GMRES iteration only requires the action of the Jacobian in the form of matrix–vector products and therefore avoids the need of forming and storing the Jacobian matrix explicitly. The JFNK method has become established in computational fluid dynamics (CFD) to deal with the nonlinear convection term [3,6,16,25]. In this paper, we employ the JFNK method to advance the membrane location implicitly while still approximating the convection term explicitly. Once the location of the boundary is obtained, the elastic forces acting at the discrete nodes of the membrane are computed. In the present work, an elastic membrane model with bending stiffness proposed in [38,42] is employed. In our numerical studies, the immersed boundary method provides the means of calculating the hydrodynamics and fluid–structure interaction effects such as membrane deformation, and the finite element method with membrane model is used to calculate the elastic forces on the membrane.

The remainder of this paper is organized as follow. In Section 2, we describe the governing equations, the immersed boundary algorithm, the discrete model of capsule membranes and the method for advancing the membrane evolution in time. We then present a detailed implementation of the present method in Section 3. In Section 4, some numerical examples are presented and finally, some conclusions are given in Section 5.

2. Numerical methods

2.1. Governing equations

In a three-dimensional bounded domain Ω that contains an enclosed membrane $\Gamma(t)$, we consider the incompressible Navier–Stokes equations formulated in primitive variables, written as

$$\rho(\boldsymbol{u}_t + (\boldsymbol{u}\cdot\nabla)\boldsymbol{u}) = -\nabla p + \mu\Delta\boldsymbol{u} + \boldsymbol{F}, \tag{1}$$

$$\nabla\cdot\boldsymbol{u} = 0 \tag{2}$$

with boundary conditions

$$\boldsymbol{u}|_{\partial\Omega} = \boldsymbol{u}_b, \tag{3}$$

where $\boldsymbol{u}$ is the fluid velocity, p is the pressure, ρ and μ are constant density and viscosity of the fluid, respectively. The effect of the membrane $\Gamma(t)$ immersed in the fluid results in a singular force $\boldsymbol{F}$ which has the form

$$\boldsymbol{F}(\boldsymbol{x},t) = \int_{\Gamma(t)} \boldsymbol{f}(q,r,s,t)\delta(\boldsymbol{x}-\boldsymbol{X}(q,r,s,t))\,\mathrm{d}q\,\mathrm{d}r\mathrm{d}s, \tag{4}$$

where (q,r,s) are curvilinear coordinates attached to the membrane at a material point, $\boldsymbol{X}(q,r,s,t)$ is the position at time t in Cartesian coordinates of the material point whose label is (q,r,s), $\boldsymbol{x}=(x,y,z)$ is spatial position, and $\boldsymbol{f}(q,r,s,t)$ is the force

strength. Here, $\delta(\boldsymbol{x})$ is the three-dimensional Dirac function. The motion of the boundary can be determined by integrating the equation

$$\frac{\mathrm{d}\boldsymbol{X}(q,r,s,t)}{\mathrm{d}t} = \int_{\Omega} \boldsymbol{u}(\boldsymbol{x},t)\delta(\boldsymbol{x}-\boldsymbol{X}(q,r,s,t))\,\mathrm{d}\boldsymbol{x}. \tag{5}$$

2.2. Description of the IB and projection methods

The immersed boundary method uses a set of N control points $\boldsymbol{X}_l,\ l = 1,\ldots,N$ to represent the immersed boundary. The force density is computed at these control points and is distributed to the Cartesian grid points using a discrete representation of the delta function,

$$\boldsymbol{F}(\boldsymbol{x}(i,j,k),t) = \sum_{l=1}^{N} \boldsymbol{f}_l(q,r,s,t) D_h(\boldsymbol{x}(i,j,k) - \boldsymbol{X}_l(t))\Delta q \Delta r \Delta s, \tag{6}$$

where $\boldsymbol{f}_l(q,r,s,t)$ is the force density at the control point $\boldsymbol{X}_l$ whose label is (q,r,s), $\boldsymbol{x}(i,j,k)$ and $\boldsymbol{F}(\boldsymbol{x}(i,j,k))$ are the coordinate of grid point (i,j,k) and the force at that point, respectively. $D_h(\boldsymbol{x})$ is a three-dimensional discrete delta function,

$$D_h(\boldsymbol{x}) = \frac{1}{h^3}\varphi\left(\frac{x}{h}\right)\varphi\left(\frac{y}{h}\right)\varphi\left(\frac{z}{h}\right), \tag{7}$$

where h is the grid size, x, y and z are the Cartesian components of $\boldsymbol{x}$ and φ is a continuous function which is taken from [43] and given by

$$\varphi(d) = \begin{cases} \frac{1}{6}\left(5 - 3|d| - \sqrt{1-3(1-|d|)^2}\right), & 0.5 \leqslant |d| \leqslant 1.5, \\ \frac{1}{3}\left(1 + \sqrt{-3d^2+1}\right), & |d| \leqslant 0.5, \\ 0, & \text{otherwise}. \end{cases} \tag{8}$$

Once the force density is computed at the control points and distributed to the grid, the Navier–Stokes equations with the forcing terms are then solved for the pressure and velocity field at the Cartesian grid points using the projection method [4]. Our numerical algorithm is based on the pressure-increment projection algorithm for the discretization of the Navier–Stokes equations. The spatial discretization is carried out on a standard marker-and-cell (MAC) staggered grid similar to that described in Kim and Moin [23]. Given the velocity $\boldsymbol{u}^n$, the pressure $p^{n-1/2}$ and the forcing term $\boldsymbol{F}^{n+1/2}$, we compute the intermediate velocity $\boldsymbol{u}^*$ as follows:

$$\begin{aligned} &\rho\frac{\boldsymbol{u}^* - \boldsymbol{u}^n}{\Delta t} = -\rho(\boldsymbol{u}\cdot\nabla\boldsymbol{u})^{n+\frac{1}{2}} - G^{MAC}p^{n-\frac{1}{2}} + \frac{\mu}{2}\left(\nabla_h^2\boldsymbol{u}^* + \nabla_h^2\boldsymbol{u}^n\right) + \boldsymbol{F}^{n+1/2}, \\ &\boldsymbol{u}^*|_{\partial\Omega} = \boldsymbol{u}_b^{n+1}, \end{aligned} \tag{9}$$

where the advective term is extrapolated using the formula,

$$(\boldsymbol{u}\cdot\nabla\boldsymbol{u})^{n+\frac{1}{2}} = \frac{3}{2}(\boldsymbol{u}\cdot\nabla_h\boldsymbol{u})^n - \frac{1}{2}(\boldsymbol{u}\cdot\nabla_h\boldsymbol{u})^{n-1}. \tag{10}$$

This intermediate velocity field, in general, does not satisfy the divergence-free condition (2). Therefore, we compute a pressure-increment ϕ^{n+1} and update the pressure and velocity field as

$$\nabla_h^2\phi^{n+1} = \rho\frac{D^{MAC}\boldsymbol{u}^*}{\Delta t}, \quad \boldsymbol{n}\cdot\nabla\phi^{n+1}|_{\partial\Omega} = 0, \tag{11}$$

$$\boldsymbol{u}^{n+1} = \boldsymbol{u}^* - \frac{1}{\rho}\Delta t G^{MAC}\phi^{n+1}, \tag{12}$$

$$p^{n+1/2} = p^{n-1/2} + \phi^{n+1} - \frac{\mu}{2\rho}(D^{MAC}\boldsymbol{u}^*). \tag{13}$$

We note that the above projection method is analogous to the pressure-increment projection method presented in [4]. In the above expressions, ∇_h and ∇_h^2 are the standard central difference operators, G^{MAC} and D^{MAC} are the MAC gradient and divergence operators, respectively [26].

The velocity field is then interpolated to find the velocity at the control points as,

$$\boldsymbol{u}(\boldsymbol{X}_l,t) = \sum_{i,j,k} \boldsymbol{u}(\boldsymbol{x}(i,j,k),t) D_h(\boldsymbol{x}(i,j,k) - \boldsymbol{X}_l(t))h^3, \tag{14}$$

and this velocity is used to advance the position of the membrane.

2.3. Discrete model of the capsule membrane

2.3.1. Membrane tensions

Following the standard procedure [2,42], in order to keep track of the position of a material point on the membrane, let $\overline{\boldsymbol{X}}$ and $\boldsymbol{X}(\overline{\boldsymbol{X}},\boldsymbol{t})$ be its positions in the unstressed state and after deformation at time t, respectively. Let $\boldsymbol{C}$ be the deformation gradient tensor defined as

$$\boldsymbol{C} = \frac{\partial \boldsymbol{X}}{\partial \overline{\boldsymbol{X}}}. \tag{15}$$

Following [2], the surface deformation gradient tensor is defined as

$$\boldsymbol{A} = (\boldsymbol{I} - \boldsymbol{nn}) \cdot \boldsymbol{C} \cdot (\boldsymbol{I} - \bar{\boldsymbol{n}}\bar{\boldsymbol{n}}), \tag{16}$$

where $\bar{\boldsymbol{n}}$ and $\boldsymbol{n}$ are the unit normal vector to the undeformed and deformed membranes, respectively. The left Cauchy–Green strain tensor is then defined as

$$\boldsymbol{B} = \boldsymbol{\Lambda}^2 = \boldsymbol{A} \cdot \boldsymbol{A}^T. \tag{17}$$

The membrane tension tensor $\boldsymbol{\tau}$ is related to the surface strain energy function $W(\Lambda_1, \Lambda_2)$ and $\boldsymbol{B}$ by

$$\boldsymbol{\tau} = e^{-\Lambda_1}\left(\frac{\partial W}{\partial \Lambda_1}\boldsymbol{P} + \frac{\partial W}{\partial \Lambda_2}\boldsymbol{B}\right), \tag{18}$$

where $\boldsymbol{P} = \boldsymbol{I} - \boldsymbol{nn}$ and $\Lambda_1,\ \Lambda_2$ are the strain invariants

$$\Lambda_1 = \log \lambda_1\lambda_2 = \frac{1}{2}\log\left(\frac{1}{2}(\mathrm{tr}(\boldsymbol{B})^2 - \mathrm{tr}(\boldsymbol{B}^2))\right), \tag{19}$$

$$\Lambda_2 = \frac{1}{2}(\lambda_1^2 + \lambda_2^2) - 1 = \frac{1}{2}\mathrm{tr}\boldsymbol{B} - 1. \tag{20}$$

Here, the eigenvalues $\lambda_1,\ \lambda_2$ of $\boldsymbol{\Lambda}$ are the principle planar stretches. The strain energy function for a neo-Hookean membrane is given by

$$W = \frac{E\bar{h}}{6}(2\Lambda_2 + e^{-2\Lambda_1} - 1), \tag{21}$$

where E is the Young's modulus and $\bar{h}$ is the membrane thickness. For a thin membrane,

$$W = \frac{E_s}{6}(2\Lambda_2 + e^{-2\Lambda_1} - 1), \tag{22}$$

where $E_s = E\bar{h}$ is the surface elastic modulus, to be distinguished from the volume modulus of elasticity of a three-dimensional material [40]. Alternative constitutive equations for hyperelastic materials such as biological membranes can be used. For example, the Yeoh form of strain energy function [54] given by

$$W = C_{10}(2\Lambda_2 + e^{-2\Lambda_1} - 1) + C_{30}(2\Lambda_2 + e^{-2\Lambda_1} - 1)^3, \tag{23}$$

where C_{10} and C_{30} are constants, can be used to model red blood cell (RBC) deformation. Another strain energy function for the red blood cell membrane which conforms to deformation measurements for human RBCs has been developed in [46]

$$W = \frac{C_{11}}{4}(2\Lambda_2(\Lambda_2 + 1) + 1 - e^{2\Lambda_1}) + \frac{C_{21}}{8}(e^{2\Lambda_1} - 1)^2, \tag{24}$$

where C_{11} and C_{21} are model constants and typically $C_{11} \ll C_{21}$.

2.3.2. Membrane bending moments

The bending moments developing along the edges of a membrane patch in the deformed state depend on the instantaneous edge curvature, as well as on the edge curvature in the rest state. For sufficiently small bending deformations, but not necessarily small in-plane deformations, the bending moments may be approximated with the linear constitutive equation

$$\boldsymbol{m} = \kappa_B(\boldsymbol{\kappa} - \kappa_m^R\boldsymbol{P}), \tag{25}$$

where κ_B is the bending modulus, $\boldsymbol{\kappa}$ is the Cartesian curvature tensor and κ_m^R are the position dependent mean curvatures of the resting configurations. More details on the derivation of Eq. (25) can be found in [40,41].

Finally, the local force density $\boldsymbol{f}$ exerted by the membrane is given by

$$\boldsymbol{f} = \nabla^s \cdot (\boldsymbol{\tau} + \boldsymbol{qn}), \tag{26}$$

where ∇^s is the surface gradient operator $\nabla^s = (\boldsymbol{I} - \boldsymbol{nn}) \cdot \nabla$ and $\boldsymbol{q}$ is the transverse shear tension. The expression for the transverse shear tension vector in terms of the surface divergence of the tensor of the bending moments is

$$\boldsymbol{q} = [(\boldsymbol{P} \cdot \nabla) \cdot \boldsymbol{m}] \cdot \boldsymbol{P} = \mathrm{tr}[(\boldsymbol{P} \cdot \nabla)\boldsymbol{m}] \cdot \boldsymbol{P}. \tag{27}$$

2.4. Advancing the membrane

2.4.1. Explicit method

The simplest version of the immersed boundary method is the explicit scheme in which the forward Euler method is used to advance the location of the membrane. This explicit scheme includes the following steps:

- Advance the control points as
$$\boldsymbol{X}^{n+1} = \boldsymbol{X}^n + \Delta t \boldsymbol{u}^n(\boldsymbol{X}^n). \tag{28}$$
- Use $\boldsymbol{X}^{n+1}$ to compute the boundary force as described in Section 2.3. Distribute the boundary force to the nearby Cartesian grid points using discrete delta function.
- Solve the Navier–Stokes equations using the projection method to calculate the velocity field and pressure field.
- Interpolate the velocity field to determine the velocity at the control points $\boldsymbol{u}^{n+1}(\boldsymbol{X}^{n+1})$.

This method is simple, but for problems where the membrane is stiff, a very small time step is required to maintain stability. Details on the stability issues of the explicit immersed boundary method are discussed in [51].

2.4.2. Quasi-Newton method

In order to improve the time step, one could use an implicit method. The position of the control points is updated using the trapezoidal rule

$$\boldsymbol{X}^{n+1} = \boldsymbol{X}^n + \frac{1}{2}\Delta t(\boldsymbol{u}^n(\boldsymbol{X}^n) + \boldsymbol{u}^{n+1}(\boldsymbol{X}^{n+1})). \tag{29}$$

Eq. (29) is implicit and couples the motion of the membrane with the solution at all grid points. Therefore at each time step, we need to solve a nonlinear system of equations for the position of the control points of the form

$$\boldsymbol{g}(\boldsymbol{X}^{n+1}) = \boldsymbol{0}, \tag{30}$$

where

$$\boldsymbol{g}(\boldsymbol{X}) = \boldsymbol{X} - \boldsymbol{X}^n - \frac{1}{2}\Delta t(\boldsymbol{u}^n(\boldsymbol{X}^n) + \boldsymbol{u}^{n+1}(\boldsymbol{X})). \tag{31}$$

Normally this nonlinear system of equations is solved by a Newton's method in which the Jacobian matrix is required,

$$\boldsymbol{J}(\boldsymbol{X}) = \boldsymbol{g}'(\boldsymbol{X}) = \boldsymbol{I} - \frac{1}{2}\Delta t \boldsymbol{u}'(\boldsymbol{X}). \tag{32}$$

Starting from $\boldsymbol{X}^{(0)}$ a Newton iteration involves a sequence of linear systems given by

$$\boldsymbol{J}(\boldsymbol{X}^{(k)})\delta\boldsymbol{X}^{(k)} = -\boldsymbol{g}(\boldsymbol{X}^{(k)}), \tag{33}$$
$$\boldsymbol{X}^{(k+1)} = \boldsymbol{X}^{(k)} + \delta\boldsymbol{X}^{(k)}, \quad k = 0, 1, \ldots \tag{34}$$

However, the evaluation of the Jacobian matrix and its inverse can be time consuming for many applications. A quasi-Newton method has been devised wherein the inverse Jacobian matrix $\boldsymbol{J}^{-1}$ is replaced by a suitable matrix $\boldsymbol{H}$, which is easy to compute. The matrix $\boldsymbol{H}$ at iteration $(k+1)$th, $\boldsymbol{H}_{k+1}$, is updated from the matrix $\boldsymbol{H}_k$ at the previous iteration as follows

$$\boldsymbol{H}_{k+1} = \gamma_k \boldsymbol{H}_k + \left(1 + \gamma_k \theta_k \frac{q_k^T \boldsymbol{H}_k q_k}{p_k^T q_k}\right)\frac{p_k p_k^T}{p_k^T q_k} - \gamma_k \frac{(1-\theta_k)}{q_k^T \boldsymbol{H}_k q_k}\boldsymbol{H}_k q_k \cdot q_k^T \boldsymbol{H}_k - \frac{\gamma_k \theta_k}{p_k^T q_k}\left(p_k q_k^T \boldsymbol{H}_k + \boldsymbol{H}_k q_k p_k^T\right), \tag{35}$$

where

$$p_k := \boldsymbol{X}^{(k+1)} - \boldsymbol{X}^{(k)}, \quad q_k := \boldsymbol{g}^{(k+1)} - \boldsymbol{g}^{(k)},$$

and the parameters $\gamma_k \geqslant 0,\ \theta_k \geqslant 0$. With $\gamma_k = 1,\ \theta_k = 1$, we have the rank two method of Broyden, Fletcher, Goldfard and Shanno (BFGS method) [49]. In practice, the BFGS method requires only a few iterations to converge as the solution at the previous time step provides a very good initial guess for the iteration. This has been observed in [27–29] for two-dimensional problems in the context of the immersed interface method. Note that the BFGS method needs to form and store the inverse Jacobian matrix $\boldsymbol{H}$, and consequently it requires a large storage in three-dimensional problems especially for those with multiple membranes in the fluid domain. The quasi-Newton method was not used in the present work for three-dimensional problems because of the limitations in computer memory. In order to avoid such large storage, we employ a Jacobian-free Newton–Krylov method as described in the next section.

2.4.3. Jacobian-free Newton–Krylov method

An alternative approach to solve (30) is the Jacobian-free Newton–Krylov (JFNK) method [24]. In the JFNK approach, the GMRES method is used to solve the sequence of linear systems given by Eq. (33). The GMRES method requires the action of the Jacobian only in the form of a matrix–vector product, which may be approximated by [5,7]:

$$\boldsymbol{J}(\boldsymbol{X})\boldsymbol{y} \approx \frac{\boldsymbol{g}(\boldsymbol{X}+\epsilon\boldsymbol{y}) - \boldsymbol{g}(\boldsymbol{X})}{\epsilon}, \tag{36}$$

where ϵ is a small perturbation. The error of this approximation is proportional to ϵ. In [7], ϵ was set equal to something larger than the square root of machine error (ϵ_{mach}). Another effective formula for the evaluation of ϵ is

$$\epsilon = \frac{\sqrt{(1+\|\boldsymbol{X}\|)\epsilon_{mach}}}{\|\boldsymbol{y}\|}. \tag{37}$$

More details on the Jacobian-free Newton–Krylov can be found in [24] and the references therein.

3. Implementation

The present implicit immersed boundary method involves the following steps:

1. Trace the immersed boundary using a collection of control points that define an unstructured grid of triangular elements as shown in Fig. 1.
2. Compute the surface forces at the control points.
3. Solve the Navier–Stokes equations to compute the velocity and pressure fields.
4. Advance the position of the control points.

In the numerical implementation, the immersed boundary is discretized into a mesh of six-node curved triangles. A function $u(\xi,\eta)$ defined over a triangle is approximated as

$$u(\xi,\eta) = \sum_{i=1}^{6} u_i N_i(\xi,\eta), \tag{38}$$

where (ξ,η) are the local parametric coordinates, u_i is the value of u at node i and $N_i(\xi,\eta)$ are the basis functions for a quadratic six-node triangular finite element [39]. To evaluate the membrane tension tensor $\boldsymbol{\tau}$, one needs to calculate the left Cauchy–Green strain tensor, which is determined from the surface deformation gradient tensor, $\boldsymbol{A}$. For each element, the surface deformation gradient tensors at the nodes are obtained by solving the following system of equations,

$$\boldsymbol{A}\cdot\frac{\partial\overline{\boldsymbol{X}}}{\partial\xi} = \frac{\partial\boldsymbol{X}}{\partial\xi}, \quad \boldsymbol{A}\cdot\frac{\partial\overline{\boldsymbol{X}}}{\partial\eta} = \frac{\partial\boldsymbol{X}}{\partial\eta}, \quad \boldsymbol{A}\cdot\bar{\boldsymbol{n}} = \boldsymbol{0}, \tag{39}$$

at each node of the element [42]. The components of $\boldsymbol{A}$ are then averaged over the elements sharing the node to obtain a smooth distribution. Once the deformation tensor has been determined, the strain and stress tensors follow from (17) and (18).

The transverse shear tension is computed from the bending moment, which is determined from the Cartesian curvature tensor, $\boldsymbol{\kappa}$. To evaluate the curvature tensor $\boldsymbol{\kappa}$ at a point, one needs to solve

$$\frac{\partial\boldsymbol{X}}{\partial\xi}\cdot\boldsymbol{\kappa} = \frac{\partial\boldsymbol{n}}{\partial\xi}, \quad \frac{\partial\boldsymbol{X}}{\partial\eta}\cdot\boldsymbol{\kappa} = \frac{\partial\boldsymbol{n}}{\partial\eta}, \quad \boldsymbol{n}\cdot\boldsymbol{\kappa} = \boldsymbol{0}, \tag{40}$$

at each element node and then average over the elements sharing that node [40]. Having computed the curvature tensor, one can evaluate the bending moments $\boldsymbol{m}$ using the constitutive equation (25). To evaluate the transverse shear tension, we need to compute the surface gradient of the bending moments denoted by $\boldsymbol{K} \equiv (\boldsymbol{P}\cdot\nabla)\boldsymbol{m}$, using the relations

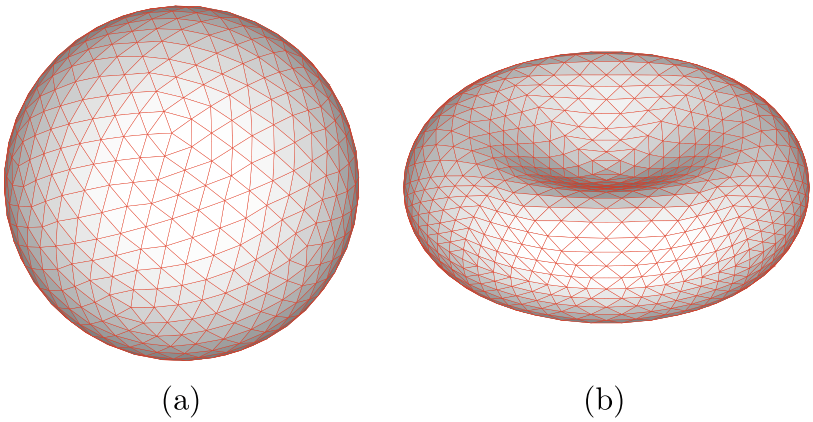


Fig. 1. (a) Discretization of a sphere. (b) Discretization of a biconcave shape.

$$\frac{\partial \boldsymbol{X}}{\partial \xi}\cdot\boldsymbol{K}=\frac{\partial \boldsymbol{m}}{\partial \xi},\quad \frac{\partial \boldsymbol{X}}{\partial \eta}\cdot\boldsymbol{K}=\frac{\partial \boldsymbol{m}}{\partial \eta},\quad \boldsymbol{n}\cdot\boldsymbol{K}=\boldsymbol{0}. \tag{41}$$

The evaluation of $\boldsymbol{K}$ at each element node requires solving nine systems of 3 linear equations [40]. The components of $\boldsymbol{K}$ are also averaged over the elements sharing the node, and the transverse shear tension can then be calculated using Eq. (27).

To evaluate the force density, we adopt the method developed in [38] where the average value of the force is evaluated on an element. The average value of the force on the element E_n with area S_n enclosed by the contour C_n is computed from the line integral

$$\overline{\Delta f}=\frac{1}{S_n}\oint_{C_n}[\boldsymbol{b}\cdot\boldsymbol{\tau}+(\boldsymbol{b}\cdot\boldsymbol{q})\boldsymbol{n}]\,\mathrm{d}\bar{l}, \tag{42}$$

where $\bar{l}$ is the arclength along the element contour, $\boldsymbol{b}\equiv\boldsymbol{t}\times\boldsymbol{n}$ is the cross-product of the unit tangential vector $\boldsymbol{t}$ along the contour C_n and the surface unit normal vector $\boldsymbol{n}$.

Once the forces are computed over all elements and distributed to the grid, the Navier–Stokes equations with the forcing terms are then solved for the pressure and velocity at the Cartesian grid points. This velocity field is interpolated to find the velocity at the control points, and the position of the control points is updated implicitly using the trapezoidal rule (29). In summary, given the location of the control points, $\boldsymbol{X}^n$, the velocity field, $\boldsymbol{u}^n$ and the pressure field $p^{n-1/2}$, the process of computing the new velocity $\boldsymbol{u}^{n+1}$, pressure field $p^{n+1/2}$ and the location of the control points $\boldsymbol{X}^{n+1}$ can be summarized as follows:

Step 1: Set $k := 0$ and make an initial guess for $\boldsymbol{X}^{n+1}$, i.e. $\boldsymbol{X}^{(0)}$ as

$$\boldsymbol{X}^{(0)}=2\boldsymbol{X}^n-\boldsymbol{X}^{n-1}.$$

Step 2: Evaluate the forces over all elements using Eq. (42).

Step 3:

- Distribute the forces to the nearby Cartesian grid points using (6).
- Solve the Navier–Stokes equations using the projection method as described in Section 2.2.
- Compute the velocity at the control points, $\boldsymbol{u}^{n+1}(\boldsymbol{X}^{(k)})$ by interpolating from the velocity at the surrounding grid points using (14).

Step 4:

- Evaluate

$$\boldsymbol{g}(\boldsymbol{X}^{(k)})=\boldsymbol{X}^{(k)}-\boldsymbol{X}^n-\frac{1}{2}\Delta t(\boldsymbol{u}^n(\boldsymbol{X}^n)+\boldsymbol{u}^{n+1}(\boldsymbol{X}^{(k)})).$$

- If $\|\boldsymbol{g}^{(k)}\|<\epsilon$ then $\boldsymbol{X}^{n+1}=\boldsymbol{X}^{(k)}$ and stop the iteration. Otherwise, update $\boldsymbol{X}^{(k+1)}$ by solving (33) using GMRES. Note that the matrix–vector product required by GMRES is approximated as show in (36). Set $k := k+1$ and go to step 2.

We note that solving the Navier–Stokes equations using the projection method involves solving three Helmholtz equations for the intermediate velocity and one Poisson equation for the pressure-increment. These equations are solved using a Fast Fourier Transform algorithm. For the numerical computations presented we employ the subroutines from the FISHPACK library [1,45] especially adapted to our grid arrangement.

4. Numerical results

In this section, we present some numerical experiments involving membrane capsules immersed in a three-dimensional fluid domain. The biconcave-shape membranes are governed by the Yeoh form of the strain energy function (23) while the other capsules are modeled as neo-Hookean membranes. The relative tolerance for stopping the GMRES solver is chosen to be 10^{-3} and the Newton iteration converges when $\|\boldsymbol{g}^{(k)}\|/\|\boldsymbol{X}\|<10^{-8}$. All the calculations in this section were performed on a Dual-core AMD Opteron 2 GHz Processor. Most simulations were completed within a couple of minutes to hours.

4.1. Oscillatory membrane

Our first numerical experiment involves an elastic membrane initially placed in a still fluid. The initial configuration of the membrane is an ellipsoid,

$$\frac{X^2}{a^2}+\frac{Y^2}{b^2}+\frac{Z^2}{c^2}=1, \tag{43}$$

with semi-axes $(a,b,c)=(0.25,0.22,0.2)$. The domain of the experiment is the $1\times1\times1$ cube. The domain is discretized with a $64\times64\times64$ uniform grid. The membrane is located at the center of the domain and is discretized into 8192 quadratic triangular elements connecting 16,386 nodes distributed over the membrane. The initial configuration of the membrane is stretched from an undeformed reference configuration which is a spherical membrane of radius 0.2. Because of the

incompressibility condition, we expect that the membrane should perform damped oscillations and relax to the spherical equilibrium state of radius $r = \sqrt[3]{abc} \approx 0.2224$. We start our simulation by setting the velocity and pressure fields to zero, and a homogeneous Dirichlet boundary condition for the velocity is applied at all boundaries. We assume that the surface is a neo-Hookean membrane with the surface elastic modulus $E_s = 0.1$. In this simulation, the fluid density is $\rho = 1$, the fluid viscosity is $\mu = 0.01$ and the time step is $\Delta t = \Delta x$. With the Jacobian-free Newton–Krylov implicit method, we are able to increase the time step further without loss of stability. Fig. 2(a) and (b) show the number of iterations required at each time step to run the simulations to $t = 10$ using $\Delta t = \Delta x$ and $\Delta t = 4\Delta x$, respectively. It can be seen that we only need 1–4 Newton and GMRES iterations for both time steps. At the first 10 time steps, the Newton and GMRES iterations are zeros because we use a very small time step initially to start the simulation.

Fig. 3 shows the configurations of the elastic membrane at different times. The evolution of the three axes of the ellipsoidal membrane is shown in Fig. 4. It can be seen that the membrane oscillates before settling down to the equilibrium state. Fig. 5 shows the time evolution of the three semi-axes of the membrane at $\mu = 0.1$. In this case, the Reynolds number is smaller and the membrane relaxes gradually to the equilibrium state without oscillations.

For sufficiently small time steps, the explicit and the JFNK implicit method are stable as expected. For larger time steps, we expect that the explicit method will become unstable. For example, when $\Delta t = 4\Delta x$ the JFNK method gives good results as

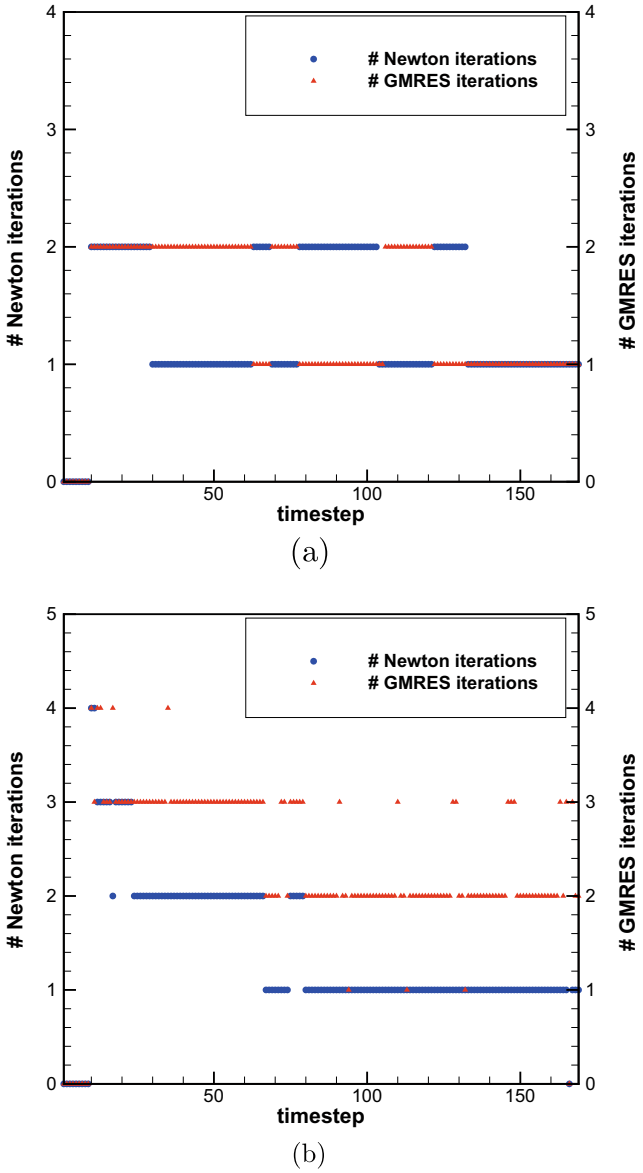


Fig. 2. Number of Newton iterations and average number of GMRES iterations performed at each time step to run the simulations to $t = 10$. (a) $\Delta t = \Delta x$ and (b) $\Delta t = 4\Delta x$.

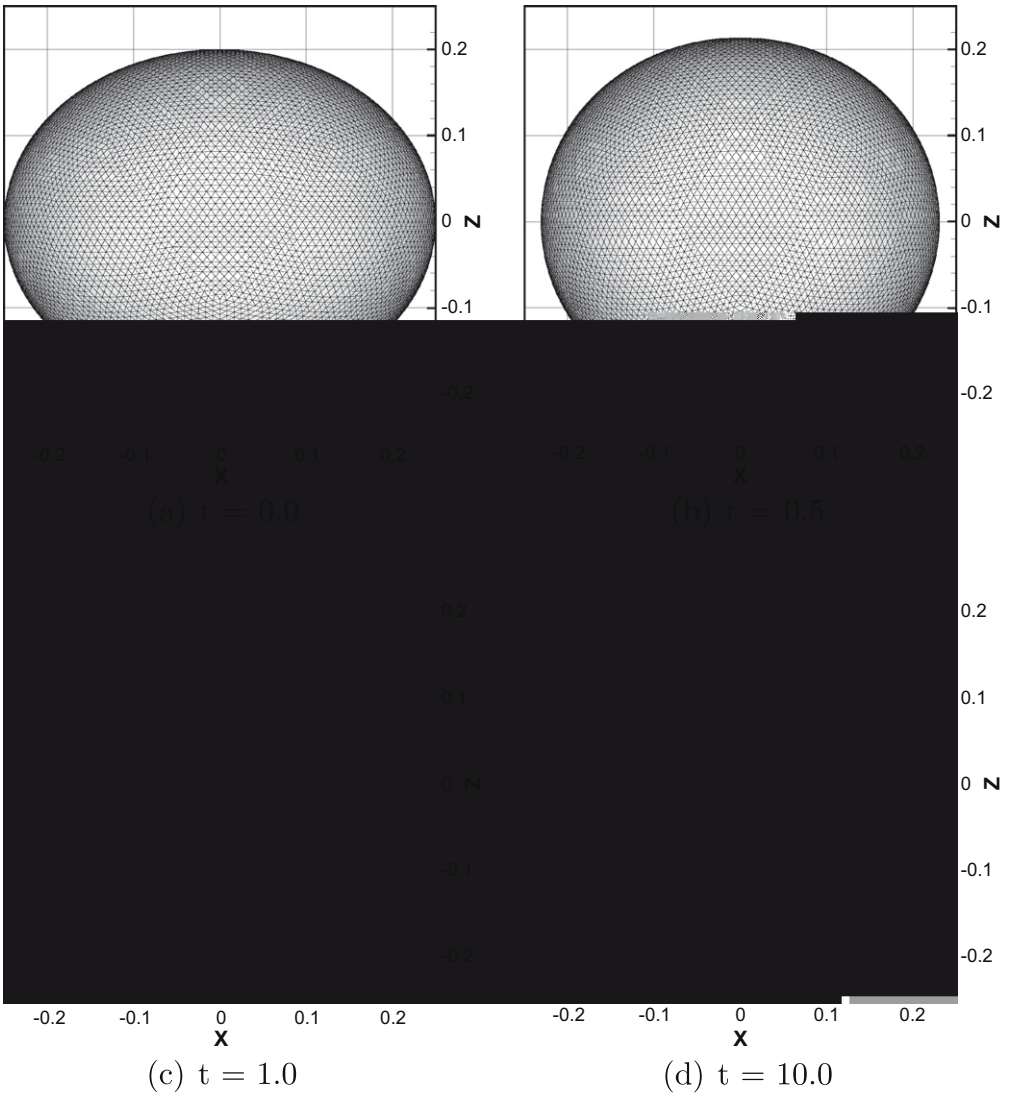


Fig. 3. Configurations of the membrane at different times.

that obtained at a smaller time step, but the explicit method is unstable. Fig. 6 shows the time evolution of the membrane for both methods before the explicit method goes unstable. The figure shows that the explicit method quickly diverges at step 12 and assumes an erroneous geometry. The JFNK method is stable and the ellipsoidal membrane gradually assumes a spherical configuration.

In this example, we study the convergence rate of our scheme in time with $\mu = 0.01, 0.05$. Following [21,33], we compute the time discretization error at time T as follows:

$$e_T(v;\Delta t) = \|v(T;\Delta t) - v(T;\Delta t/2)\|_{L^2}.$$

For a vector field $\boldsymbol{w}(\boldsymbol{x}) = (w_1(\boldsymbol{x}), w_2(\boldsymbol{x}), w_3(\boldsymbol{x}))$ defined on the Cartesian grid, the discrete L^2 norm is defined as follows

$$\|\boldsymbol{w}\|_{L^2} = \left(\sum_{i,j,k}\left(w_1^2(\boldsymbol{x}(i,j,k)) + w_2^2(\boldsymbol{x}(i,j,k)) + w_3^2(\boldsymbol{x}(i,j,k))\right)h^3\right)^{1/2}.$$

Similarly, the discrete L^2 norm for a vector field $\boldsymbol{W}(\boldsymbol{X}) = (W_1(\boldsymbol{X}), W_2(\boldsymbol{X}), W_3(\boldsymbol{X}))$ defined on the interface $\Gamma(t)$ is defined as follows

$$\|\boldsymbol{W}\|_{L^2} = \left(\sum_{l}\left(W_1^2(\boldsymbol{X}_l) + W_2^2(\boldsymbol{X}_l) + W_3^2(\boldsymbol{X}_l)\right)\Delta q\Delta r\Delta s\right)^{1/2}.$$

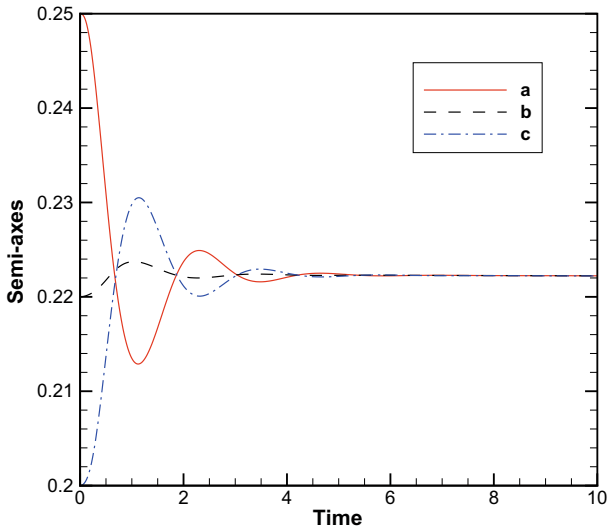


Fig. 4. The evolution of the membrane axes with $\mu = 0.01$. The membrane oscillates as it converges to the equilibrium state.

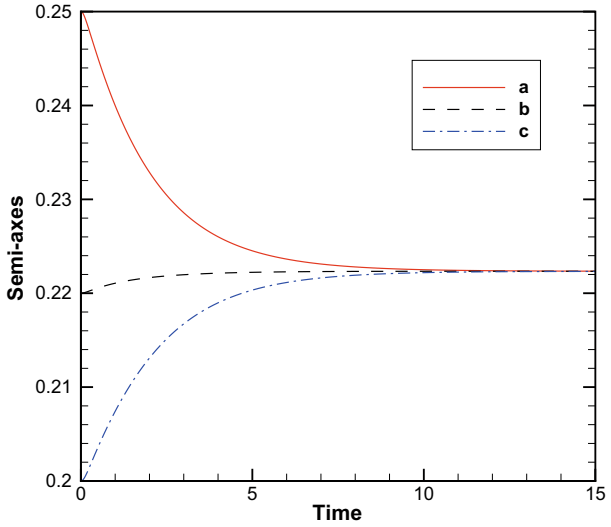


Fig. 5. The evolution of the membrane axes with $\mu = 0.1$. The membrane relaxes gradually to the equilibrium state without oscillations.

We calculate the convergence rate in time at $T = 1$ by varying the time steps in powers of 2 from $\Delta t = 1/16$ to $\Delta t = 1/128$. The results are shown in Table 1. We can see the second-order convergence in time for both the velocity field and the interface location.

4.2. Membrane capsule in simple shear flow

4.2.1. Spherical membrane

In this example, we study the deformation of a spherical neo-Hookean membrane in a shear flow given by the velocity $\boldsymbol{u} = (\mathrm{k}z, 0, 0)$, where k is the shear rate. We compare the results with the linear theory [2] and those obtained by the boundary element method (BEM) [42]. A spherical membrane of radius a is discretized into 5120 quadratic triangular elements connecting 10,242 nodes. This spherical shape is taken to be a strain free state and the shape factors for each element are obtained from this undeformed configuration. The radius of the spherical membrane is chosen to be sufficiently small so that the Reynolds number of the flow is effectively zero; in this work, it is set at $Re = 0.001$. Hence, the inertia effect is negligible and the results can be compared with those obtained by the linear theory [2] or the BEM [40,42]. The center of the membrane is placed at the center of a cube whose side is $10a$ and the cube is discretized with $64 \times 64 \times 64$ uniform grid. The computational domain is chosen to be large enough so that boundary effects are not important. Boundary conditions for the velocity

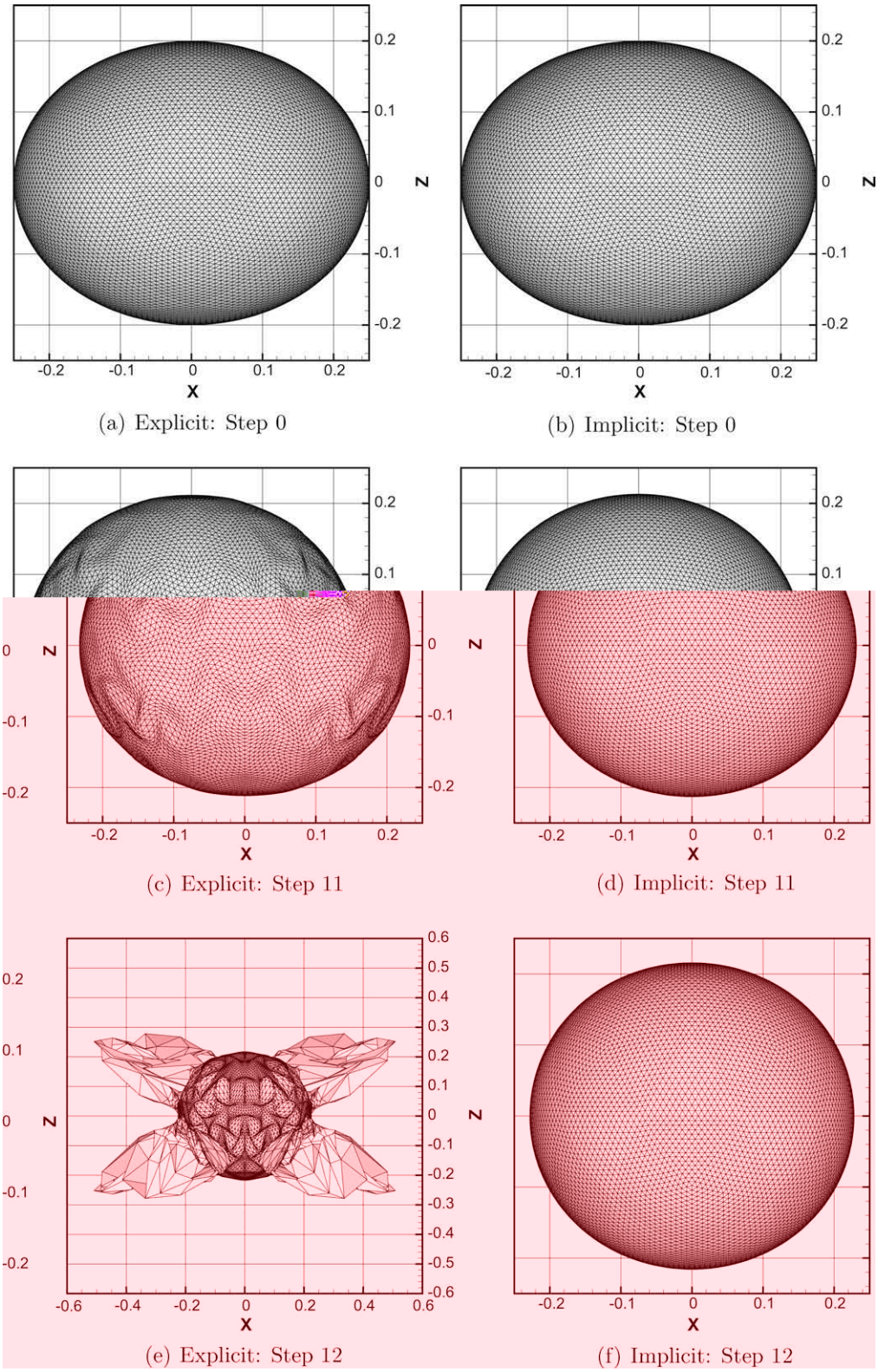


Fig. 6. Configurations of the membrane at different times with explicit method and implicit method.

are of the Dirichlet type at $z = \pm 5a$ and periodic at other boundaries. The deformation of the membrane is described by the Taylor shape parameter $D_{xz} = (L - B)/(L + B)$, where L and B are the maximum and minimum radial distances from the origin in the plane of shear, respectively. We consider the deformation of the membrane for a wide range of dimensionless shear rates,

$$G = \frac{\mu \dot{k} a}{E_s}, \tag{44}$$

Table 1
Numerical errors of $\boldsymbol{X}$ and $\boldsymbol{u}$ with different time steps.

	μ	$\Delta t = 1/16$	$\Delta t = 1/32$	$\Delta t = 1/64$	Convergence rate
$\boldsymbol{X}$	0.01	1.3600×10^{-4}	3.3555×10^{-5}	8.3011×10^{-6}	2.02
	0.005	2.9613×10^{-4}	6.2315×10^{-5}	1.4922×10^{-5}	2.06
$\boldsymbol{u}$	0.01	4.2159×10^{-4}	1.0016×10^{-4}	2.4759×10^{-5}	2.02
	0.005	8.6409×10^{-4}	2.0554×10^{-4}	5.0726×10^{-5}	2.02

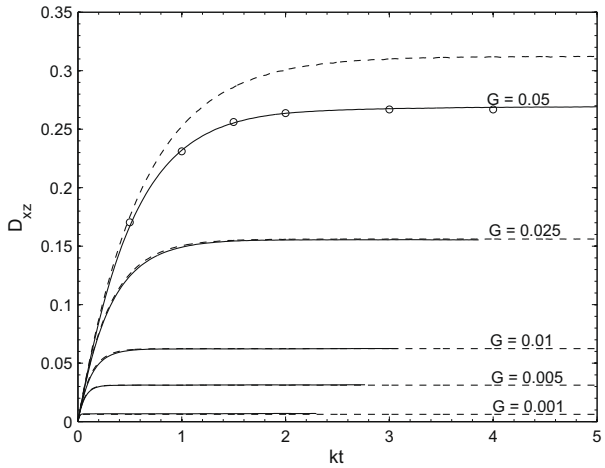


Fig. 7. The evolution of the Taylor shape parameter D_{xz} at a sequence of dimensionless shear rates G. The solid lines are the results obtained by the present algorithm, the dashed lines are obtained with the linear theory, and the circles are found using the BEM [42].

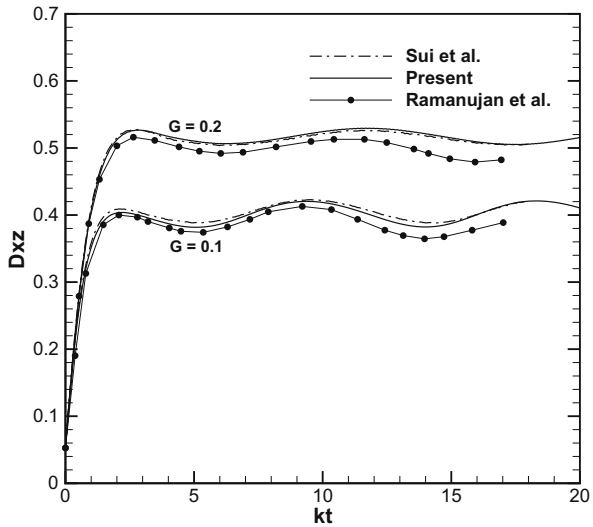


Fig. 8. The evolution of the Taylor shape parameter D_{xz} for oblate spheroidal capsules with $b/a = 0.9$ at different dimensionless shear rates G.

those calculate using the BEM [42] and the Lattice Boltzmann method (LBM) [50] under the same conditions. A $64 \times 64 \times 64$ fluid grid and membrane discretization of 10,242 nodes and 5120 quadratic triangular elements was employed. The deformation parameter D_{xz} is calculated for two shear rates $G = 0.1$ and 0.2 and is shown in Fig. 8 as solid lines. Unlike the spherical capsule, the oblate spheroid capsule undergoes oscillations in the deformation parameter. This has also been observed in [42,50]. Fig. 8 shows good agreement between our results and those obtained in [50]. The BEM [42] predicts smaller deformations than the present method and the LBM [50].

Next, we consider an oblate spheroid capsule with more-oblate unstressed shape of aspect ratio $b/a = 0.5$, inclined at the angle $\theta_0 = \pi/4$ with respect to the streamlines of the unperturbed flow. The deformation parameter D_{xz} is calculated for $G = 0.2$ and is shown in Fig. 9 as a solid line. The oblate spheroid capsule undergoes oscillation with larger amplitude in the deformation parameter. Good agreement between our result and that obtained in [50] can be seen in Fig. 9. And again, the BEM [42] predicts smaller deformations than the present method and the LBM [50].

To determine if numerical errors lead to the differences between the present method and the BEM, a grid refinement study was conducted for the oblate spheroid capsule with aspect ratio $b/a = 0.9$ at $G = 0.2$. The results are shown in Fig. 10. First a coarser fluid grid, $64 \times 64 \times 64$ with a coarser membrane discretization (4098 nodes, 2048 elements) was used. Then, on the $96 \times 96 \times 96$ fluid grid, a finer membrane discretization (10,242 nodes, 5120 elements) was employed.

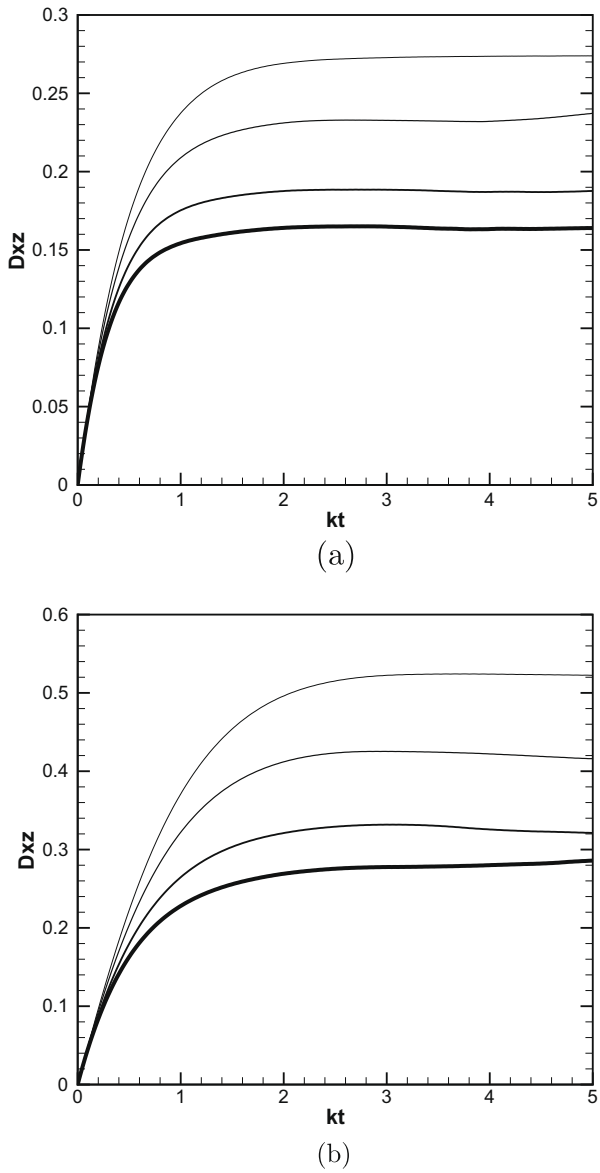
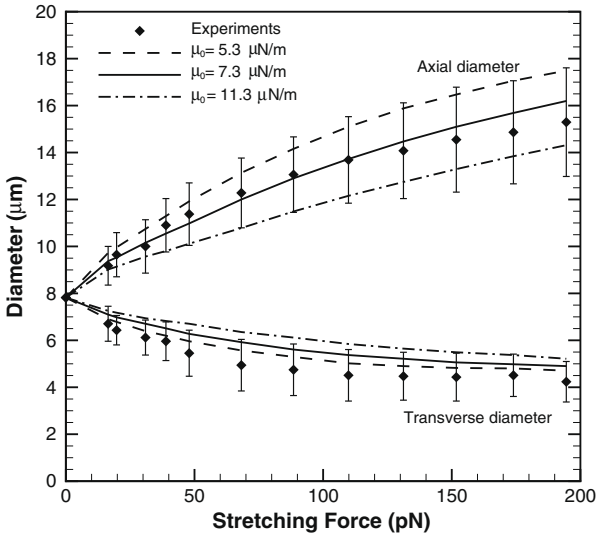


Fig. 11. The evolution of the Taylor shape parameter D_{xz} for capsules with spherical initial shapes and reduced modulus of bending. (a) $\hat{\kappa}_B = 0$, 0.01, 0.025 and 0.0375 at $G = 0.05$. (b) $\hat{\kappa}_B = 0$, 0.04, 0.1 and 0.15 at $G = 0.2$. The thickness of the lines increases as the bending modulus is raised.

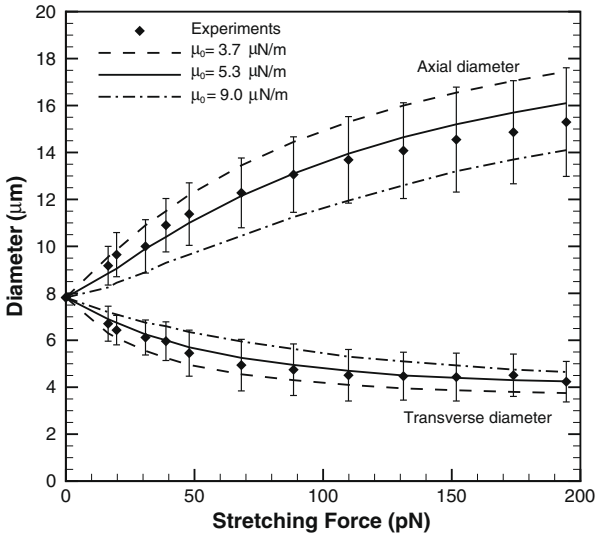
Finally, the simulation was performed on the $128 \times 128 \times 128$ fluid grid with the finest membrane discretization (16,386 nodes, 8192 elements). The results show that the numerical error using the implicit immersed boundary method for the oblate spheroid capsule with aspect ratio $b/a = 0.9$ at $G = 0.2$ using these meshes, is small.

4.2.3. Effect of membrane bending stiffness

Bending stiffness is believed to play a central role in determining the equilibrium configuration and the shape oscillations of biological membranes. Here, we consider the effect of membrane bending stiffness on the flow-induced deformation of spherical capsules in simple shear flows. We expect that the bending stiffness will restrict the overall capsule deformation. The deformation of the capsule is determined by the dimensionless shear rate G and the reduced ratio of the bending modulus to the elastic modulus, $\hat{\kappa}_B \equiv \kappa_B/(a^2 E_S)$. The reference state for computing the bending moment is the membrane with flat resting shape, corresponding to zero reference curvature. The reference configuration for calculating the in-plane elastic tension is the membrane of an initially spherical capsule of radius a. We note that the reference state of the bending moments is not necessarily the same as that of the elastic tensions, reflecting differences in the physical mechanisms that are responsible for their respective development [40]. Simulations were performed for spherical capsules with different



(a)



(b)

Fig. 12. Variation of the axial and transverse diameters of the red blood cell against the stretching force. (a) Computational results from [32] using the finite element program ABAQUS. (b) Computational results from [8] using the finite element code ADINA 8.3. The experimental results are taken from [32].

shear rates and reduced bending modulus. Fig. 11(a) shows a family of Taylor shape parameters for reduced bending modulus $\hat{\kappa}_B = 0, 0.01, 0.025$ and 0.0375 at the low shear rate of $G = 0.05$. Fig. 11(b) illustrates a family of Taylor shape parameters for reduced bending modulus $\hat{\kappa}_B = 0$, 0.04, 0.1 and 0.15 at the higher shear rate of $G = 0.2$. We note that the line thickness increases as the magnitude of the bending modulus increases. The Taylor shape parameters reveal that spherical capsules deform and reach stationary shapes and that raising the bending modulus reduces the capsule deformation.

4.3. Red blood cell deformed by optical tweezers

Next, we consider the deformation of human red blood cells subjected to direct stretching by optical tweezers. Numerical simulations were performed to extract the large deformation elastic properties from the experimental results obtained in [9,30,32] during loading as well as upon relaxation of the load. Direct tensile stretching of the human red blood cell using optical tweezers to extract elastic properties was first reported by Henon et al. [19] who attached two silica beads non-specifically to diametrically opposite ends of the cell, trapped both beads with laser beams, and imposed tensile elastic deformation on the cell by moving the trapped beads in opposite directions. In [9,30,32], the optical tweezers system is designed to consist only of a single optical trap, one of the beads is adhered to the glass surface while the other is free to be trapped using the laser beam. By moving one of the beads with the laser beam, the cell is directly stretched.

Fig. 12(a) shows the variation of axial and transverse diameters of the red blood cell against the stretching force obtained by both numerical simulations and experimental measurements [32]. In [32], the numerical simulations were performed using the commercially available general purpose finite element package ABAQUS for red blood cells with an initial diameter of 7.82 μm. The shape of the red blood cell is a biconcave shape given as

$$Z(R) = \pm 0.5R_0\left[1 - \left(\frac{R}{R_0}\right)^2\right]^{\frac{1}{2}}\left[C_0 + C_1\left(\frac{R}{R_0}\right)^2 + C_2\left(\frac{R}{R_0}\right)^4\right], \tag{45}$$

where $R^2 = X^2 + Y^2 \leqslant R_0^2$, $R_0 = 3.91$ μm, $C_0 = 0.207161$, $C_1 = 2.002558$, and $C_2 = -1.122762$. The contact dimension between the beads and the cell was taken to be 2 μm in the simulations. The strain energy function (23) was employed with $C_{10} = \mu_0/2\bar{h}_0$ and $C_{30} = \mu_0/30\bar{h}_0$, where μ_0 is the initial in-plane shear modulus and $\bar{h}_0$ is the initial membrane thickness. The results for the in-plane shear modulus $\mu_0 = 5.3$–11.3 μN/m are shown in Fig. 12(a). These values are comparable to the range of 4.0–10 μN/m reported in the literature where the estimates have been principally based on micropipette aspiration experiments.

Fig. 12(b) illustrates the variation of the axial and transverse diameters of the red blood cell against the stretching force obtained in [8] by numerical simulations. Computational results from [8] were obtained using the finite element code ADINA 8.3 with the same strain energy function (23). The model proposed in [8] consists of a deformable liquid capsule modeled as Newtonian fluid enclosed by a hyperelastic membrane with viscoelastic properties. The membrane shear modulus was estimated to range between 3.7 and 9.0 μN/m. This range also compares well with the reported results obtained from micropipette aspiration experiments.

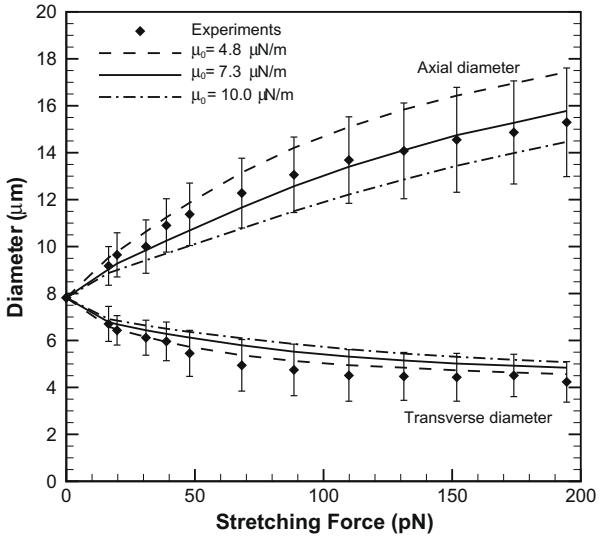


Fig. 13. Variation of the axial and transverse diameters of the red blood cell against the stretching force. The dotted, solid and dash-dotted lines represent computational results for the axial/transverse diameters with $\mu_0 = 4.8$, 7.3 and 10.0 μN/m, respectively. The experimental results are taken from [32].

In the present paper, we employ the implicit immersed boundary method to study the deformation of human red blood cells subjected to the direct stretching by optical tweezers in order to extract the membrane shear modulus from the experimental results. The tensile forces are applied uniformly on the contact areas of sizes 2 μm between the cell and the beads at opposite ends of the cell. The red blood cell of diameter 7.82 μm is discretized with 16,386 nodes and 8192 quadratic triangular elements. The red blood cell is placed initially at the center of a cube whose side is $8R_0$. The computational domain is discretized with $128 \times 128 \times 128$ uniform grid. The computational domain is filled with the fluid whose density and viscosity are given as 1050 kg m^{-3} and 4.1 mPa s. These are typical values for the red blood cell cytoplasm density and viscosity. Comparisons of predicted and measured changes in the axial and transverse diameters of the red blood cell using the (Yeoh) strain energy function (23) are plotted in Fig. 13 for $\mu_0 = 4.8$, 7.3 and 10 μN/m. The simulation is able to capture the experimental trend over the entire range of experimental data well, including the small deformation range and the error bars. The in-plane shear modulus $\mu_0 = 4.8$–10 μN/m are well comparable to the range of 4.0–10 μN/m obtained from micropipette aspiration experiments. Comparing Figs. 12 and 13, it is apparent that our results are closer to those obtained in [32] than

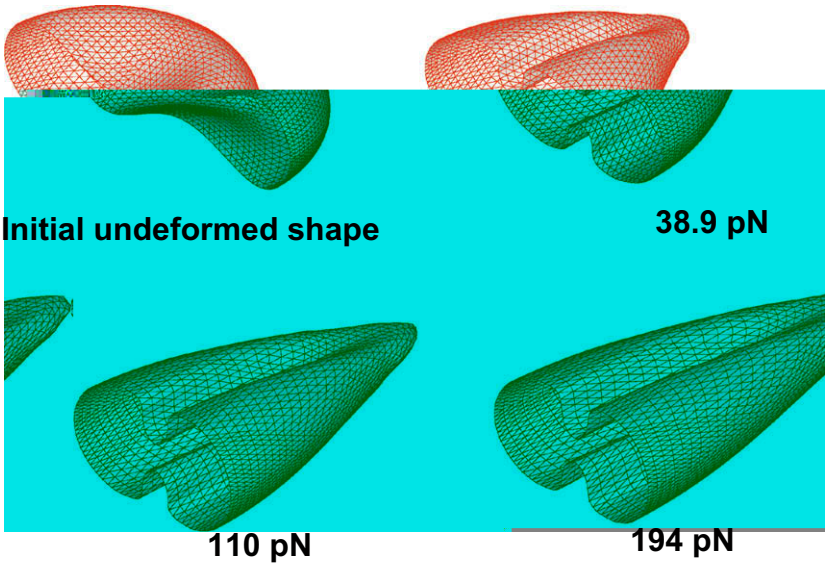


Fig. 14. The deformed shapes of the red blood cell under the stretching forces.

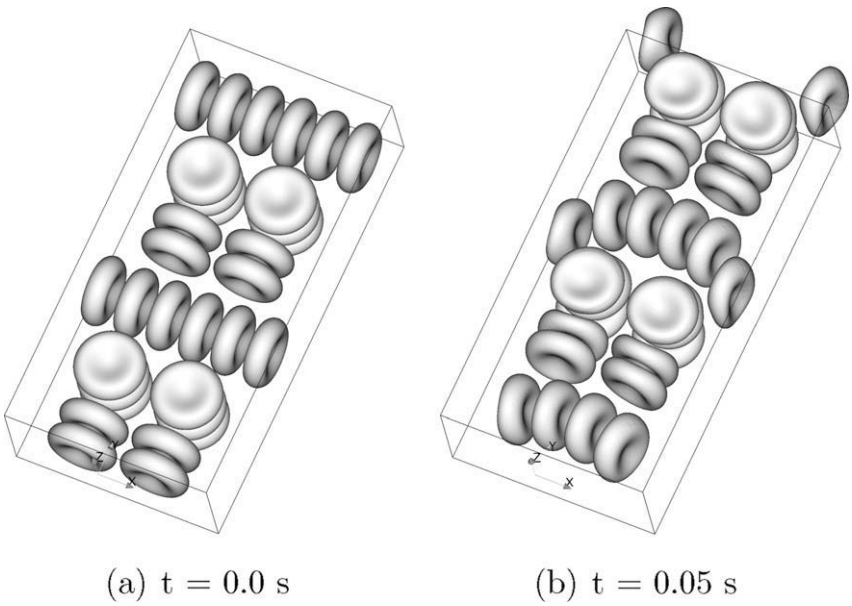


Fig. 15. Positions of the red blood cells at different times.

those in [8]. Fig. 14 illustrates the initial undeformed shape and the deformed shapes of the red blood cell under the stretching force of 38.9, 110 and 194 pN.

4.4. Red blood cells in pressure-driven flow

Using the shear modulus obtained from the previous example, we perform a simulation of the pressure-driven flow involving 32 red blood cells (RBCs) immersed in the fluid. The purpose of this simulation is to illustrate the performance of the present implicit immersed boundary method. Specifically, we compare the CPU time that is required for the present implicit scheme and the second-order Runge–Kutta scheme used in [37] to obtain a solution at a given time. In the simulation, the RBCs are initially placed in a box of size 24 μm × 48 μm × 12 μm as shown in Fig. 15(a). The computational domain is discretized with 64 × 128 × 32 uniform grid. The fluid viscosity and density are the same as those in the previous example. The pressure drop is maintained at 10 cm of water column. Each red blood cell is discretized with 10,242 nodes and 5120 quadratic triangular elements. We measure the computational times required for each scheme to obtain the solution at $t = 0.05$ s with the in-plane shear modulus of 10 μN/m. We used $\Delta t = 4.5 \times 10^{-4}$ for our implicit scheme. We can actually choose larger time step while still maintain the stability, but this choice of Δt is due to the accuracy considerations. The maximum time step that is required for the second-order Runge–Kutta scheme to maintain the stability is $\Delta t = 1.875 \times 10^{-6}$. All the simulations are performed on a Dual-core AMD Opteron 2 GHz Processor. The positions of the RBCs at $t = 0.05$ s are shown in Fig. 15(b), applicable for the two different time schemes. We consider periodic boundary conditions for the cells so that the cells which exit at one end of the channel will enter at the other end. We observe that the CPU time for our implicit scheme is about 22 min and this CPU time is about 43 times faster than that of the explicit scheme. The gain over the explicit scheme is even more substantial with larger shear modulus.

5. Conclusions

In this paper, we have presented the implicit immersed boundary method for the incompressible Navier–Stokes equations capable of handling three-dimensional membrane–fluid flow interactions. The present study was undertaken with the objective of eliminating time step restrictions by using the Jacobian-free Newton–Krylov method (JFNK) to advance the location of the elastic membrane implicitly.

We have shown the capability of the present technique by applying it to the viscous flow problems involving elastic membranes with bending stiffness in three-dimensional domains. Numerical simulations have been performed for the oscillatory membrane in viscous flow to illustrate that the method can maintain stability under relatively large time steps. Simulations have also been performed to reproduce some results for the membrane capsule in simple shear flow as a validation test for our method. It is found that our numerical results are in good agreement with those reported in literature. The present method was also used to simulate the large deformation of human red blood cells subjected to direct stretching by optical tweezers. We have compared the in-plane shear modulus of the normal human red blood cell with that determined by experimental measurements. The range of shear modulus obtained by the present method compares well to that reported previously in the literature. With the obtained shear modulus, the present implicit method has been used to perform simulations of red blood cells in the pressure-driven flow to further demonstrate the performance of the method.

The present algorithm does not include any preconditioned techniques with the GMRES solver. We expect to improve the time step further without increasing the number of GMRES iterations by applying appropriate preconditioners. This will be the subject of future work.

References

[1] J. Adams, P. Swarztrauber, R. Sweet, FISHPACK: Efficient FORTRAN Subprograms for the Solution of Separable Elliptic Partial Differential Equations, 1999. <http://www.scd.ucar.edu/css/software/fishpack/>.
[2] D. Barthes-Biesel, J.M. Rallison, The time-dependent deformation of a capsule freely suspended in a linear shear flow, J. Fluid Mech. 113 (1981) 251–267.
[3] I. Borazjani, L. Ge, F. Sotiropoulos, Curvilinear immersed boundary method for simulating fluid–structure interaction with complex 3D rigid bodies, J. Comput. Phys. 227 (2008) 7587–7620.
[4] D.L. Brown, R. Cortez, M.L. Minion, Accurate projection methods for the incompressible Navier–Stokes equations, J. Comput. Phys. 168 (2001) 464–499.
[5] P.N. Brown, Y. Saad, Hybrid Krylov methods for nonlinear systems of equations, SIAM J. Sci. Stat. Comput. 11 (1990) 450–481.
[6] X.-C. Cai, D.E. Keyes, V. Venkatakrishnan, Newton–Krylov–Schwarz: an implicit solver for CFD, in: Proceedings of the Eighth International Conference on Domain Decomposition Methods, Wiley, New York, 1997.
[7] T.F. Chan, K.R. Jackson, Nonlinearly preconditioned Krylov subspace methods for discrete Newton algorithms, SIAM J. Sci. Stat. Comput. 5 (1984) 533–542.
[8] C.Y. Chee, H.P. Lee, C. Lu, Using 3D fluid–structure interaction model to analyse the biomechanical properties of the erythrocyte, Phys. Lett. A 372 (2008) 1357–1362.
[9] M. Dao, C.T. Lim, S. Suresh, Mechanics of the human red blood cell deformed by optical tweezers, J. Mech. Phys. Solid 51 (2003) 2259–2280.
[10] R. Dillon, L.J. Fauci, D. Graver, A microscale model of bacterial swimming, chemotaxis and substrate transport, J. Theor. Biol. 177 (1995) 325–340.
[11] C.D. Eggleton, A.S. Popel, Large deformation of red blood cell ghosts in a simple shear flow, Phys. Fluid 10 (1998) 1834–1845.
[12] L.J. Fauci, A.L. Fogelson, Truncated Newton methods and the modeling of complex immersed elastic structures, Commun. Pure Appl. Math. 66 (1993) 787–818.
[13] L.J. Fauci, C.S. Peskin, A computational model of aquatic animal locomotion, J. Comput. Phys. 77 (1988) 85–108.
[14] A.L. Fogelson, A mathematical model and numerical method for studying platelet adhesion and aggregation during blood clotting, J. Comput. Phys. 56 (1984) 111–134.

[15] A.L. Fogelson, Continuum models of platelet aggregation: formulation and mechanical properties, SIAM J. Appl. Math. 52 (1992) 1089–1110.
[16] L. Ge, F. Sotiropoulos, A numerical method for solving the 3D unsteady incompressible Navier–Stokes equations in curvilinear domains with complex immersed boundaries, J. Comput. Phys. 225 (2007) 1782–1809.
[17] E. Givelberg, Modeling elastic shells immersed in fluid, Commun. Pure Appl. Math. LVII (2004) 0283–0309.
[18] Z. Gong, H. Huang, C. Lu, Stability analysis of the immersed boundary method for a two-dimensional membrane with bending rigidity, Commun. Comput. Phys. 3 (3) (2008) 704–723.
[19] S. Henon, G. Lenormand, A. Richert, F. Gallet, A new determination of the shear modulus of the human erythrocyte membrane using optical tweezers, Biophys. J. 76 (1999) 1145–1151.
[20] T.Y. Hou, J. Lowengrub, M. Shelley, Removing the stiffness from interfacial flows with surface tension, J. Comput. Phys. 114 (1994) 312–338.
[21] T.Y. Hou, Z. Shi, An efficient semi-implicit immersed boundary method for the Navier–Stokes equations, J. Comput. Phys. 227 (2008) 8968–8991.
[22] T.Y. Hou, Z. Shi, Removing the stiffness of elastic force from the immersed boundary method for the 2D Stokes equations, J. Comput. Phys. 227 (2008) 9138–9169.
[23] J. Kim, P. Moin, Application of a fractional step method to incompressible Navier–Stokes equations, J. Comput. Phys. 59 (1985) 308–323.
[24] D.A. Knoll, D.E. Keyes, Jacobian-free Newton–Krylov methods: a survey of approaches and applications, J. Comput. Phys. 193 (2004) 357–397.
[25] D.A. Knoll, V.A. Mousseau, On Newton–Krylov multigrid methods for the incompressible Navier–Stokes equations, J. Comput. Phys. 163 (2000) 262–267.
[26] D.V. Le, B.C. Khoo, K.M. Lim, An implicit-forcing immersed boundary method for simulating viscous flows in irregular domains, Comput. Method Appl. Mech. Eng. 197 (2008) 2119–2130.
[27] D.V. Le, B.C. Khoo, J. Peraire, An immersed interface method for viscous incompressible flows involving rigid and flexible boundaries, J. Comput. Phys. 220 (2006) 109–138.
[28] D.V. Le, C. Rosales, B.C. Khoo, J. Peraire, Numerical design of electro-mechanical traps, Lap Chip 8 (2008) 755–763.
[29] R.J. LeVeque, Z. Li, Immersed interface method for Stokes flow with elastic boundaries or surface tension, SIAM J. Sci. Comput. 18 (3) (1997) 709–735.
[30] C.T. Lim, M. Dao, S. Suresh, C.H. Sow, K.T. Chew, Large deformation of living cells using laser traps, Acta Mater. 52 (2004) 1837–1845.
[31] A. Mayo, C.S. Peskin, An implicit numerical method for fluid dynamics problems with immersed elastic boundaries, Contemp. Math. 141 (1993) 261–277.
[32] J.P. Mills, L. Qie, M. Dao, C.T. Lim, S. Suresh, Nonlinear elastic and viscoelastic deformation of the human red blood cell with optical tweezers, Mech. Chem. Biosyst. 1 (3) (2004) 169–180.
[33] Y. Mori, C.S. Peskin, Implicit second-order immersed boundary methods with boundary mass, Comput. Method Appl. Mech. Eng. 197 (2008) 2049–2067.
[34] E.P. Newren, Unconditionally stable discretizations of the immersed boundary equations, J. Comput. Phys. 222 (2007) 702–719.
[35] E.P. Newren, A.L. Fogelson, R.D. Guy, R.M. Kirby, A comparison of implicit solvers for the immersed boundary equations, Comput. Method Appl. Mech. Eng. 197 (2008) 2290–2304.
[36] C.S. Peskin, Numerical analysis of blood flow in the heart, J. Comput. Phys. 25 (1977) 220–252.
[37] C.S. Peskin, The immersed boundary method, Acta Numer. 11 (2) (2002) 479–517.
[38] C. Pozrikidis, Finite deformation of liquid capsules enclosed by elastic membranes in simple shear flow, J. Fluid Mech. 297 (1995) 123–152.
[39] C. Pozrikidis, Numerical Computation in Science and Engineering, Oxford University Press, 1998.
[40] C. Pozrikidis, Effect of membrane bending stiffness on the deformation of capsules in simple shear flow, J. Fluid Mech. 440 (2001) 269–291.
[41] C. Pozrikidis, Numerical simulation of the flow-induced deformation of red blood cells, Ann. Biomed. Eng. 31 (2003) 1194–1205.
[42] S. Ramanujan, C. Pozrikidis, Deformation of liquid capsules enclosed by elastic membrane in simple shear flow: large deformations and the effect of fluid viscosities, J. Fluid Mech. 361 (1998) 117–143.
[43] A.M. Roma, C.S. Peskin, M.J. Berger, An adaptive version of the immersed boundary method, J. Comput. Phys. 153 (1999) 509–534.
[44] Y. Sadd, GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems, SIAM J. Sci. Stat. Comput. 7 (1986) 856–869.
[45] U. Schumann, R.A. Sweet, A direct method for the solution of Poisson's equation with Neumann boundary conditions on a staggered grid of arbitrary size, J. Comput. Phys. 20 (1976) 171–182.
[46] R. Skalak, A. Tozeren, R.P. Zarda, S. Chien, Strain energy function of red blood cell membranes, Biophys. J. 13 (1973) 245–264.
[47] J.M. Stockie, B.R. Wetton, Stability analysis for the immersed fiber problem, SIAM J. Appl. Math. 55 (1995) 1577–1591.
[48] J.M. Stockie, B.R. Wetton, Analysis of stiffness in the immersed boundary method and implications for time-stepping schemes, J. Comput. Phys. 154 (1999) 41–64.
[49] J. Stoer, R. Bulirsch, Introduction to Numerical Analysis, third ed., Springer-Verlag, 2002.
[50] Y. Sui, Y.T. Chew, P. Roy, H.T. Low, A hybrid method to study flow-induced deformation of three-dimensional capsules, J. Comput. Phys. 227 (2008) 6351–6371.
[51] C. Tu, C.S. Peskin, Stability and instability in the computation of flows with moving immersed boundaries: a comparison of three methods, SIAM J. Sci. Statist. Comput. 13 (1992) 1361–1376.
[52] N.T. Wang, A.L. Fogelson, Computational methods for continuum models of platelet aggregation, J. Comput. Phys. 151 (1999) 649–675.
[53] X.S. Wang, An iterative matrix-free method in implicit immersed boundary/continuum methods, Comput. Struct. 85 (2007) 739–748.
[54] O.H. Yeoh, Characterization of elastic properties of carbon-black-filled rubber vulcanizates, Rubber Chem. Technol. 63 (1990) 792–805.