Journal of Computational Physics 509 (2024) 113042

Contents lists available at ScienceDirect

Journal of Computational Physics

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

Numerical simulation of an extensible capsule using regularized Stokes kernels and overset finite differences

Dhwanit Agarwal *, George Biros

Oden Institute, University of Texas at Austin, Austin, TX, 78712, United States

ARTICLE INFO

Keywords:
Stokesian particulate flows
Integral equations
Numerical methods
Fluid membranes
Extensible capsules
Fast summations methods

ABSTRACT

In this paper, we present a novel numerical scheme for simulating deformable and extensible capsules suspended in a Stokesian fluid. The main feature of our scheme is a partition-of-unity (POU) based representation of the surface that enables asymptotically faster computations compared to spherical-harmonics based representations. We use a boundary integral equation formulation to represent and discretize hydrodynamic interactions. The boundary integrals are weakly singular. We use the quadrature scheme based on the regularized Stokes kernels by Tlupova and Beale 2019 (given in [34]). We also use partition-of unity based finite differences that are required for the computation of interfacial forces. Given an N-point surface discretization, our numerical scheme has fourth-order accuracy and $\mathcal{O}(N)$ asymptotic complexity, which is an improvement over the $\mathcal{O}(N^2 \log N)$ complexity of a spherical harmonics based spectral scheme that uses product-rule quadratures by Veerapaneni et al. 2011 [36]. We use GPU acceleration and demonstrate the ability of our code to simulate the complex shapes with high resolution. We study capsules that resist shear and tension and their dynamics in shear and Poiseuille flows. We demonstrate the convergence of the scheme and compare with the state of the art.

Contents

* Corresponding author.
E-mail addresses: dhwanit16@gmail.com (D. Agarwal), biros@oden.utexas.edu (G. Biros).

https://doi.org/10.1016/j.jcp.2024.113042
Received 20 October 2023; Received in revised form 7 April 2024; Accepted 19 April 2024

Available online 24 April 2024

1. Introduction

Capsules suspended in a Stokesian fluid describe complex biological flows and microfluidic devices [11,20,37]. In particular, we're interested in modeling red blood cell (RBC) suspensions. Several previous works [7,12,19,23,31,44] have modeled red blood cells as elastic membranes filled with a Newtonian fluid and suspended in a Newtonian fluid, also referred to as *"capsules"*. In this work, we present a numerical scheme for simulating a single deformable three-dimensional capsule suspended in Stokes flow where its membrane resists shear and tension [29,32] (see Fig. 1).

The proposed scheme allows for non-uniform discretization and fast evaluations of integral operators. We a singular quadrature based on the regularized Stokes kernel introduced in [34]. We use multiple overlapping patches to parameterize the capsule surface assuming a spherical topology surface at all times. We use a fourth-order finite difference scheme using an overset grid based discretization of the patches to calculate the interfacial elastic forces. For N number of discretization points of the surface, our scheme has $O(N^2)$ work complexity, which is similar to the lower order boundary element schemes [10]. By choosing the regularization parameter in [34] appropriately, our overall scheme becomes fourth-order accurate. As our singular quadrature scheme is not a product quadrature, which allows acceleration via fast multipole methods (FMMs) [35]. With FMM acceleration, our numerical scheme becomes an $O(N)$ scheme. In summary our contributions are as follows:

- We describe an overlapping patch based parameterization for capsules that are diffeomorphic to the unit sphere.
- We provide a finite difference scheme based on the overset grid based discretization of these patches to calculate the surface derivatives.
- We evaluate a high order singular quadrature scheme based on the regularized Stokes kernels given in [34]; combined with FMM it has an $O(p^2)$ cost for evaluating the single layer Stokes kernel.
- We use GPU acceleration for the evaluation of singular quadrature to do high resolution simulations.

Related work: Stokesian flows with deformable capsules involve moving interfaces, fluid-structure interaction, large deformations, near and long-range stiff interactions, and often require high-order surface derivatives. Efficient methods for simulating Stokesian capsule suspensions include immersed boundary/interface methods [2], lattice-Boltzmann methods [22], and boundary integral equations methods [10,27,36,44]. Each of these methods trades-off aspects of accuracy, simplicity of implementation, the ability to simulate complex physics, and numerical efficiency. Without advocating a particular approach, we focus on integral equation formulations and discuss the related literature in more detail.

A nice feature of boundary integral formulations for Stokes flows is that they require only the discretization of the capsule surface. But they require specialized singular quadrature schemes to compute layer potentials. The methods described in [36,44] use spherical harmonics representations for the surface and surface fields, *e.g.,* elastic forces, velocity, and shape, to simulate the time evolution of the surface. This surface representation is spectrally accurate in the degree p of the spherical harmonics that translates to $O(p^2)$ discretization points. The authors [44] used the Bruno-Kunyansky quadrature [6,43] that uses a floating partition of unity to resolve weakly singular points around each global quadrature point. This scheme has $O(p^4 \log p)$ work complexity and can be reduced to $O(p^3)$ using an FFT-based scheme that embeds the surface [6] in a volumetric Cartesian grid. The authors in [36] adapted the Graham-Sloan quadrature [17] to the Stokes kernel. Given $\mathcal{O}(p^2)$ global points to represent the capsule, the scheme uses a product quadrature rule that requires $\mathcal{O}(p^2)$ spherical harmonics rotations. For small p, the fastest implementation uses dense linear algebra

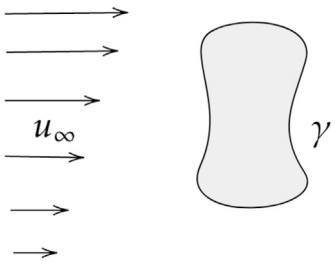


Fig. 1. A representation of the problem setup. The grey filled region is the interior of the capsule with membrane γ. Exterior of the capsule is filled with a Newtonian fluid and the capsule is suspended freely in it. $\boldsymbol{u}_\infty$ is the imposed background fluid velocity.

at a $\mathcal{O}(p^6)$ cost; this becomes prohibitively expensive for large scheme. Fast rotation schemes based on Legendre transforms and FFTs or combinations of 1D non-uniform FFTS can lower the cost to $O(p^4 \log p)$ [16]. Either way the product-quadrature has excellent accuracy, and even for small values of p for nearly-spherical shapes it outperforms the Bruno-Kunyansky quadrature. This is due to Bruno-Kunyansky's highly localized floating-partition of unity that increases the resolution requirements. However, the product rule is too expensive for large p due to the p^4 scaling. For example, resolving shapes with high curvature, like the shapes shown in Fig. 13c and Fig. 16c, requires $p > 128$, which is prohibitively expensive for the product rule. The are two alternatives: change the surface representation and/or change the integration rule.

The first alternative is to avoid using global polynomials for the surface representation. Several studies [4,5,10,33,45] have used triangulation based surface representation to provide more control over local resolution. However, these schemes are only second-order accurate, which makes the computation of higher order derivatives more difficult and complicate the quadrature rules because the most accurate methods require geometry-dependent rule precomputation and this doesn't work for time evolving geometries.

The second approach is to change the quadrature rule. Indeed a very promising approach is the quadrature by expansion (QBX) family of methods [1,18,38]. These methods are also spectrally accurate. Direct implementations scale as p^4 but they can be accelerated with FMM to p^2. Two alternative schemes to QBX are the arbitrary-order weight-corrected trapezoidal rule [41] and the regularized Stokes kernel [34]. The former is arbitrary-order accurate but a bit involved to implement. The latter that is ∼fourth-order accurate and relatively easy to implement.

Limitations: (i) Our scheme only works for capsules that can be smoothly mapped to the unit sphere. We do not consider the bending elastic forces [10,28,36] but our code can be extended to support them as is. (ii) We do not consider the case of viscosity contrast, i.e., when the fluid inside and outside the capsule has different viscosity. Our algorithm can be extended to include differences in the interior and exterior fluid viscosity by including a double layer Stokes potential term in the problem formulation as described in detail in [28,30] combined with the regularized Stokes double layer potential from [34]. (iii) Our scheme has several parameters that can affect its accuracy, for example, the configuration of patches, the extent of overlap of the patches, the partition of unity functions, and a regularization parameter for the singular quadrature, whose values we set heuristically. (iv) Our scheme is not curvature adaptive Instead of a regular grid, we eventually want to switch to a quadtree-based adaptive representation. (v) Our current implementation's accuracy is set by the regularized kernel. We observe the theoretical predictions for smooth shapes. For more complex shapes the asymptotic regime is not reached at the resolutions we're interested in and therefore upsampling is needed. Such an upsampling introduces a large constant in the complexity estimate. The QBX and locally corrected trapezoidal rules can be used to deliver arbitrary accuracy and they're also compatible with FMM. The schemes would deliver optimal accuracy and complexity. Although the discussion in this paper is about a single capsule, the scheme can be extended using existing techniques to simulating an arbitrary number of capsules [36] and confined boundaries.

Outline of the paper: In the next section, we state the force differential operators and the boundary integral formulation of the dynamics of an extensible capsule suspended in Stokes flow. In Section 3, we describe the surface parameterization and the numerical schemes (differentiation, integration and time stepping) we use to simulate the capsule dynamics. In Section 4, we report the numerical results demonstrating the convergence and accuracy of the numerical schemes along with the results of capsule dynamics in shear and Poiseuille flow. In Section 5, we discuss the FMM based acceleration of our scheme and show results for the speedup obtained via acceleration using a single level FMM. In Section 6, we provide a brief summary of the paper along with the conclusions and the future directions that can be taken to improve upon this work.

2. Problem formulation

In this section, we formally describe the mathematical formulation of the problem and introduce the notation. We discuss the differential formulation in Section 2.1 and then specify the boundary integral formulation of the problem in Section 2.2. A representation of the setup is shown in Fig. 1. Our discussion follows the work in [27,36].

2.1. Formulation

Let γ be the membrane of an extensible capsule filled with a viscous Newtonian fluid of viscosity μ_i and suspended in another viscous Newtonian fluid of viscosity μ_e. In this work, we assume that the interior and exterior fluids have same viscosity, *i.e.,*

$\mu_i = \mu_e = \mu$. The microscopic length scale of the problem implies that the dynamics of the problem can be described by the Stokes equation. The boundary value problem is given by [36]:

$$-\mu\Delta\boldsymbol{u}(\boldsymbol{x}) + \nabla p(\boldsymbol{x}) = 0 \quad \forall \boldsymbol{x} \in \mathbb{R}^3\backslash\gamma, \tag{2.1}$$

$$\nabla \cdot \boldsymbol{u}(\boldsymbol{x}) = 0 \quad \forall \boldsymbol{x} \in \mathbb{R}^3\backslash\gamma, \tag{2.2}$$

$$[[-p\boldsymbol{n} + \mu(\nabla\boldsymbol{u} + \nabla\boldsymbol{u}^T)\boldsymbol{n}]] = \boldsymbol{f} \quad on\ \gamma, \tag{2.3}$$

$$\boldsymbol{u}(\boldsymbol{x}) \longrightarrow \boldsymbol{u}_\infty \quad for\ \boldsymbol{x} \longrightarrow \infty, \tag{2.4}$$

$$\frac{\partial \boldsymbol{x}}{\partial t} = \boldsymbol{u} \quad on\ \gamma. \tag{2.5}$$

Here μ is the viscosity of the exterior and interior fluid, $\boldsymbol{u}$ is the velocity of the fluid, p is the pressure, $[[q]]$ represents the jump in a quantity q across the capsule membrane γ, $\boldsymbol{n}$ is the unit normal to the capsule membrane, $\boldsymbol{f}$ is the total force exerted by the capsule membrane onto the fluid, $\boldsymbol{u}_\infty$ is the imposed background velocity far away from the capsule. Equation (2.1) is the Stokes equation representing the conservation of momentum, Equation (2.2) is the conservation of mass, Equation (2.3) is the balance of force on the interface between fluid and membrane, Equation (2.4) sets the far field velocity to be the background velocity and Equation (2.5) enforces the no-slip condition on the capsule membrane.

Interfacial force: Now we discuss in detail the interfacial force $\boldsymbol{f}$ exerted by capsule membrane onto the surrounding fluid due to membrane elasticity. We only consider the elastic forces due to in-plane shear deformations of the capsule. Previous works have also included bending forces [10,40,44] but we leave it for future work and focus mainly on the numerical schemes in this paper. The in-plane shear force $\boldsymbol{f}_s$ is equal to the surface divergence of the symmetric part of the in-plane shear stress tensor denoted by Λ [29,44]. Thus, we write,

$$\boldsymbol{f} = \boldsymbol{f}_s = \nabla_\gamma \cdot \Lambda. \tag{2.6}$$

In-plane shear stress tensor Λ*:* Our discussion follows the work based on the Skalak model in [27,29]. The in-plane shear stress tensor Λ is a function of the surface deformation gradient of the capsule surface relative to the stress-free reference configuration of the membrane. Let γ be the surface in the current configuration and γ_r be the reference configuration of the membrane. If $\boldsymbol{x}_r \in \gamma_r$ is a point that maps to $\boldsymbol{x} \in \gamma$ in the current configuration, let $\xi(\boldsymbol{x}_r) = \boldsymbol{x}$ be the bijective map between the configurations. The deformation gradient $\boldsymbol{F}$ is defined as $\boldsymbol{F} = \frac{\partial \xi}{\partial \boldsymbol{x}_r}$. If $\boldsymbol{n}_r$ is the normal to the reference configuration at $\boldsymbol{x}_r \in \gamma_r$ and $\boldsymbol{n}$ is the normal to the current configuration at $\boldsymbol{x} \in \gamma$, then the relative surface deformation gradient $\boldsymbol{F}_S$ is defined as $\boldsymbol{F}_S = (\boldsymbol{I} - \boldsymbol{n}\boldsymbol{n}^T)\boldsymbol{F}(\boldsymbol{I} - \boldsymbol{n}_r\boldsymbol{n}_r^T)$, where $\boldsymbol{I}$ is the identity tensor. It follows from the above equation that $\boldsymbol{F}_S$ maps the surface tangents $\boldsymbol{a}_{1r}$ and $\boldsymbol{a}_{2r}$ in the reference configuration γ_r to the tangents $\boldsymbol{a}_1$ and $\boldsymbol{a}_2$ in the current configuration γ. Also, it maps the reference normal $\boldsymbol{n}_r$ to 0. Thus, the following set of equations at every point $\boldsymbol{x}_r$ uniquely determine $\boldsymbol{F}_S$ at that point on the reference configuration: $\boldsymbol{F}_S\boldsymbol{a}_{1r} = \boldsymbol{a}_1$, $\boldsymbol{F}_S\boldsymbol{a}_{2r} = \boldsymbol{a}_2$, $\boldsymbol{F}_S\boldsymbol{n}_r = 0$. The left Cauchy-Green deformation tensor $\boldsymbol{V}^2$ and the surface projection tensor $\boldsymbol{P}$ at point $\boldsymbol{x}$ in the current configuration are then defined as

$$\boldsymbol{V}^2(\boldsymbol{x}) = \boldsymbol{F}_s(\xi^{-1}(\boldsymbol{x}))(\boldsymbol{F}_s(\xi^{-1}(\boldsymbol{x})))^T, \ \boldsymbol{P}(\boldsymbol{x}) = \boldsymbol{I} - \boldsymbol{n}\boldsymbol{n}^T. \tag{2.7}$$

Following the work in [29,32], if λ_1^2, λ_2^2 are the two non-zero eigenvalues of $\boldsymbol{V}^2$, we define two scalar invariants $I_1 = \lambda_1^2 + \lambda_2^2 - 2$ *and* $I_2 = \lambda_1^2\lambda_2^2 - 1$. We now define the shear stress tensor Λ at every point $\boldsymbol{x}$ on the current configuration as

$$\Lambda(\boldsymbol{x}) = \frac{E_s}{2J_s}(I_1 + 1)\boldsymbol{V}^2(\boldsymbol{x}) + \frac{J_s}{2}(E_D I_2 - E_s)\boldsymbol{P}(\boldsymbol{x}), \tag{2.8}$$

where $J_s = \lambda_1\lambda_2$, E_s is the elastic shear modulus of the membrane and E_D is the dilatation modulus. High values of the shear modulus E_s represent higher membrane shear resistance. The dilatation modulus E_D controls the local membrane extensibility.

2.2. Boundary integral formulation

Following the work in [27,29], Equations (2.1) to (2.5) can be formulated into integral equations on the capsule membrane γ. This formulation can be written as:

$$\boldsymbol{f}(\boldsymbol{x}) = \nabla_\gamma \cdot \Lambda(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \gamma, \tag{2.9}$$

$$\boldsymbol{u}(\boldsymbol{x}) = \boldsymbol{u}_\infty(\boldsymbol{x}) + \mathcal{S}_\gamma[\boldsymbol{f}](\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \gamma, \tag{2.10}$$

$$\frac{\partial \boldsymbol{x}}{\partial t} = \boldsymbol{u}(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \gamma. \tag{2.11}$$

Here $\Lambda(\boldsymbol{x})$ is computed using Equation (2.8). $\mathcal{S}_\gamma[\boldsymbol{f}]$ represents the single layer potential of layer density $\boldsymbol{f}$ over the capsule membrane γ defined as follows:

$$\mathcal{S}_\gamma[\boldsymbol{f}](\boldsymbol{x}) = \int_\gamma \mathcal{G}(\boldsymbol{x}, \boldsymbol{y})\boldsymbol{f}(\boldsymbol{y})d\gamma, \tag{2.12}$$

where $\mathcal{G}$ is the Stokes kernel given by $\mathcal{G}(\boldsymbol{x},\boldsymbol{y}) = \frac{1}{8\pi\mu}\left(\frac{I}{||\boldsymbol{r}||} + \frac{\boldsymbol{r}\otimes\boldsymbol{r}}{||\boldsymbol{r}||^3}\right)$ with $\boldsymbol{r} = \boldsymbol{x} - \boldsymbol{y}$. Given the initial position of the capsule and its stress-free reference configuration, we will use Equations (2.9) to (2.11) to simulate the time evolution of capsule under the imposed background flow $\boldsymbol{u}_\infty$.

3. Numerical algorithms

In this section, we describe the numerical algorithms we use to simulate the dynamics of the capsule. We discretize Equations (2.9) to (2.11) discussed in previous section. In Section 3.1, we discuss our parameterization of capsule surface (assuming it is diffeomorphic to the unit sphere) using multiple patches. In Section 3.2, we discuss the discretization of the surface based on our parameterization of the surface. In Section 3.3, we discuss the numerical scheme for the surface integration of a smooth function. In Section 3.4, we discuss the numerical scheme for the singular integration to compute Stokes single layer potential. In Section 3.5, we discuss the numerical scheme for calculating the surface derivatives for the computation of the elastic force. Finally in Section 3.6, we discuss the time stepping we use to simulate the capsule dynamics and provide a summary of the overall algorithm.

3.1. Surface parameterization

We assume the capsule membrane γ to be smooth and diffeomorphic to the unit sphere $\mathbb{S}^2$ embedded in $\mathbb{R}^3$. Thus, we have a smooth bijective map $\phi : \mathbb{S}^2 \longrightarrow \gamma$ with the smooth inverse ϕ^{-1}. We define n_p number of overlapping patches $\{\mathcal{P}_i^0\}_{i=1}^{i=n_p}$, where $\mathcal{P}_i^0 \subset \mathbb{S}^2$ form an open cover of the unit sphere $\mathbb{S}^2$, *i.e.,* $\bigcup_{i=1}^{n_p} \mathcal{P}_i^0 = \mathbb{S}^2$. Additionally, we assume each patch $\mathcal{P}_i^0$ is diffeomorphic to an open set $\mathcal{U}_i$ ($\subset \mathbb{R}^2$) via a coordinate chart η_i^0, *i.e.,* $\eta_i^0 : \mathcal{U}_i \longrightarrow \mathcal{P}_i^0$ is a diffeomorphism. The domain $\mathcal{U}_i$ is called the coordinate domain for the patch $\mathcal{P}_i^0$. Thus, the set $\mathcal{A}^0 = \{(\mathcal{U}_i, \eta_i^0)\}_{i=1}^{n_p}$ forms an atlas for the manifold $\mathbb{S}^2$ [9]. We also know such an atlas admits a smooth partition of unity subordinate to the open cover $\{\mathcal{P}_i^0\}_{i=1}^{n_p}$ [39]. We choose a smooth partition of unity $\{\psi_i^0\}_{i=1}^{n_p}$ where $\psi_i^0 : \mathbb{S}^2 \longrightarrow \mathbb{R}$ such that $\text{supp}(\psi_i^0) \subset \mathcal{P}_i^0$ for $i = 1, \ldots, n_p$, where $\text{supp}(\cdot)$ denotes the support of the function. Thus, for every $\boldsymbol{x}_0 \in \mathbb{S}^2$, $\sum_{i=1}^{n_p} \psi_i^0(\boldsymbol{x}_0) = 1$. The precise definition of ψ_i^0 is given in Equation (3.9).

Define $\mathcal{P}_i := \phi(\mathcal{P}_i^0)$. Now $\{\mathcal{P}_i\}_{i=1}^{n_p}$ form a set of overlapping patches that cover the surface γ, *i.e.,* $\bigcup_{i=1}^{n_p} \mathcal{P}_i = \gamma$. We then define coordinate maps $\eta_i : \mathcal{U}_i \longrightarrow \mathcal{P}_i$, where $\eta_i \equiv \phi \circ \eta_i^0$ and therefore, a corresponding atlas $\mathcal{A} := \{(\mathcal{U}_i, \eta_i)\}_{i=1}^{n_p}$ for the capsule surface γ. A representation of the parameterization is shown in Fig. 2. Since the patches are overlapping, a point $\boldsymbol{x} \in \gamma$ can belong to multiple $\mathcal{P}_i$. For $\boldsymbol{x} \in \gamma$, let us define the set $\mathcal{I}_{\boldsymbol{x}} = \{1 \le i \le n_p |\ \boldsymbol{x} \in \mathcal{P}_i\}$. Thus, $\mathcal{I}_{\boldsymbol{x}}$ is the collection of indices i for which patch $\mathcal{P}_i$ contains $\boldsymbol{x}$. We also define $\mathcal{P}_{ij} = \mathcal{P}_i \cap \mathcal{P}_j$ and $\mathcal{U}_{ij} = \eta_i^{-1}(\mathcal{P}_{ij})$ for $i, j = 1, \ldots, n_p$. We also define transition maps $\tau_{ij} : \mathcal{U}_{ij} \longrightarrow \mathcal{U}_j$ as $\tau_{ij} \equiv (\eta_j^0)^{-1} \circ \eta_i^0|_{\mathcal{U}_{ij}}$. Furthermore, the diffeomorphism ϕ allows us to create a partition of unity $\{\psi_i\}_{i=1}^{n_p}$ on γ by defining it as

$$\psi_i(\boldsymbol{x}) = \psi_i^0(\phi^{-1}(\boldsymbol{x})), \quad \forall \boldsymbol{x} \in \gamma. \tag{3.1}$$

For any function $f : \gamma \longrightarrow \mathbb{R}^d$, we can write

$$f(\boldsymbol{x}) = \sum_{i=1}^{n_p} f(\boldsymbol{x})\psi_i(\boldsymbol{x}), \quad \forall \boldsymbol{x} \in \gamma. \tag{3.2}$$

If f is smooth, then $f\psi_i$ is smooth and compactly supported in $\mathcal{P}_i$ for $i = 1, \ldots, n_p$. We use this partition of unity representation to compute derivatives and integrals on γ.

In this paper, we do not consider the adaptive case, and we assume that the patch parameterization remains unchanged through the calculation. In our numerical experiments, we used just six patches. Precisely, consider $\mathcal{U}_i = (0,\pi) \times (0,\pi)$, $i = 1, \ldots, 6$. The coordinate maps $\{\eta_i^0\}_{i=1}^6$ are given below:

$$\eta_1^0(u,v) = (\sin u \cos v, \sin u \sin v, \cos u), \tag{3.3}$$

$$\eta_2^0(u,v) = (-\sin u \cos v, -\sin u \sin v, \cos u), \tag{3.4}$$

$$\eta_3^0(u,v) = (\sin u \sin v, -\sin u \cos v, \cos u), \tag{3.5}$$

$$\eta_4^0(u,v) = (-\sin u \sin v, \sin u \cos v, \cos u), \tag{3.6}$$

$$\eta_5^0(u,v) = (\sin u \cos v, -\cos u, \sin u \sin v), \tag{3.7}$$

$$\eta_6^0(u,v) = (\sin u \cos v, \cos u, -\sin u \sin v). \tag{3.8}$$

These six charts form six hemispherical patches $\{\mathcal{P}_i^0\}_{i=0}^6$ that cover the unit sphere as shown in Fig. 3. The computation of the transition maps τ_{ij} is relatively simple and is given in the Appendix A.1. We use the bump function, $b(r) = e^{\frac{2e^{-1/|r|}}{|r|-1}}$ for $|r| < 1$ and 0 otherwise, to construct the partition of unity functions on each patch. For the particular parameterization we use, for each patch of the unit sphere, we define

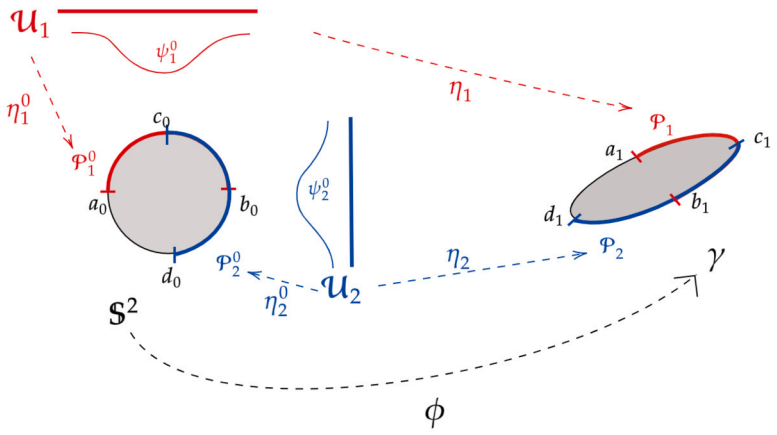


Fig. 2. Here we summarize the notation for the atlas construction. The unit sphere $\mathbb{S}^2$ is on the left and the capsule γ is on the right with the diffeomorphism $\phi : \mathbb{S}^2 \longrightarrow \gamma$. We show two overlapping patches colored in red and blue. The first patch $\mathcal{P}_1^0$ is the red colored arc from point a_0 to b_0 on $\mathbb{S}^2$. Its corresponding patch $\mathcal{P}_1$ on the capsule surface γ is shown in red as the arc from points a_1 to b_1. The second patch $\mathcal{P}_2^0$ is the blue colored arc from point c_0 to d_0 on $\mathbb{S}^2$. Its corresponding patch $\mathcal{P}_2$ on the capsule surface γ is shown in blue as the arc from points c_1 to d_1. Their corresponding coordinate domains $\mathcal{U}_1$ and $\mathcal{U}_2$ are also shown in the red and blue color. The coordinate charts $\eta_i^0 : \mathcal{U}_1 \longrightarrow \mathcal{P}_i^0,\ i = 1, 2$, are shown as dashed lines in the respective colors. The diffeomorphism ϕ also gives the coordinate charts $\eta_i : \mathcal{U}_1 \longrightarrow \mathcal{P}_i,\ i = 1, 2$, for the patches on the capsule surface γ shown as colored dashed lines from $\mathcal{U}_i$ to $\mathcal{P}_i$. The corresponding partition of unity functions $\psi_i^0, i = 1, 2$, with $\text{supp}(\psi_i^0) \subset \mathcal{P}_i^0$ are drawn over coordinate domains $\mathcal{U}_i$ for visual clarity since $\mathcal{P}_i^0$ is diffeomorphic to $\mathcal{U}_i$. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

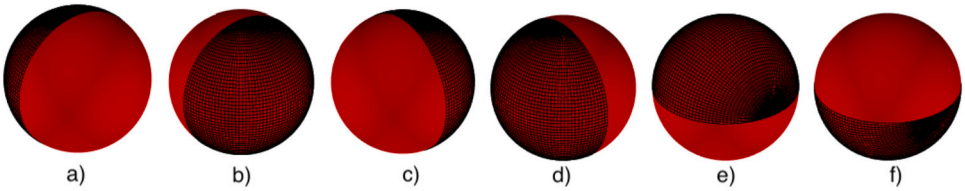


Fig. 3. The six hemispherical patches forming an open cover of the unit sphere S^2. Each one is represented by the black grid. a)–f) $\mathcal{P}_i^0$ for $i = 1, 2, \ldots, 6$.

$$\psi_i^0(\boldsymbol{x}_0) = \frac{b\left(\frac{d(\boldsymbol{x}_0, \eta_i^0(\pi/2,\pi/2))}{r_0}\right)}{\sum_{j=1}^{n_p} b\left(\frac{d(\boldsymbol{x}_0, \eta_j^0(\pi/2,\pi/2))}{r_0}\right)},\ \forall \boldsymbol{x}_0 \in \mathbb{S}^2 \tag{3.9}$$

where $d(\boldsymbol{x}_0, \boldsymbol{y}_0)$ denotes the great circle distance between $\boldsymbol{x}_0$ and $\boldsymbol{y}_0$ on the unit sphere, and $r_0 > 0$ determines the support of the partition of unity inside the patch. The argument of $b(\cdot)$ is the normalized great circle distance from a point $\boldsymbol{x}$ to the center of the patch $\eta_i^0(\pi/2, \pi/2)$, where the normalizing factor r_0 is chosen to be $r_0 = 5\pi/12$ (see Appendix A.3). Now, for a given surface γ and diffeomorphism ϕ, we readily have an atlas and the transition maps for the parameterization of γ. The corresponding partition of unity on γ is available using Equation (3.1) and Equation (3.9).

3.2. Surface discretization

We now describe in detail the discretization of the surface γ using the parameterization described above in Section 3.1. As mentioned above, $\mathcal{U}_i = (0, \pi) \times (0, \pi)$ for our parameterization. We use the following m_{th}-order uniform grid in $\mathcal{U}_i$ as follows:

$$U_{j,k}^{m,i} = \left(\frac{j\pi}{m}, \frac{k\pi}{m}\right),\ \forall j, k \in \{1, \ldots, m-1\}. \tag{3.10}$$

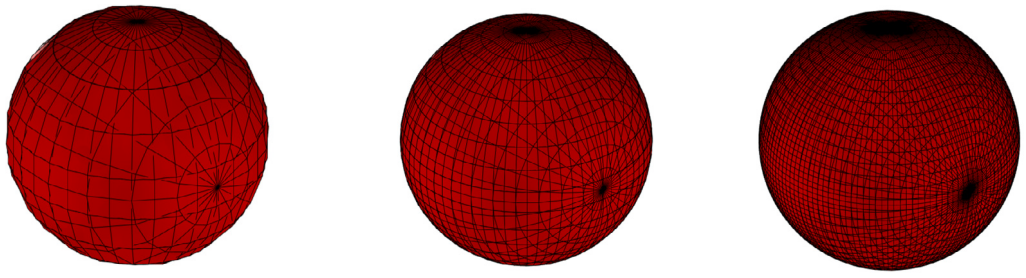
We use h_m to denote the m_{th}-order grid spacing in each $\mathcal{U}_i$ space, *i.e.,* $h_m = \frac{\pi}{m}$. The discretization points for each patch $\mathcal{P}_i^0$ on the unit sphere are given by

$$X_{j,k}^{0,m,i} = \eta_i^0(U_{j,k}^{m,i}),\ \forall j, k \in \{1, \ldots, m-1\}. \tag{3.11}$$

The discretization points for each patch $\mathcal{P}_i$ on γ are given by

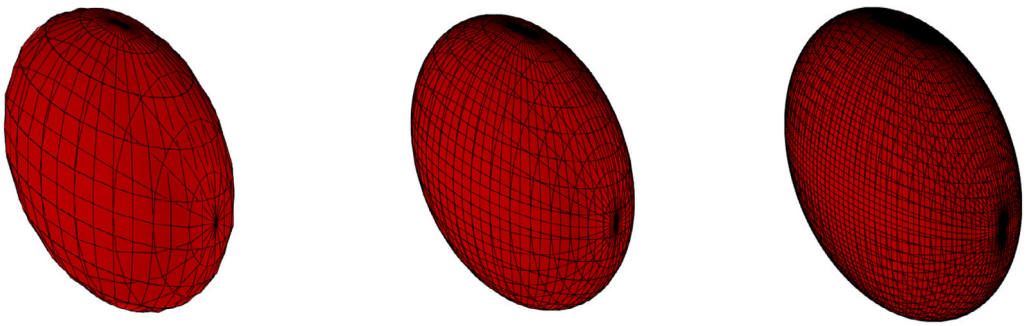
$$X_{j,k}^{m,i} = \eta_i(U_{j,k}^{m,i}),\ \forall j, k \in \{1, \ldots, m-1\}. \tag{3.12}$$

Thus, each patch contains $(m-1)^2$ discretization points for an m_{th}-order grid leading to a total of $N = 6(m-1)^2$ points for the surface. The dynamics of the capsule are represented as the time trajectories $X_{j,k}^{m,i}(t)$. The partition of unity values at the discretization points



(a) $m = 8$ (b) $m = 16$ (c) $m = 32$

Fig. 4. Representation of discretization of a unit sphere using m_{th}-order grids for (a) $m = 8$, (b) $m = 16$, (c) $m = 32$.



(a) $m = 8$ (b) $m = 16$ (c) $m = 32$

Fig. 5. Representation of discretization of the ellipsoid $x^2/a^2 + y^2/b^2 + z^2/c^2 = 1$ with $a = 0.5, b = 1, c = 1$, using m_{th}-order grids for (a) $m = 8$, (b) $m = 16$, (c) $m = 32$.

$\psi_{j,k}^{m,i} = \psi_i(X_{j,k}^{m,i})$ can be precomputed and stored to be used later for computing integrals and derivatives (see Equation (3.1)). The sample grids for a unit sphere and an ellipsoid are given in Fig. 4 and Fig. 5. To get the diffeomorphism ϕ for an ellipsoid γ given by the level surface $\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1$, we consider the spherical angles parameterization $\beta(u, v) : [0, \pi] \times [0, 2\pi) \longrightarrow \gamma$ given by

$$\beta(u, v) = (a \sin u \cos v, b \sin u \sin v, c \cos u), u \in [0, \pi], v \in [0, 2\pi). \tag{3.13}$$

We also consider the spherical angles parameterization of the unit sphere $\beta_0(u, v) : [0, \pi] \times [0, 2\pi) \longrightarrow \mathbb{S}^2$ given by

$$\beta_0(u, v) = (\sin u \cos v, \sin u \sin v, \cos u), u \in [0, \pi], v \in [0, 2\pi). \tag{3.14}$$

The diffeomorphism $\phi : \mathbb{S}^2 \longrightarrow \gamma$ is then given as

$$\phi(\boldsymbol{x}_0) = \begin{cases} \beta(\beta_0^{-1}(\boldsymbol{x}_0)) & \text{if } \boldsymbol{x}_0 \in \mathbb{S}^2 \backslash \{(0,0,-1),(0,0,1)\}, \\ (0,0,-c) & \text{if } \boldsymbol{x}_0 = (0,0,-1), \\ (0,0,c) & \text{if } \boldsymbol{x}_0 = (0,0,1). \end{cases}$$

Remark 1. Given an initial capsule surface γ, we only use the diffeomorphism $\phi : \mathbb{S}^2 \longrightarrow \gamma$ to compute the atlas $\mathcal{A}$, the partition of unity $\{\psi_i\}_{i=1}^{n_p}$ for γ and the transition maps τ_{ij} for the coordinate domains. Once computed, we initialize the surface discretization points (see Equation (3.12)), and track their time trajectories to simulate the dynamics of the capsule. We do not need to use ϕ afterwards for capsule dynamics simulation. We do not change the partition of unity values after initialization. Hence, we have $\psi_{j,k}^{m,i} = \psi_i(X_{j,k}^{m,i}(t_0)) = \psi_i(X_{j,k}^{m,i}(t))$ where t_0 is the initial time. These partition of unity values are computed at initialization and stored for subsequent use. Also, the transition maps between the coordinate domains remain unchanged and are precomputed for usage later.

Notation: For brevity, we use $\boldsymbol{U}^m$ to denote the $N \times 2$ matrix containing all the m_{th}-order grid points, *i.e.*, $\boldsymbol{U}^m = [\{U_{j,k}^{m,i}\}_{1 \le j,k \le m, 1 \le i \le n_p}]^T$. Similarly, we define $\boldsymbol{X}^m = [\{X_{j,k}^{m,i}\}_{1 \le j,k \le m, 1 \le i \le n_p}]^T$ to be the $N \times 3$ matrix of corresponding discretization points on γ, $\boldsymbol{\Psi}^m = [\{\psi_{j,k}^{m,i}\}_{1 \le j,k \le m, 1 \le i \le n_p}]^T$ to be the $N \times 1$ column vector containing the values of partition of unity functions and $\boldsymbol{F}^m$ to be the $N \times 3$ matrix containing the interfacial force $\boldsymbol{f}$ values at discretization points of the m_{th}-order grid. Further, we use $\boldsymbol{U}^{m,i}$ to denote the $(m-1)^2 \times 2$ matrix of grid points belonging to $\mathcal{U}_i$. Similarly, we use $\boldsymbol{X}^{m,i}, \boldsymbol{\Psi}^{m,i}, \boldsymbol{F}^{m,i}$ to denote the values in the patch $\mathcal{P}_i$.

For the unit sphere, we denote the corresponding discretization points as $\boldsymbol{X}^{0,m}$. The partition of unity column vector on $\boldsymbol{X}^{0,m}$ is the same as $\boldsymbol{\Psi}^m$.

Remark 2. While we do not need ϕ for simulating capsule dynamics after the initialization of atlas, transition maps and partition of unity, we use $||\nabla_{\mathbb{S}^2}\phi||_\infty$ as a quality metric to gauge the smoothness of the capsule surface γ during the simulations in this paper (see Section 3.6). To compute $||\nabla_{\mathbb{S}^2}\phi||_\infty$, we will need the discretization points $\boldsymbol{X}^{0,m}$. Hence, we store these values as well. Note that $\phi(\boldsymbol{X}^{0,m}) = \boldsymbol{X}^m$.

3.3. Smooth surface integrals

Let $f : \gamma \longrightarrow \mathbb{R}^d$ be a smooth function. We can write its surface integral as a sum of the integral over all the patches using the partition of unity $\{\psi_i\}_{i=1}^{n_p}$. Then, we use the product periodic trapezoidal rule using the function values at the grid points mentioned in Equation (3.12), which gives superalgebraic convergence [6]. We write

$$\int_\gamma f\,d\gamma = \sum_{i=1}^{n_p}\int_{\mathcal{P}_i} f\psi_i d\gamma = \sum_{i=1}^{n_p}\int_{\mathcal{U}_i} f\psi_i W_i d\mathcal{U}_i \approx \sum_{i=1}^{n_p}\left(\sum_{j,k\in\{1,\ldots,m-1\}} f(X^{m,i}_{j,k})\psi^{m,i}_{j,k}W_i(X^{m,i}_{j,k})h_m^2\right), \tag{3.15}$$

where W_i is the surface area element for patch $\mathcal{P}_i$ (numerical computation of W_i is discussed in Section 3.5).

Work complexity: The work complexity to compute this integral numerically for m_{th}-order grid is $O(n_p m^2)$. In our specific parameterization, $n_p = 6$. Hence, the work required is $O(6m^2)$.

3.4. Singular integration

We use the regularized Stokes kernel described in [34] to evaluate Stokes potentials to high-order accuracy using the discretization described above in Section 3.2.

Consider a smooth vector-valued function $\boldsymbol{f} : \gamma \longrightarrow \mathbb{R}^3$. The single-layer Stokes potential of $\boldsymbol{f}$ at $\boldsymbol{x} \in \gamma$ as described in Section 2.2 can be written as

$$S_\gamma[\boldsymbol{f}](\boldsymbol{x}) = \frac{1}{8\pi\mu}\int_\gamma \left(\frac{\boldsymbol{f}(\boldsymbol{y})}{r} + (\boldsymbol{f}(\boldsymbol{y})\cdot(\boldsymbol{x}-\boldsymbol{y}))(\boldsymbol{x}-\boldsymbol{y})\frac{1}{r^3}\right)d\gamma, \tag{3.16}$$

where $r = ||\boldsymbol{x}-\boldsymbol{y}||$. The regularized version of Stokes single layer potential [34] is given as

$$S^\delta_\gamma[\boldsymbol{f}](\boldsymbol{x}) = \frac{1}{8\pi\mu}\int_\gamma \left(\frac{\boldsymbol{f}(\boldsymbol{y})s_1(r/\delta)}{r} + (\boldsymbol{f}(\boldsymbol{y})\cdot(\boldsymbol{x}-\boldsymbol{y}))(\boldsymbol{x}-\boldsymbol{y})\frac{s_2(r/\delta)}{r^3}\right)d\gamma, \tag{3.17}$$

where $\delta > 0$ is the regularization parameter and s_1, s_2 are smoothing factors such that $\frac{s_1(r/\delta)}{r}$ and $\frac{s_2(r/\delta)}{r^3}$ are smooth functions as $r \longrightarrow 0$. These smoothing factors ensure that $S^\delta_\gamma[\boldsymbol{f}]$ is an integral involving a smooth kernel and can be evaluated using the periodic trapezoidal rule as in Section 3.3. Following [24,34], we choose the smoothing factors s_1 and s_2 to be

$$s_1(r) = \text{Erf}(r) - \frac{2}{3}r(2r^2-5)\frac{e^{-r^2}}{\sqrt{\pi}}, \tag{3.18}$$

$$s_2(r) = \text{Erf}(r) - \frac{2}{3}r(4r^4-14r^2+3)\frac{e^{-r^2}}{\sqrt{\pi}}, \tag{3.19}$$

where $\text{Erf}(\cdot)$ is the error function. Using Taylor expansion, we can show that as $r \longrightarrow 0$, $\frac{s_1(r/\delta)}{r} \longrightarrow \frac{16}{3\delta\sqrt{\pi}}$ and $\frac{s_2(r/\delta)}{r^3} \longrightarrow \frac{32}{3\delta^3\sqrt{\pi}}$. This choice of smoothing factors ensures that the regularized Stokes potential is $O(\delta^5)$ accurate [34]. This regularized integral is discretized by the trapezoidal rule for smooth functions. However, the question is what should be the relation between m and δ?

Regularization parameter δ: The choice of regularization parameter δ is crucial. As discussed in [3,34], the chosen δ should be large enough that regularization error dominates the discretization error in computing the integral in Equation (3.17). In our simulations, when computing the Stokes potential for a target point $X^{m,i}_{j,k}$ in patch $\mathcal{P}_i$, we choose a $\mathcal{P}_i$-dependent regularization parameter δ_i at every time step as the capsule evolves. It is defined by

$$\delta^{m,i}_{j,k} = C\delta^* \text{ where } \delta^* = \left(\max_{l,l'\in\{1,\ldots,m-1\}}\{d_e(X^{m,i}_{l,l'}, \mathcal{X}(X^{m,i}_{l,l'})\}\right). \tag{3.20}$$

Here, we choose the constant $C = 1$ (see Appendix A.7 for more details on the choice C), $\mathcal{X}(X^{m,i}_{l,l'}) = \{X^{m,i}_{l-a,l'-b} : a,b \in \{-1,0,1\}\}$ refers to the set of points which are adjacent neighbors of $X^{m,i}_{l,l'}$ and $d_e(a,B)$ denotes the maximum Euclidean distance between a and the points in set B. For brevity, we define $\boldsymbol{\delta}^m = [\{\delta^{m,i}_{j,k}\}_{1\le j,k\le m-1, 1\le i\le n_p}]^T$. We experimentally observed that our choice for δ improves the accuracy the singular quadrature as in our simulations as capsule changes shapes and patches evolve over time, as opposed

to using a fixed global regularization parameter $\delta = C' h_m$ where C' is a positive constant (like in [34]). We tabulate the relative errors for different values of C' in using fixed global regularization parameter $\delta = C' h_m$ on a sample ellipsoidal shape in Table 11 to illustrate this.

Upsampling: In our experiments, the observed prefactor in our weakly singular quadrature scheme is quite significant. For example at small m, say $m = 16$, we can resolve the surface well but we need more quadrature points to resolve the weakly singular integrals. For this reason we use upsampling, especially in the small-m regime. Using numerical experiments, we determined that a four times upsampling works well. Thus, we use an upsampled grid with $N_{\text{up}} = 6(4m-1)^2$ grid points for evaluating the single layer potential. We use cubic spline interpolation [8] on each patch to upsample the surface discretization points $\boldsymbol{X}^m$ and surface interfacial force $\boldsymbol{F}^m$. We define this upsampling operator as $\boldsymbol{I}_u^m$. Thus, the upsampled surface points and surface force vector can be written as

$$\boldsymbol{X}_u^m = \boldsymbol{I}_u^m \boldsymbol{X}^m, \boldsymbol{F}_u^m = \boldsymbol{I}_u^m \boldsymbol{F}^m. \tag{3.21}$$

The partition of unity values on the upsampled grid, denoted by $\boldsymbol{\Psi}_u^m$, can also be precomputed and stored for use. We use the upsampled values to compute the Stokes single layer potential (Equation (3.17)) and then downsample the Stokes potential to get the values on m_{th}-order grid. Note that both $\boldsymbol{I}_u^m$ and $\boldsymbol{I}_d^m$ are cubic spline based weighted interpolation operators. Their matrices can be precomputed and stored for use throughout the simulation. We also compute the $\boldsymbol{\delta}_u^m$ for the upsampled grid using Equation (3.20).

Work complexity: The work complexity for the evaluation of Stokes single layer potential on upsampled grid for a target point is $O(96m^2)$. For N_{up} target points in upsampled grid, the total work complexity for computing single layer potential is $O(9216m^4)$. The complexity for upsampling and downsampling is $O(m^2)$ since $\boldsymbol{I}_u^m$ and $\boldsymbol{I}_d^m$ are sparse linear operators. Hence, the total work complexity is $O(m^4)$.

GPU acceleration: The evaluation of single layer potential on upsampled grid is the most expensive part of our numerical scheme. In order to accelerate this computation, we use a CUDA implementation of computation of Stokes single layer potential computation based on the all-pairs $O(N^2)$ calculation using CUDA on GPU (for details refer to [25]).

3.5. Surface derivatives

In this section we discuss in detail our numerical scheme for calculating surface derivatives. The derivatives are needed for computing the interfacial force $\boldsymbol{f}$ and the surface area element W (for Equation (3.15)) at each discretization point. The first step in the computation of the interfacial force $\boldsymbol{f}$ is the computation of the shear stress tensor Λ. The computation of Λ requires the surface tangents and normals in the current configuration (see Equation (2.8)). After computing the stress tensor Λ, the second step is to compute the interfacial force by calculating the surface divergence of Λ as $\boldsymbol{f} = \nabla_\gamma \cdot \Lambda$. For the computation of surface area element W, we need the coefficients E, F, G of the first fundamental form of the surface. The formulas for the surface divergence, denoted by $\nabla_\gamma \cdot$, and for the surface area element W along with E, F, G are summarized in the Appendix A.2.

Let $\eta(u,v) : \mathcal{U} \longrightarrow \mathcal{P} \subset \gamma$ be a surface patch of γ where $\eta \in \{\eta_1, \ldots, \eta_{n_p}\}$. Let $\eta(u_0, v_0) = \boldsymbol{x} \in \mathcal{P}$. Below, we summarize the steps required to evaluate the required shape derivatives $\boldsymbol{f}(\eta(u_0, v_0))$ and $W(\eta(u_0, v_0))$.

1. We need to compute the tangents, denoted by $\boldsymbol{x}_u$ and $\boldsymbol{x}_v$, and the unit normal $\boldsymbol{n}(\boldsymbol{x})$ for the current and the reference configuration. These quantities are given by $\boldsymbol{x}_u = \frac{\partial \eta}{\partial u}|_{(u_0,v_0)}, \boldsymbol{x}_v = \frac{\partial \eta}{\partial v}|_{(u_0,v_0)}$ and $\boldsymbol{n}(\boldsymbol{x}) = \frac{\boldsymbol{x}_u \times \boldsymbol{x}_v}{||\boldsymbol{x}_u \times \boldsymbol{x}_v||}$. Given these derivatives we can then compute $\Lambda(\eta(u_0, v_0))$.
2. The next step is to compute the surface divergence of $\Lambda(\eta(u,v))$ at (u_0, v_0). This involves computation of E, F, G and partial u, v derivatives of the components of $\Lambda(\eta(u,v))$ (see Appendix A.2). The computation of E, F, G at point $\boldsymbol{x}$ is straightforward using $\boldsymbol{x}_u$ and $\boldsymbol{x}_v$. The surface area element is then computed using $W(\boldsymbol{x}(u_0, v_0)) = \sqrt{EG - F^2}$.

We note here from the above summary that the computation of both $\boldsymbol{f}$ and W boils down to multiple computations of the u and v partial derivatives of functions like the coordinate chart $\eta(u,v)$ and the components of $\Lambda(\eta(u,v))$. Now given an arbitrary function $g : \gamma \longrightarrow \mathbb{R}$, we consider the parametric function $\tilde{g}^i \equiv g(\eta_i(u,v)) : \mathcal{U}_i \longrightarrow \mathbb{R}$ for $i = 1, \ldots, n_p$. We denote its partial u, v derivatives as $\tilde{g}_u^i$ and $\tilde{g}_v^i$. We describe below the numerical scheme to compute these partial derivatives $\tilde{g}_u^i$ and $\tilde{g}_v^i$. This scheme is then sufficient to compute $\boldsymbol{f}$ and W at the discretization grid $\boldsymbol{U}^m$ using the formulas mentioned in Appendix A.2. Before we describe the scheme in detail, we summarize the three main steps of the scheme below:

1. *Patch extension:* We intend to use central finite difference stencil to compute $\tilde{g}_u^i$ and $\tilde{g}_v^i$ at the discretization points for $i = 1, \ldots, n_p$. To apply central difference at the grid points near the boundary of each coordinate domain $\mathcal{U}_i$, we need function values on ghost grid points outside the discretization grid in $\mathcal{U}_i$. We call this process of obtaining values on the extended grid points as *patch extension*. We obtain the values of $\tilde{g}^i$ on these ghost grid points using the patch extension process. Let $\boldsymbol{x}_1 = \eta_i(u_1^i, v_1^i)$ for $i = 1, \ldots, n_p$, be an extended discretization point (see Fig. 6). The value of the function $\tilde{g}^i$ at the extended grid point (u_1^i, v_1^i), is given in the continuous form as the weighted average of the values across the patches as follows:

$$\tilde{g}^i(u_1^i, v_1^i) = \sum_{1 \le j \le n_p} \tilde{g}^j(u_1^j, v_1^j) \psi_j(\boldsymbol{x}_1). \tag{3.22}$$

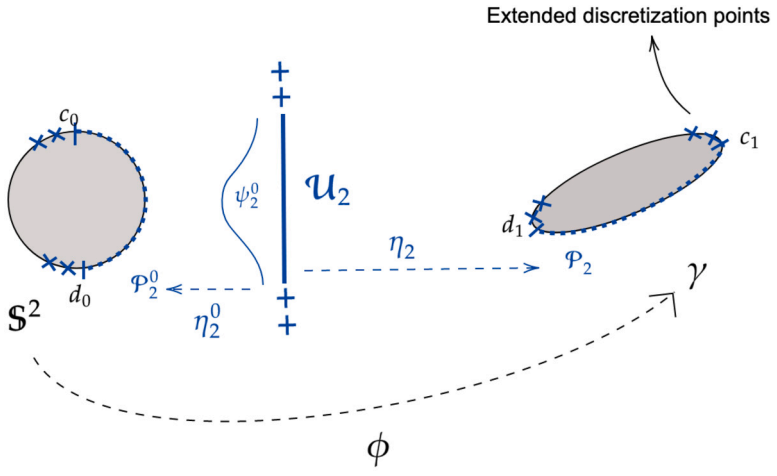


Fig. 6. A 2D representation of a patch extension to apply the central finite difference stencil. The unit sphere $\mathbb{S}^2$ is on the left and the capsule γ is on the right with the diffeomorphism $\phi : \mathbb{S}^2 \longrightarrow \gamma$. The discrete points on the patch $\mathcal{P}_2^0$ are represented by the blue colored dotted arc from point c_0 to d_0 on $\mathbb{S}^2$. Its corresponding patch $\mathcal{P}_2$ on the capsule surface γ is shown in dotted blue line as the arc from points c_1 to d_1. Its corresponding coordinate domain $\mathcal{U}_2$ is also shown in the blue color. The extended grid points are shown in '+' symbol alongside $\mathcal{U}_2$. The extended discretization points are on $\mathbb{S}^2$ and γ are shown with the '×' symbol. The coordinate map η_2^0 we use in our parameterization has a natural extension to the extended domain with extended grid points with the same expression as in Equation (3.8).

Here, the partition of unity values are used as the weights for each patch. In the continuous case, $\tilde{g}^i(u_1^i, v_1^i) = \tilde{g}^j(u_1^j, v_1^j)$ for $i, j = 1, \ldots, n_p$. Hence, it is easy to see that the Equation (3.22) is valid in the continuous setting by putting $\tilde{g}^i(u_1^i, v_1^i) = \tilde{g}^j(u_1^j, v_1^j)$ for $i, j = 1, \ldots, n_p$ $i, j = 1, \ldots, n_p$ and noting that the partition of unity values add up to unity. In the discrete case, the above equation provides a unique definition of discretized function values at every extended grid point since $\tilde{g}^i(u_1^i, v_1^i)$ and $\tilde{g}^j(u_1^j, v_1^j)$ are not necessarily the same for $i, j \in \{1, \ldots, n_p\}$ in the discrete case due to numerical errors.

2. *Finite difference:* After extension of function values to ghost grid points, we use the standard fourth order central finite difference operator on extended grids to obtain $\tilde{g}_u^i$ and $\tilde{g}_v^i$ on the grid points $\boldsymbol{U}^{m,i}$ for $i = 1, \ldots, n_p$.
3. *Blending:* We note that a point $\boldsymbol{x} \in \gamma$ can belong to several different patches. We suppose $\boldsymbol{x} = \eta_i(u_0^i, v_0^i)$ for $i = 1, \ldots, n_p$. The interfacial force $\boldsymbol{f}(\boldsymbol{x})$ is intrinsic to the surface γ and is independent of the local surface parameterization. Using patch parameterization dependent numerical derivatives $\tilde{g}_u^i$ and $\tilde{g}_v^i$ leads to different discrete derivative values at (u_0^i, v_0^i) for different $i = 1, \ldots, n_p$. To get unique derivative values consistent across all the patches, we *blend* the derivatives across all the patches. The blending process is taking the weighted average of derivative values across all the patches except that we also have to transform the derivatives from, say the domain $\mathcal{U}_j$ to the domain $\mathcal{U}_i$. This transformation is given by the Jacobian $\boldsymbol{J}_{\tau_{ij}}$ of the transition map $\tau_{ij}(u, v) = (\tau_{ij}^{(1)}(u, v), \tau_{ij}^{(2)}(u, v)) \in \mathcal{U}_j$ as follows:

$$\begin{bmatrix} \tilde{g}_u^i \\ \tilde{g}_v^i \end{bmatrix} = \boldsymbol{J}_{\tau_{ij}} \begin{bmatrix} \tilde{g}_u^j \\ \tilde{g}_v^j \end{bmatrix}, \tag{3.23}$$

where the Jacobian matrix is

$$\boldsymbol{J}_{\tau_{ij}} = \begin{bmatrix} \partial\tau_{ij}^{(1)}/\partial u & \partial\tau_{ij}^{(2)}/\partial u \\ \partial\tau_{ij}^{(1)}/\partial v & \partial\tau_{ij}^{(2)}/\partial v \end{bmatrix}. \tag{3.24}$$

Here, $\tau_{ij}^{(1)}, \tau_{ij}^{(2)}$ are defined as the first and second component respectively of the transition map, *i.e.,* $\tau_{ij}(u, v) = (\tau_{ij}^{(1)}(u, v), \tau_{ij}^{(2)}(u, v)) \in \mathcal{U}_j$. Both the RHS and the LHS in Equation (3.23) are evaluated at (u_0^i, v_0^i). Using this Jacobian transformation, we define the blending process as the weighted average using partition of unity values as weights for the patches:

$$\begin{bmatrix} \tilde{g}_u^i \\ \tilde{g}_v^i \end{bmatrix} = \sum_{j=1}^{n_p} \psi_j(\boldsymbol{x}) \boldsymbol{J}_{\tau_{ij}} \begin{bmatrix} \tilde{g}_u^j \\ \tilde{g}_v^j \end{bmatrix} \tag{3.25}$$

for all $i = 1, \ldots, n_p$, where $\boldsymbol{x} = \eta_i(u_0^i, v_0^i)$. The partition of unity values ψ_j form the respective weights for the coordinate domain $\mathcal{U}_i$ like in the Equation (3.22). Notice that the Equation (3.25) is trivially valid in the continuous case since Equation (3.23) holds and ψ_j sum to unity. The above equation can be expanded as follows:

$$\tilde{g}_u^i(u_0^i, v_0^i) = \tilde{g}_u^i(u_0^i, v_0^i)\psi_i(\boldsymbol{x}) + \sum_{j \neq i, 1 \leq j \leq n_p} \left(\tilde{g}_u^j \left. \frac{\partial \tau_{ij}^{(1)}}{\partial u} \right|_{(u_0^i, v_0^i)} + \tilde{g}_v^j \left. \frac{\partial \tau_{ij}^{(2)}}{\partial u} \right|_{(u_0^i, v_0^i)} \right) \psi_j(\boldsymbol{x}) \tag{3.26}$$

$$\tilde{g}_v^i(u_0^i, v_0^i) = \tilde{g}_v^i(u_0^i, v_0^i)\psi_i(\boldsymbol{x}) + \sum_{j\neq i, 1\leq j\leq n_p} \left(\tilde{g}_u^j \left.\frac{\partial \tau_{ij}^{(1)}}{\partial v}\right|_{(u_0^i, v_0^i)} + \tilde{g}_v^j \left.\frac{\partial \tau_{ij}^{(2)}}{\partial v}\right|_{(u_0^i, v_0^i)} \right) \psi_j(\boldsymbol{x}). \tag{3.27}$$

The terms inside the parenthesis in Equation (3.26) and Equation (3.27) come from the Jacobian transformation in Equation (3.23). We list all the transition maps for the specific parameterization we use in Appendix A.1. Equation (3.26) and Equation (3.27) describe the blending equation we use to get the weighted average of derivatives across all patches.

We now describe in detail all the three steps mentioned above and write the discretized versions of these steps.

Notation: We use $\tilde{\boldsymbol{g}}^m$ to denote the column vector of the values of $\tilde{g}^i$ at the grid points $\boldsymbol{U}^{m,i}$ for all $i = 1, \ldots, n_p$. We use $\tilde{\boldsymbol{g}}^{m,i}$ to denote the column vector of the values at the grid points $\boldsymbol{U}^{m,i}$.

Patch extension: As discussed above, for the discretization points near the boundary of $\mathcal{U}_i, i = 1, \ldots, n_p$, we need function values on *ghost points* lying outside $\mathcal{U}_i$ in order to use the central FDM stencil. In our simulation, we use a fifth order 7-point symmetric 1D-stencil [15] for each of the partial derivatives. Thus, in order to calculate the derivative values on the grid $\boldsymbol{U}^m$, we need the values of $\tilde{g}$ on the extended grid (including ghost nodes) given by

$$U_{j,k}^{m,i,ext} = \left(\frac{j\pi}{m}, \frac{k\pi}{m} \right), \ \forall j, k \in \{-2, -1, \ldots, m+1, m+2\}, \tag{3.28}$$

where the set of points defined by $G^{m,i} := \{U_{j,k}^{m,i,ext}\}_{j,k\in\{-2,-1,\ldots,m+1,m+2\}} \backslash \{U_{j,k}^{m,i,ext}\}_{j,k\in\{1,\ldots,m-1\}}$ denotes the ghost discretization points for $\mathcal{U}_i$. We define the extended coordinate chart domain as $\mathcal{U}_i^{m,ext} = (\frac{-3\pi}{m}, \frac{(m+3)\pi}{m}) \times (\frac{-3\pi}{m+1}, \frac{(m+4)\pi}{m+1})$ containing the extended grid points as defined in Equation (3.28). The parameterization of unit sphere η_i^0 has a natural extension to $\mathcal{U}_i^{m,ext}$ with the same function expressions as given in Equation (3.8). Using the diffeomorphism ϕ, we also have mappings $\eta_i^{m,ext} : \mathcal{U}_i^{m,ext} \longrightarrow \gamma$. We define $\mathcal{P}_i^{m,ext} := \eta_i^{m,ext}(\mathcal{U}_i^{m,ext})$. We define the matrix of extended grid points for all the coordinate domains as $\boldsymbol{U}^{m,ext} = [\{U_{j,k}^{m,i,ext}\}_{-2\leq j,k\leq m+2, 1\leq i\leq n_p}]^T$, while $\boldsymbol{U}^{m,i,ext}$ refers to the matrix of extended grid points for the coordinate domain $\mathcal{U}_i$. The corresponding discretization points on the surface γ are theoretically given by

$$X_{j,k}^{m,i,ext} = \eta_i \left(U_{j,k}^{m,i,ext} \right), \ \forall j, k \in \{-2, \ldots, m+2\}. \tag{3.29}$$

Let the matrix of all surface discretization nodes on extended patch be $\boldsymbol{X}^{m,ext} = [\{X_{j,k}^{m,i}\}_{-2\leq j,k\leq m+2, 1\leq i\leq n_p}]^T$. Our goal of patch extension is to find the values of the function $\tilde{g}^i$ on $\boldsymbol{U}^{m,i,ext}$, denoted by $\tilde{\boldsymbol{g}}^{m,i,ext}$, using the known values on $\boldsymbol{U}^{m,i}$. The discretized form of patch extension is given as

$$\tilde{\boldsymbol{g}}^{m,i,ext} = \sum_{1\leq j\leq n_p} \left(\boldsymbol{I}_{ij}^{m,ext} \tilde{\boldsymbol{g}}^{m,j} \right) \psi_j \Big|_{\boldsymbol{X}^{m,i,ext}}. \tag{3.30}$$

This equation is the discretized version of the Equation (3.22). Here, we additionally use the cubic splines interpolation, denoted by the operator $\boldsymbol{I}_{ij}^{m,ext}$, to get the values of $\tilde{g}^j$ on $\boldsymbol{U}^{m,i,ext}$ using the known values on the discrete grid $\boldsymbol{U}^{m,j}$.

Finite Differences: We use $\boldsymbol{D}_u^{m,i}$ and $\boldsymbol{D}_v^{m,i}$ to denote the central finite difference operators for computing partial derivative with respect to u and v respectively on the m_{th}-order grid in coordinate chart domain $\mathcal{U}_i$. Thus, we write the partial derivatives on $\boldsymbol{U}^{m,i}$ as

$$\tilde{\boldsymbol{g}}_u^{m,i} = \boldsymbol{D}_u^{m,i} \tilde{\boldsymbol{g}}^{m,i,ext}, \tilde{\boldsymbol{g}}_v^{m,i} = \boldsymbol{D}_v^{m,i} \tilde{\boldsymbol{g}}^{m,i,ext}, \tag{3.31}$$

where $\tilde{\boldsymbol{g}}^{m,i,ext}$ is calculated using Equation (3.30).

Blending: As discussed above, applying Equation (3.31) can lead to different numerical values of derivatives at the same discretization point for different domains $\mathcal{U}_i$. Therefore, we use blending to get unique values across all the patches. The discretized versions of the blending equations Equation (3.26) and Equation (3.27) are given as follows:

$$\tilde{\boldsymbol{g}}_u^{m,i} = \tilde{\boldsymbol{g}}_u^{m,i} \ \psi_i\big|_{\boldsymbol{X}^{m,i}} + \sum_{j\neq i, 1\leq j\leq n_p} \left(\left(\boldsymbol{I}_{ij}^m \tilde{\boldsymbol{g}}_u^{m,j} \right) \left.\frac{\partial \tau_{ij}^{(1)}}{\partial u}\right|_{\boldsymbol{U}^{m,i}} + \left(\boldsymbol{I}_{ij}^m \tilde{\boldsymbol{g}}_v^{m,j} \right) \left.\frac{\partial \tau_{ij}^{(2)}}{\partial u}\right|_{\boldsymbol{U}^{m,i}} \right) \psi_j\Big|_{\boldsymbol{X}^{m,i}}, \tag{3.32}$$

$$\tilde{\boldsymbol{g}}_v^{m,i} = \tilde{\boldsymbol{g}}_v^{m,i} \ \psi_i\big|_{\boldsymbol{X}^{m,i}} + \sum_{j\neq i, 1\leq j\leq n_p} \left(\left(\boldsymbol{I}_{ij}^m \tilde{\boldsymbol{g}}_u^{m,j} \right) \left.\frac{\partial \tau_{ij}^{(1)}}{\partial v}\right|_{\boldsymbol{U}^{m,i}} + \left(\boldsymbol{I}_{ij}^m \tilde{\boldsymbol{g}}_v^{m,j} \right) \left.\frac{\partial \tau_{ij}^{(2)}}{\partial v}\right|_{\boldsymbol{U}^{m,i}} \right) \psi_j\Big|_{\boldsymbol{X}^{m,i}}, \tag{3.33}$$

for $i = 1, \ldots, n_p$, where $\boldsymbol{I}_{ij}^m$ is the cubic splines interpolation operator. It returns approximated function values on $\boldsymbol{U}^{m,i}$ grid from the known values on the grid $\boldsymbol{U}^{m,j}$.

The three steps, namely the patch extension, the finite differences and the blending of derivatives described above constitute our numerical scheme to calculate the interfacial forces using the formulas given in Appendix A.2.

Work complexity: The finite difference operators $\boldsymbol{D}_u^{m,i}$ and $\boldsymbol{D}_v^{m,i}$ and the inter-patch interpolation operators $\boldsymbol{I}_{ij}^m, i, j \in \{1, \ldots, n_p\}$, can all be precomputed and stored for application as sparse matrices. The work complexity of patch extension is of the order of number of additional points in the extended grid, *i.e.,* $O(n_p(m+5)^2 - n_p(m-1)^2) \equiv O(n_p m)$. The work required for applying finite

difference operators is $O(n_p m^2)$. Finally, the work required for blending derivatives across patches is $O(n_p m^2)$. Thus, the total work complexity for calculating derivatives is $O(n_p m^2)$. In our implementation $n_p = 6$, so the work complexity is $O(6m^2)$.

Accuracy: While the finite difference operators we use are fifth order accurate, our numerical scheme is expected to be fourth order accurate because we use cubic splines for patch extension and blending. This is confirmed in our numerical experiments as tabulated in Table 2a and Table 2b and discussed in later sections.

Remark 3. We mention that the main purpose of blending is to get consistent unique derivative values by taking weighted average across different patches since a point $\boldsymbol{x} \in \gamma$ can belong to multiple patches. We note here that the blending also improves the accuracy of the derivatives as evidenced in the results tabulated in Table 7 and discussed in Appendix A.4.

3.6. Time stepping and the overall algorithm

Here, we describe the time stepping scheme we use to solve the boundary integral formulation given by Equations (2.9) to (2.11). The equations can be reformulated into an initial value problem as

$$\frac{\partial \boldsymbol{x}}{\partial t} = \boldsymbol{u}_\infty(\boldsymbol{x}) + S_\gamma[\boldsymbol{f}](\boldsymbol{x}), \tag{3.34}$$

$$= \mathcal{L}(t, \boldsymbol{x}), \ \forall \boldsymbol{x} \in \gamma, \tag{3.35}$$

where the initial position and initial deformation gradient tensor is known. We use an explicit Runge-Kutta-Fehlberg 4(5) [13,14] adaptive time stepping scheme to simulate the time evolution of capsules. This scheme is fourth-order accurate in time. We use fixed relative tolerance $\epsilon_{tol} = 10^{-6}$ for adaptive time stepping in our simulations. Below we give the overall algorithm for the evaluation of the right hand side of Equation (3.35):

1. Given the initial capsule surface, we first initialize the atlas, the transition maps and its derivatives (for Equation (3.26) and Equation (3.27)). We precompute $\boldsymbol{\Psi}^m, \boldsymbol{\Psi}_u^m, \boldsymbol{I}_u^m, \boldsymbol{I}_d^m, \boldsymbol{I}_{i,j}^m, \boldsymbol{I}_{ij}^{m,ext}, \boldsymbol{D}_u^{m,i}$, and $\boldsymbol{D}_v^{m,i}$ for the m_{th}-order grid. We also precompute the reference tangents and normals at the discretization points using the derivative scheme in Section 3.5. Let us denote these reference tangents by $\boldsymbol{X}_{\partial u}^{r,m}, \boldsymbol{X}_{\partial v}^{r,m}$ and the normals by $\boldsymbol{N}^{r,m}$ respectively.
2. At a time instant t, we have the discretization points $\boldsymbol{X}^m$ on the capsule surface. We compute the tangents and the normals at all discretization points in the current configuration using the scheme in Section 3.5. Let us denote these by $\boldsymbol{X}_{\partial u}^m, \boldsymbol{X}_{\partial v}^m$ and $\boldsymbol{N}^m$.
3. Given the reference and the current tangents along with the normals, we compute the coefficients of the first fundamental form, the surface area element $\boldsymbol{W}^m$ and the shear stress tensor $\boldsymbol{\Lambda}^m$.
4. We compute the interfacial force $\boldsymbol{F}^m = \nabla_\gamma \cdot \boldsymbol{\Lambda}^m$ using the derivative scheme in Section 3.5.
5. We compute the upsampled quantities $\boldsymbol{X}_u^m = \boldsymbol{I}_u^m \boldsymbol{X}^m, \boldsymbol{F}_u^m = \boldsymbol{I}_u^m \boldsymbol{F}^m, \boldsymbol{W}_u^m = \boldsymbol{I}_u^m \boldsymbol{W}^m$ and the regularization parameters $\boldsymbol{\delta}_u^m$.
6. We compute the Stokes single layer potential on the upsampled grid, denoted by $\boldsymbol{S}_u^m$, using $\boldsymbol{X}_u^m, \boldsymbol{F}_u^m, \boldsymbol{\Psi}_u^m$ and $\boldsymbol{W}_u^m$.
7. We downsample the Stokes potential $\boldsymbol{S}^m = \boldsymbol{I}_d^m \boldsymbol{S}_u^m$ and add $\boldsymbol{u}_\infty(\boldsymbol{X}^m)$ to get the discretized version of the RHS of Equation (3.35) at time instant t and position $\boldsymbol{X}^m$.

Accuracy and work complexity of the overall algorithm: Our quadrature scheme, differentiation scheme and the time stepping scheme are all fourth order convergent. Hence, our overall scheme is fourth order convergent. We observe this convergence numerically in the convergence results in Table 3. The work complexity of our quadrature scheme and the differentiation scheme is $O(m^4)$ and $O(m^2)$ respectively for m_{th}-order grid. Thus, the overall work complexity of our algorithm for a single time step is $O(m^4)$.

Computation of $||\nabla_{\mathbb{S}^2}\phi||_\infty$: In the results section, we monitor the norm of the surface gradient of $\phi : \mathbb{S}^2 \longrightarrow \gamma$ with respect to the unit sphere $\mathbb{S}^2$ which measures the smoothness of the capsule surface during the simulation. It serves as a proxy for measuring the stability of capsule dynamics during the simulation. This surface gradient $\nabla_{\mathbb{S}^2}\phi$ is calculated using the surface gradient formula given in the Appendix A.2 by using the first fundamental form coefficients E, F, G for the unit sphere and the surface area element W of the unit sphere $\mathbb{S}^2$. We use the derivative scheme in Section 3.5 to calculate the first fundamental form using the stored discrete points $\boldsymbol{X}^{0,m}$ on the sphere and noting that $\phi(\boldsymbol{X}^{0,m}) = \boldsymbol{X}^m$.

4. Results

Now we discuss the various results to verify the accuracy and convergence of our numerical schemes. We also present the results of simulation of extensible capsule under shear flow and the Poiseuille flow. We validate our simulations with the existing results in the literature. We also use the spherical harmonics based numerical scheme used in [36] for quantitative comparisons with our simulations. Below we give a quick summary of the spherical harmonics based scheme used in [36].

Spherical harmonics: Spherical harmonics provide an orthonormal basis [26] for the square-integrable functions defined on the sphere $\mathbb{S}^2$. Hence, they provide a spectral representation of the surfaces and have been used for simulating Stokesian particulate flows in [36]. Below we summarize the number of discretization points used and the work complexity for the differentiation and singular integration in this scheme.

1. *Discretization:* We denote by p the degree of spherical harmonics used to represent the surface and surface fields. For degree p, the scheme uses $2p(p+1)$ number of discretization points.

Table 1
Relative error in the computation of Stokes single layer potential (with density $\boldsymbol{f} = (x^2, y^2, z^2)$ on (a) an ellipsoid and (b) the 4-bump shape (as shown in Fig. 7). $\epsilon_{S[f]}$ is relative error with no upsampling and $\epsilon^{up}_{S[f]}$ is the relative error with 4× upsampling. ϵ_A and ϵ_V are the relative errors in computing area and volume respectively. N is the number of discretization points. Reference solution is computed using $p = 64$ spherical harmonics (see [36]).

m	N	$\epsilon_{S[f]}$	$\epsilon^{up}_{S[f]}$	ϵ_A	ϵ_V
8	294	6×10^{-2}	7×10^{-3}	5×10^{-3}	3×10^{-3}
16	1350	8×10^{-3}	4×10^{-4}	5×10^{-4}	4×10^{-4}
32	5766	4×10^{-4}	2×10^{-5}	1×10^{-5}	1×10^{-5}
64	23814	1×10^{-5}	1×10^{-6}	1×10^{-6}	1×10^{-6}

(a) Relative error in Stokes single layer potential on ellipsoid of reduced volume $v = 0.9$, given $\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1$, where $a = 0.6, b = 1, c = 1$.

m	N	$\epsilon_{S[f]}$	$\epsilon^{up}_{S[f]}$	ϵ_A	ϵ_V
8	294	5×10^{-1}	4×10^{-1}	2×10^{-1}	2×10^{-1}
16	1350	2×10^{-1}	9×10^{-2}	3×10^{-2}	4×10^{-2}
32	5766	3×10^{-2}	8×10^{-3}	2×10^{-3}	3×10^{-3}
64	23814	3×10^{-3}	6×10^{-4}	2×10^{-4}	2×10^{-4}

(b) Relative error in Stokes single layer potential for the 4-bump shape.

2. *Singular quadrature:* The scheme uses Graham-Sloan quadrature [17] to compute the Stokes layer potential which is $O(p^5)$ in work complexity.
3. *Differentiation:* For surface derivatives, the scheme uses spherical harmonics based spectral differentiation which has work complexity $O(p^3)$. Thus, the overall work complexity of the algorithm is $O(p^5)$.

4.1. Integration results

Here, we discuss the numerical accuracy and convergence of our integration schemes discussed in Section 3.3 and Section 3.4. We present the relative error in computing Stokes single layer potential with and without upsampling on two different surfaces. First, we take an ellipsoid surface given by

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1, \tag{4.1}$$

where $a = 0.6, b = 1, c = 1$. The relative error in a quantity q, denoted by ϵ_q, is defined as $\epsilon_q := \frac{||q - q_{\text{ref}}||_\infty}{||q_{\text{ref}}||_\infty}$, where q_{ref} is the reference solution. We use spherical harmonic solutions with degree $p = 64$ as the high accuracy reference solutions for the surface derivatives [36]. We report the relative errors in computing single layer Stokes potential with and without upsampling, denoted by $\epsilon^{up}_{S[f]}$ and $\epsilon_{S[f]}$ respectively, in Table 1a.

As a second example, we take a 4-bump shape which can be written in standard spherical parameters as

$$\boldsymbol{X}(u, v) = \rho(u, v)\begin{pmatrix} \sin u \cos v \\ \sin u \sin v \\ \cos u \end{pmatrix}, \ \forall u \in [0, \pi] \times v \in [0, 2\pi) \tag{4.2}$$

where $\rho(u, v) = 1 + e^{-3Re(Y_{lm}(u,v))}$, *with* $l = 3, m = 2$. This 4-bump shape is shown in Fig. 7. We report the relative errors in Stokes single layer potential on this surface in Table 1b.

The upsampling improves the accuracy of the integration scheme and helps us to long time-horizon simulations that we discuss in the later sections. As evident from the tabulated results, upsampling for singular quadrature is required to get similar digits of accuracy as the derivative accuracy[1] (Table 2a and Table 2b). The reference solution is computed using the Graham-Sloan quadrature [17,36] for spherical harmonics order $p = 64$. We also present the relative errors in computing area A and volume V[2] to demonstrate the convergence and accuracy of our integration scheme for smooth functions. For further verification and to study the change in relative errors with the reduced volume v, we provide the error in our upsampled quadrature scheme in Appendix A.6. In Appendix A.5, we also tabulate the relative errors demonstrating the convergence of our derivative and integration schemes for the complex shapes obtained in actual shear and Poiseuille flow simulations discussed in the later sections. This further verifies the correctness of our code.

4.2. Derivative results

Now we discuss the results showing the accuracy and convergence of our numerical differentiation scheme. We report the relative errors in derivative calculations on the ellipsoid and the 4-bump shape in Table 2a and Table 2b respectively. The derivatives converge as the grid length h_m decreases. Empirically, the convergence is of the fourth order which is what we predicted because of the cubic

[1] The digits of accuracy for mean curvature H and Gaussian curvature K are lower than the upsampled quadrature but since they are not required in shear force calculation, we don't need upsampling for derivatives. However, since the bending forces [36] require curvature calculations, upsampling for derivatives could be desirable if bending force is to be included.

[2] Volume V of surface γ is given by $V = \int_{enc(\gamma)} dV$, where $enc(\gamma)$ refers to the volume enclosed by γ. Using divergence theorem, we can write it as a smooth surface integral as $V = \int_\gamma \boldsymbol{M}(x, y, z) \cdot \boldsymbol{n} d\gamma$, where $\boldsymbol{M}(x, y, z) = (x, 0, 0)$.

Fig. 7. The 4-bump shape with its first three patches shaded with dark mesh.

Table 2
Relative error in the derivative scheme for (a) an ellipsoid and (b) the 4-bump shape (shown in Fig. 7). We report relative error in surface normals $\boldsymbol{n}$, mean curvature H, Gaussian curvature K and surface divergence div_γ. N is the number of discretization points. Spherical harmonics results for $p = 64$, using the algorithms in [36], are used as true values and the error is computed relative to those values. Surface divergence is computed for the smooth function $\boldsymbol{g}(x, y, z) = (x^2, y^2, z^2)$ on the surface.

m	N	$\epsilon_{\boldsymbol{n}}$	ϵ_H	ϵ_K	ϵ_{div_γ}
8	294	5×10^{-3}	4×10^{-3}	6×10^{-3}	4×10^{-2}
16	1350	2×10^{-4}	5×10^{-4}	1×10^{-4}	3×10^{-3}
32	5766	1×10^{-5}	3×10^{-5}	6×10^{-5}	1×10^{-4}
64	23814	7×10^{-7}	2×10^{-6}	3×10^{-6}	1×10^{-5}

(a) Relative error in derivatives for an ellipsoid of reduced volume $v = 0.9$, given by $\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1$, where $a = 0.6, b = 1, c = 1$.

m	N	$\epsilon_{\boldsymbol{n}}$	ϵ_H	ϵ_K	ϵ_{div_γ}
8	294	2×10^{-1}	3×10^{-1}	8×10^{-1}	6×10^{-1}
16	1350	6×10^{-2}	1×10^{-1}	1×10^{-1}	3×10^{-1}
32	5766	6×10^{-3}	1×10^{-2}	2×10^{-2}	6×10^{-2}
64	23814	5×10^{-4}	1×10^{-3}	2×10^{-3}	5×10^{-3}

(b) Relative error in derivatives for the 4-bump shape.

Table 3
Self-convergence results for an extensible capsule simulation under background (a) shear flow (given by $\boldsymbol{u}_\infty(x, y, z) = (y, 0, 0)$) and (b) Poiseuille flow (given by $\boldsymbol{u}_\infty(x, y, z) = ((25 - y^2 - z^2), 0, 0)$). The initial shape is an ellipsoid given by $x^2/a^2 + x^2/b^2 + x^2/c^2 = 1$, where $a = 0.9, b = 1.0, c = 1.0$. We set the shear modulus to be $E_s = 2$ and the dilatation modulus to be $E_D = 20$. We simulate the capsule for a time horizon $[0, T]$ and report the relative errors in area A, volume V and moment of inertia tensor J of the capsule shape at time $T = 0.5$. The reference solution for the relative error is the numerical solution computed using the $m = 48$ grid. We use Runge-Kutta-Fehlberg time stepping scheme (see Section 3.6) with fixed relative tolerance of $\epsilon_{tol} = 10^{-6}$.

m	ϵ_A	ϵ_V	ϵ_J
8	2×10^{-2}	4×10^{-2}	3×10^{-2}
16	2×10^{-3}	1×10^{-3}	2×10^{-3}
32	1×10^{-4}	1×10^{-4}	1×10^{-4}

(a) Relative errors in the capsule dynamics driven by a shear flow.

m	ϵ_A	ϵ_V	ϵ_J
8	2×10^{-2}	4×10^{-2}	2×10^{-2}
16	3×10^{-3}	1×10^{-3}	2×10^{-3}
32	2×10^{-4}	1×10^{-4}	1×10^{-4}

(b) Relative errors in the capsule dynamics driven by a background Poiseuille flow.

spline interpolation used in patch extension operators and blending. We also provide verification of derivatives for ellipsoids of different reduced volumes and tabulate the results in Appendix A.6.

4.3. Accuracy and convergence of the full numerical scheme

Now, we discuss the accuracy and convergence of our full numerical solver for the extensible capsule simulation. To this end, we take an ellipsoidal capsule with an initial surface configuration given by $\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1$, where $a = 0.9, b = 1, c = 1$. The initial shape is taken to be the stress-free reference configuration. We do the simulations under two different imposed background flows, *i.e.,* shear flow and Poiseuille flow. The flows are described mathematically as

$$\boldsymbol{u}_\infty(x, y, z) = \dot{\gamma}(y, 0, 0), \ \textit{(shear flow)}\ , \tag{4.3}$$

$$\boldsymbol{u}_\infty(x, y, z) = (\alpha(R_0^2 - y^2 - z^2), 0, 0)\ \textit{(Poiseuille flow)}\ , \tag{4.4}$$

where $\dot{\gamma}$ is the shear rate of the shear flow, α controls the curvature of the Poiseuille flow and R_0 is the radius of the circular cross-section of the Poiseuille flow. We set $\dot{\gamma} = 1$, $\alpha = 1$ and $R_0 = 5$ for these simulations. We use the differentiation, integration and the time stepping schemes discussed above to simulate these setups for a fixed time horizon $[0, T]$ using our solver. In Table 3, we tabulate the relative errors in area A, volume V and moment of inertia tensor J of the final capsule shape for different values of the discretization order m. We observe fourth order convergence in the relative errors. The parachute shapes at time $T = 0.5$ for the Posieuille flow simulations for different levels of discretization are given in Fig. 8.

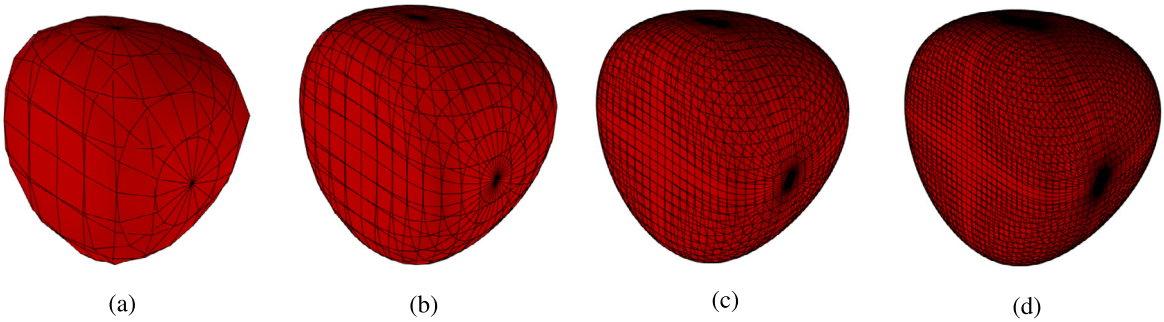


Fig. 8. The parachute shape under background Poiseuille flow (see Equation (4.4)) obtained after time $T = 0.5$ for different levels of discretization (a) $m = 8$, (b) $m = 16$, (c) $m = 32$ and (d) $m = 48$ for $E_s = 2, E_D = 20$. The initial shape is same as the stress-free reference state and is given by $x^2/a^2 + x^2/b^2 + x^2/c^2 = 1$, where $a = 0.9, b = 1.0, c = 1.0$.

4.4. Relaxation of capsule

Now, we simulate the relaxation of a capsule to its stress-free reference state to validate our code. We take a capsule with an initial shape of a unit sphere. The initial shape is also the stress-free reference configuration of the capsule. We simulate the dynamics of the capsule under a background linear shear flow with shear rate $\dot{\gamma} = 1$ for a fixed time horizon $[0, T_1]$ followed by a zero background flow velocity for $[T_1, T]$. We report the resulting shapes at different times in Fig. 9. We observe that once the background flow diminishes the capsule returns to the stress-free state, *i.e.,* a unit sphere. Using the moment of inertia tensor J and volume V of the capsule, we compute the instantaneous Taylor asphericity parameter [21] D_a of the capsule shape defined as

$$D_a = \frac{L - S}{L + S}, \tag{4.5}$$

$$\text{where } S = \sqrt{\frac{J_{xx} + J_{yy} - \sqrt{(J_{xx} - J_{yy})^2 + 4J_{xy}^2}}{2V}}, \; L = \sqrt{\frac{J_{xx} + J_{yy} + \sqrt{(J_{xx} - J_{yy})^2 + 4J_{xy}^2}}{2V}}. \tag{4.6}$$

For a sphere, $D_a = 0$. We plot the Taylor asphericity as a function of time to monitor the relaxation back to the reference unit sphere shape and plot it in Fig. 9d. As expected, we observe that the capsule relaxes back to a unit sphere when the background flow is removed as D_a drops back to zero. Our results agree quantitatively with the results obtained using spherical harmonics based scheme used in [36].

4.5. Capsule in shear flow

Now, we present results for steady shapes of extensible capsule under background shear flow (see Equation (4.3)). We start with a stress-free spherical capsule. Under linear shear flow, the capsule is known to take a terminal nearly-ellipsoidal shape and exhibit a stable tank treading motion [10,21]. The terminal shape and inclination angle of its major axes with the flow direction depends on three key parameters, namely, the shear rate $\dot{\gamma}$ of the flow, the shear modulus E_s and the dilatation modulus E_D of the membrane. We simulate these dynamics for different values of these parameters and plot the terminal inclination angles θ (with respect to the flow direction) in Fig. 10a. We compare our results with the numerical results from [10] and obtain good quantitative agreement. We also plot the evolution of Taylor asphericity D_a with time in Fig. 10b and observe good agreement with the numerical results obtained using the spherical harmonics based scheme mentioned in [36]. Figs. 11 to 13 shows the terminal shapes of different reduced volumes obtained in shear flows using our code. Our code is able to resolve shapes for different reduced volumes obtained for a wide range of ratios of shear modulus E_s and dilatation modulus E_D.

To further demonstrate the effectiveness of four times upsampling, we plot the norm of the gradient of mapping ϕ from a unit sphere to the capsule surface γ with time in shear flow simulations for different grid orders m in Fig. 14. The gradient norm serves as a proxy for the stability of the capsule shape. As evident from the plots, we need high numerical accuracy to do long time horizon simulations. We observe that grid order $m = 16$ with four times upsampling is sufficient to do shear flow simulations. Lower upsampling factors or lower grid order m results in unstable gradients which blow up over long time scales as shown in the Fig. 14.

4.6. Capsule in Poiseuille flow

In Fig. 16, we present the terminal shapes for a capsule under Poiseuille flow (see Equation (4.4)) for different membrane elasticity parameters leading to shapes of reduced volume as low as $v = 0.6$. As for the shear flow simulations, we plot the norm of the gradient of mapping ϕ from a unit sphere to the capsule surface γ with time for the Poiseuille flow simulations for different grid orders m in Fig. 15. We observe that grid order $m = 32$ with four times upsampling is sufficient to do Poiseuille flow simulations. Lower upsampling factors or lower grid order m results in unstable gradients which blow up over long time scales as shown in the Fig. 15.

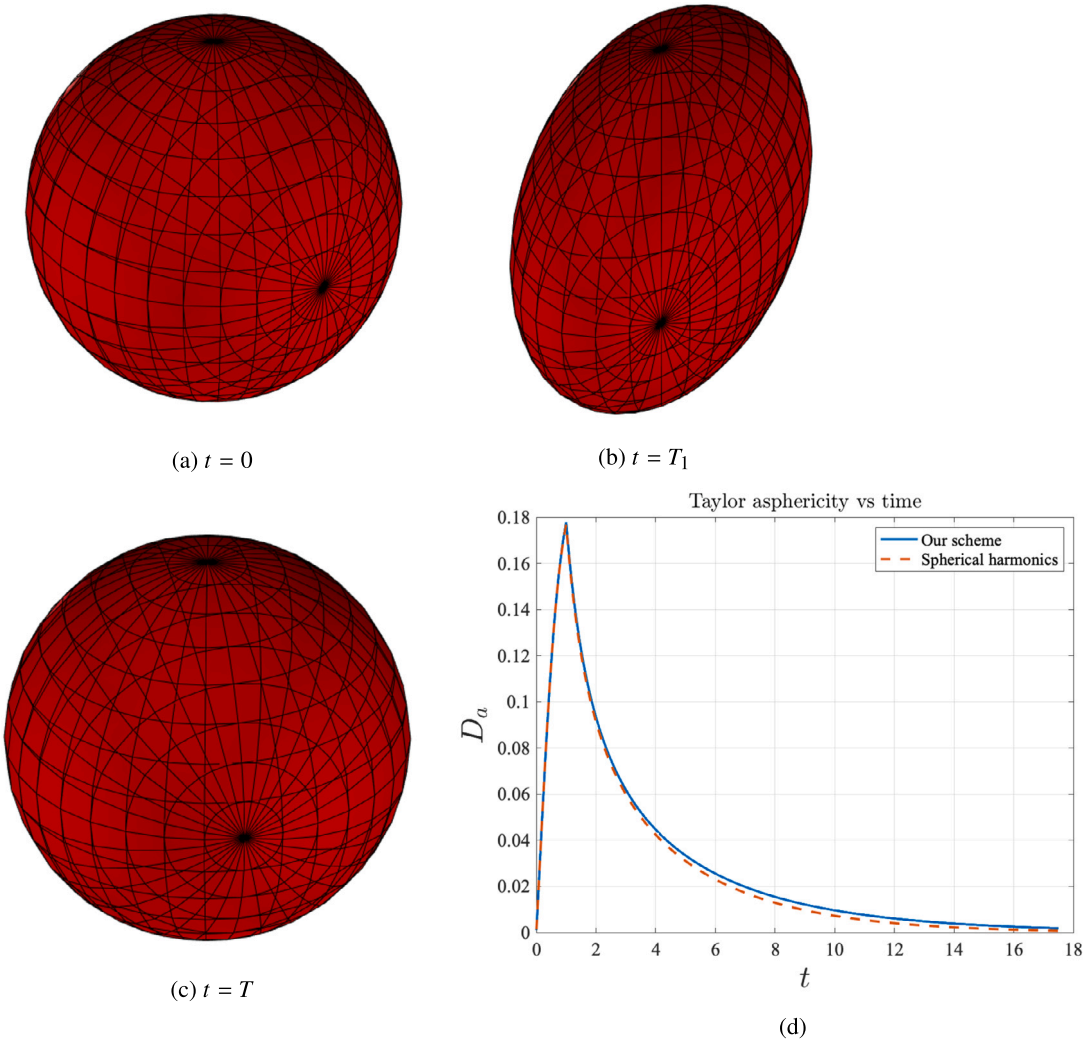


Fig. 9. Shapes obtained at different times during simulation of the relaxation of the capsule with initial spherical shape as the stress-free reference state. We impose background shear flow $\boldsymbol{u}_\infty(x, y, z) = (y, 0, 0)$ for time horizon $[0, T_1]$. This is followed by zero background velocity from $[T_1, T]$. The capsule relaxes back to the stress-free reference shape as expected. We take $m = 16, T_1 = 1, T = 17.5, E_s = 2$ and $E_D = 20$. (a) Shape at time $t = 0$. (b) Shape at time $t = T_1$. (c) Shape at time $t = T$. (d) The plot of Taylor asphericity D_a vs time t, where D_a increases till $t = 1$ under background shear flow and then drops to zero during the relaxation phase when the background flow is zero. Blue solid lines are the plot using our scheme and red dotted lines denote the plot using spherical harmonics based scheme (with spherical harmonic degree $p = 32$).

4.7. Timing results

We compute the GPU wall clock times required for our scheme and give its breakdown over different stages of our scheme. To this end, we take an initially spherical capsule in stress free state (with $E_s = 2, E_D = 20$) and simulate its dynamics under shear flow with shear rate $\dot{\gamma} = 1$ till time $T = 0.1$. We do these simulations for different grid order m and plot the total time and its breakdown for different stages in Fig. 17b. We observe that for $m \geq 16$, majority of the time is required to compute the singular quadrature. In fact, for $m \geq 32$ virtually all of the time is consumed in computing the quadrature. The wall clock times required to compute the singular quadrature once are also plotted separately in Fig. 17a. We also provide a comparison of the wall clock time per time step taken by our GPU accelerated scheme with the spherical harmonics CPU code [36] in Appendix A.8.

5. Fast multipole method based acceleration

Our quadrature does not involve a product quadrature and is amenable to acceleration via fast multipole method (FMM) [35]. A full FMM acceleration reduces the time complexity of our scheme to $O(N)$ allowing us to perform high resolution simulations. We do not implement a multilevel FMM for the purpose of this work and focus on showcasing the correctness and advantages of our numerical scheme using a simple single level FMM acceleration for our scheme. Based on the kernel independent FMM of [42], we describe our scheme below.

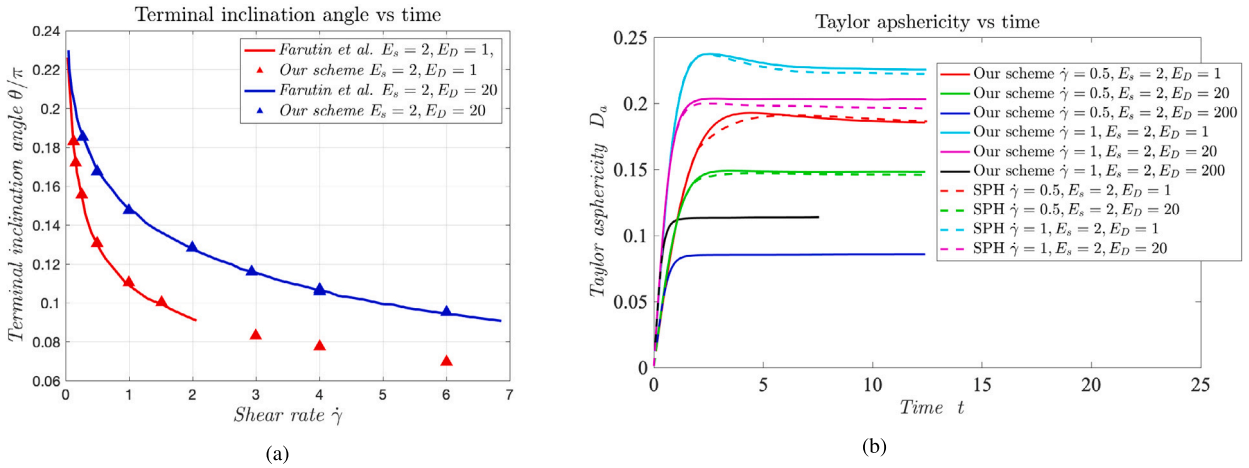


Fig. 10. Validation of shear flow simulation results computed using $m = 16$. (a) Plot of terminal inclination angle θ for different shear rates and membrane mechanical properties (E_s and E_D). We compare our results with results in [10] and obtain good quantitative agreement. (b) We plot the evolution of Taylor asphericity D_a of the capsule vs time and compare our results (solid lines) with the spherical harmonics ($p = 32$) based scheme in [36] (dashed lines). We observe good quantitative agreement. For $E_D = 200$, the spherical harmonics code is unable to resolve the shapes and the surface fields and therefore, we do not provide plots for those.

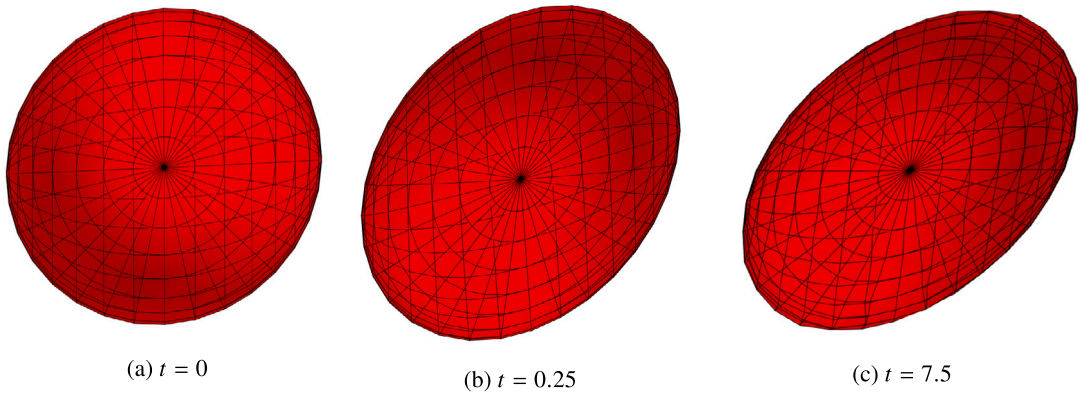


Fig. 11. Snapshots of capsule shape at different time instants for shear flow simulation with $\dot{\gamma} = 1, E_s = 2, E_D = 200$. (a) $t = 0$, (b) $t = 0.25$ and (c) $t = 7.5$. The shape shown in (c) is the terminal shape.

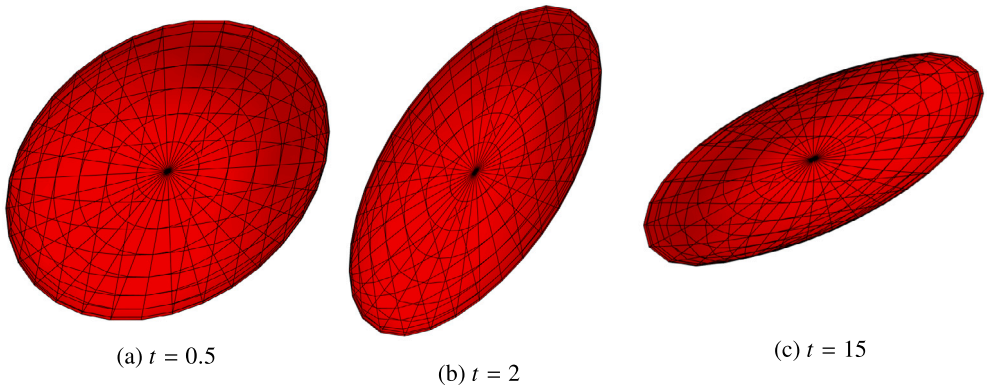
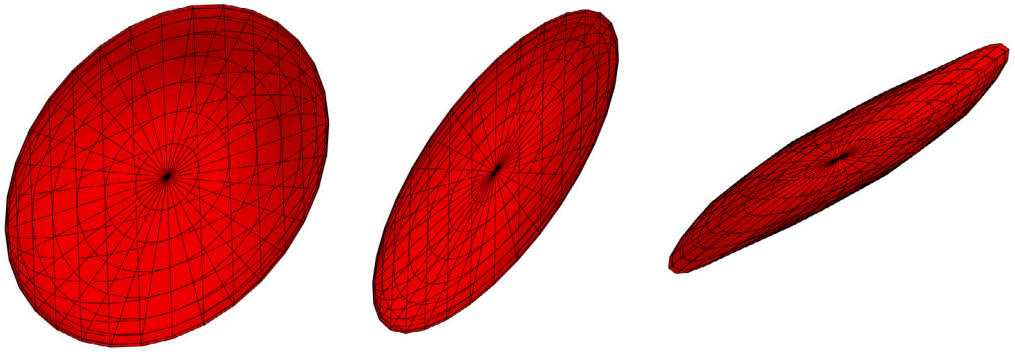
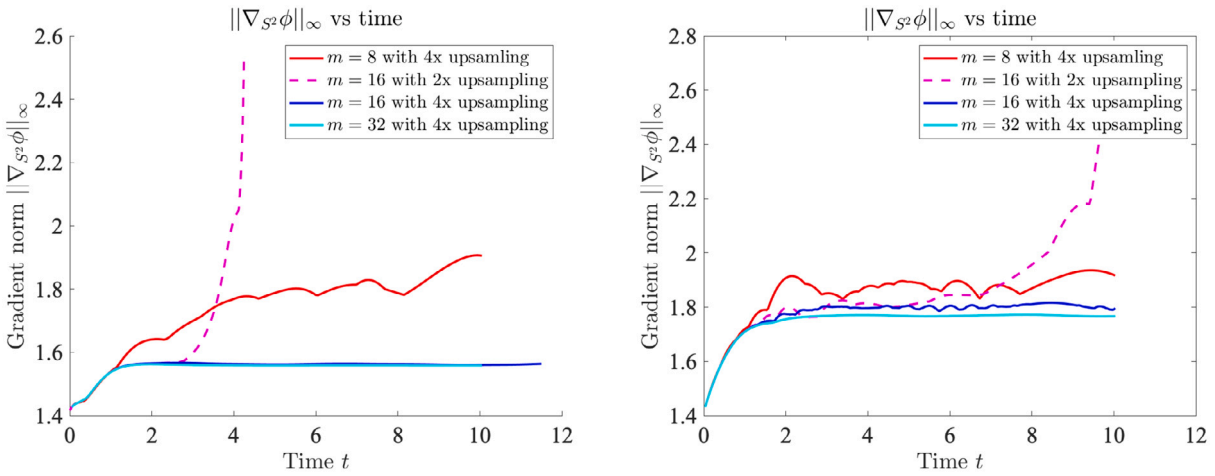


Fig. 12. Snapshots of capsule shape at different time instants for shear flow simulation with $\dot{\gamma} = 0.5, E_s = 2, E_D = 1$. (a) $t = 0.5$, (b) $t = 2$ and (c) $t = 15$. The shape shown in (c) is the terminal shape.



(a) $t = 0.0417$ (b) $t = 0.25$ (c) $t = 1.25$

Fig. 13. Snapshots of capsule shape at different time instants for shear flow simulation with $\dot{\gamma} = 6, E_s = 2, E_D = 1$. (a) $t = 0.0417$, (b) $t = 0.25$ and (c) $t = 1.25$. The shape shown in (c) is the terminal shape.

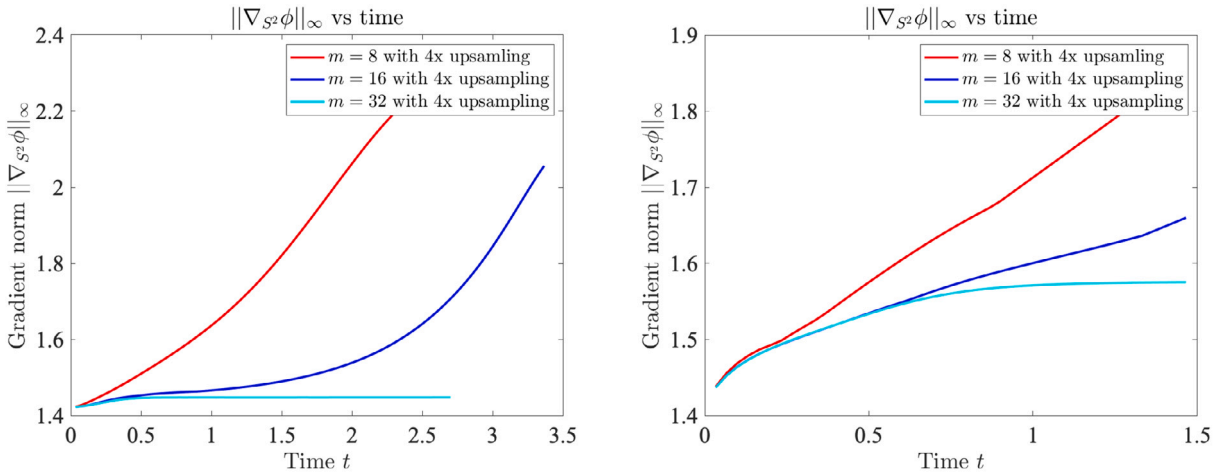


(a) Terminal reduced volume $\nu = 0.85$. (b) Terminal reduced volume $\nu = 0.65$.

Fig. 14. Plots of the $||\nabla_{\mathbb{S}^2}\phi||_\infty$ vs time t in shear flow simulation where $\phi : \mathbb{S}^2 \longrightarrow \gamma$ is the mapping from the unit sphere to the capsule γ. Initial shape is stress-free unit sphere with (a) $\dot{\gamma} = 1$, $E_s = 2$ and $E_D = 20$ and (b) $\dot{\gamma} = 1.5$, $E_s = 2$ and $E_D = 1$. The plots show the effectiveness of four times upsampling in doing stable long time horizon simulations while two times upsampling fails for $m = 16$.

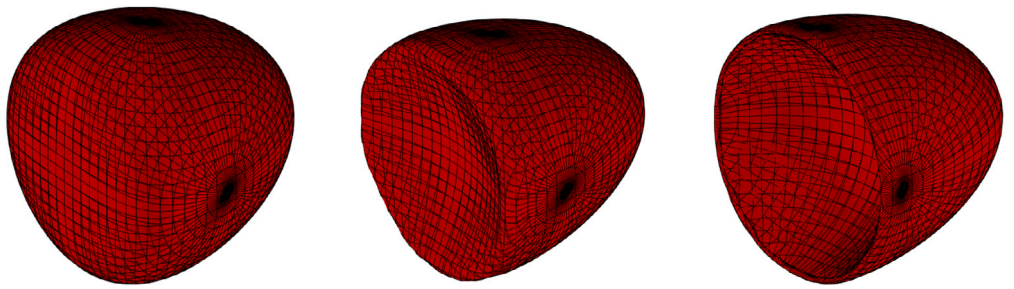
1. We first use a k-means clustering algorithm to cluster together the discretization points on the capsule surface for a reasonable value of k. We will have k clusters. We found that $k = 100$ gives us relative accuracy of 10^{-5}, which is sufficient for the simulations in this paper.
2. Each cluster is imagined to be enclosed by a surrounding cubical box, called an equivalent surface, with equivalent sources placed on a regular grid on the boundary of this cubical box. Thus, we have k equivalent surfaces and let each equivalent surface have N_{eq} number of equivalent sources. We also consider a slightly bigger cube surrounding this enclosing cube which will be used as a check surface. The density on the equivalent sources lying on the equivalent surface is estimated by equating the potential on the check surface due to the actual point sources (*i.e.,* discretization points) on the part of the capsule surface lying inside that box or equivalent surface.
3. We apply direct all pairs single layer potential calculation for the neighboring boxes while the equivalent sources are used to calculate the single layer potential for the discretization points in the far field (*i.e.,* the discretization points inside non-neighboring boxes).

We present convergence and speedup results for calculating single layer on an ellipsoid of reduced volume $\nu = 0.9$ in Table 4a. While we do not see speedup for low values of m because of FMM overhead computations, we see about two times speedup for $m = 96$ using our implementation of the single level FMM on GPU.



(a) Terminal reduced volume $\nu = 0.96$. (b) Terminal reduced volume $\nu = 0.86$.

Fig. 15. Plots of the $||\nabla_{\mathbb{S}^2}\phi||_\infty$ vs time t in Poiseuille flow simulation where $\phi : \mathbb{S}^2 \longrightarrow \gamma$ is the mapping from the unit sphere to the capsule γ. Initial shape is stress-free unit sphere with (a) $\alpha = 1.5$, $R_0 = 5$, $E_s = 2$ and $E_D = 200$ and (b) $\alpha = 1.5$, $R_0 = 5$, $E_s = 2$ and $E_D = 20$. Plots show that $m = 32$ with four times upsampling gives stable simulations for Poiseuille flow while lower values of m are not enough to resolves the shapes in Poiseuille flow.



(a) $\nu = 0.96$ (b) $\nu = 0.85$ (c) $\nu = 0.60$

Fig. 16. Terminal parachute shapes of varying reduced volumes ν under Poiseuille flow. (a) Terminal shape of reduced volume $\nu = 0.96$ obtained for Poiseuille flow simulation with $\alpha = 1.5$, $E_s = 2$, $E_D = 200$. (b) Terminal shape of reduced volume $\nu = 0.85$ obtained for Poiseuille flow simulation with $\alpha = 1.5$, $E_s = 2$, $E_D = 20$. (c) Terminal shape of reduced volume $\nu = 0.6$ obtained for shear flow simulation with $\alpha = 1.5$, $E_s = 2$, $E_D = 1$. We use grid order $m = 32$ and the Poiseuille flow cross-section radius $R_0 = 5$ in these simulations.

Table 4
Relative error and speedup for FMM accelerated simulation (with 4x upsampling) of a capsule suspended in Poiseuille flow as in Fig. 16c. All pairs direct calculation for $m = 32$ (with 4x upsampling) is used as the reference solution and the relative error is computed in the moment of inertia tensor of the capsule at the end of simulation. Time reported is wall clock time in seconds taken by one time step of the simulation. t_{direct} is direct calculation time and t_{FMM} is time taken with the FMM acceleration. Number of equivalent surfaces or boxes is fixed at 100. We vary the number of equivalent sources for an equivalent surface, denoted by N_{eq}, with m.

m	N	N_{eq}	t_{direct}	t_{FMM}	Speedup	ϵ_{FMM}
32	5766	96	10.78	10.66	1.02	6×10^{-3}
64	23814	128	163.21	104.20	1.57	4×10^{-5}
96	54150	256	905.31	501.76	1.81	5×10^{-6}

6. Conclusions

In this work, we described a novel numerical scheme to simulate Stokesian particulate flows. We described an overlapping patch based discretization of the surfaces diffeomorphic to the unit sphere. Our numerical scheme uses the regularized Stokes kernels [34] and finite differences on overset grid to calculate the Stokes layer potential and interfacial elastic forces. We presented a battery of

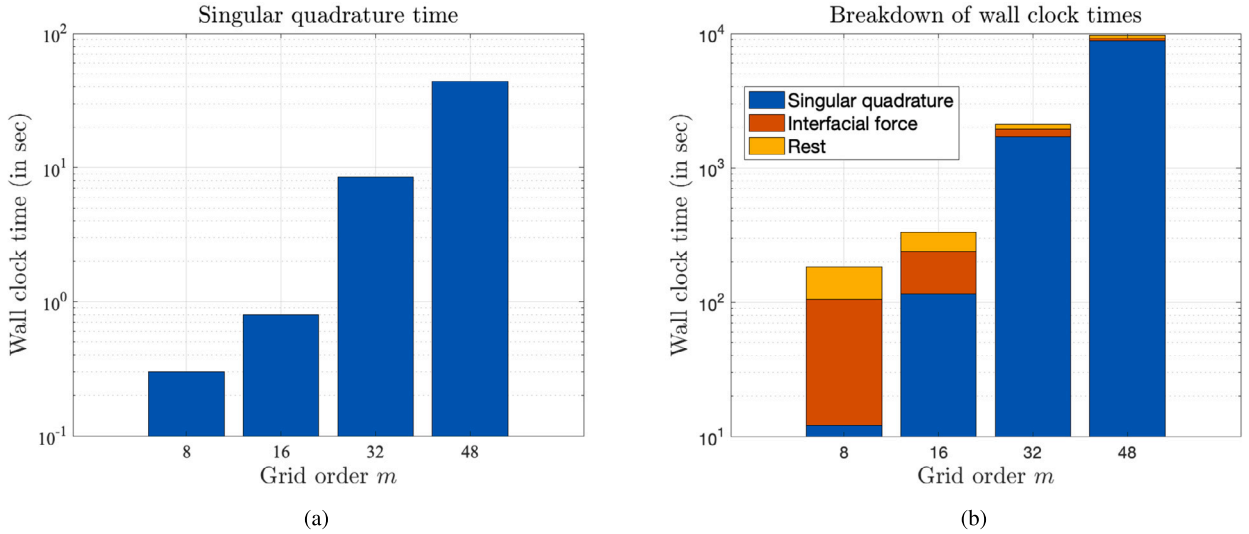


Fig. 17. (a) GPU wall clock times for computing singular layer quadrature with four times upsampling for different grid orders $m = 8, 16, 32, 48$. (b) Breakdown of wall clock times for shear flow simulation (with initial shape as unit sphere over time horizon $T = 0.1$) over different stages (computation of singular quadrature, computation of interfacial forces and the rest of the algorithm) for different grid orders m.

results for the verification of our numerical scheme using results from previous literature. We used our numerical scheme to simulate an extensible capsule suspended in Stokes flow and validated our simulations. Our numerical scheme is a fourth order convergent scheme that is $O(N^2)$ in work complexity where N is the number of discretization points and can be accelerated to run with $O(N)$ work complexity using a multilevel FMM. This is much better than the asymptotic work complexity of the spherical harmonics based spectral scheme [36]. Our scheme also allows for independent control over local resolution due to patch based parameterization of surface. We used GPU acceleration to demonstrate the ability of our code to simulate the complex shapes with high resolution.

In this paper, we implemented a single level FMM to demonstrate FMM based speedup. A GPU based implementation of a multilevel FMM will further enable us to do highly accurate simulations in a reasonable time and help us better study the Stokesian particulate flows. We leave it as the subject of our future work.

CRediT authorship contribution statement

Dhwanit Agarwal: Formal analysis, Methodology, Software, Validation, Visualization, Writing – original draft. **George Biros:** Conceptualization, Funding acquisition, Project administration, Supervision, Writing – review & editing.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Dhwanit Agarwal reports financial support was provided by The University of Texas at Austin. George Biros reports a relationship with The University of Texas at Austin that includes: employment.

Data availability

Data will be made available on request.

Acknowledgements

Dhwanit Agarwal and George Biros' research was supported in part by UT-Austin CoLab and UTEN Partnership, and the Portuguese Science and Technology Foundation under funding #201801976/UTA18-001217.

Appendix A

A.1. Transition maps

Here, we list the expressions for the transition maps τ_{ij} for $i = 1, \ldots 6$, for the parameterization we use. The expressions are given as follows:

$$\tau_{12}(u,v) = (u, \pi + v), \tau_{13}(u,v) = (u, \frac{\pi}{2} + v), \tau_{14}(u,v) = (u, v - \frac{\pi}{2}), \tag{A.1}$$

$$\tau_{15}(u,v) = \left(\cos^{-1}(-\sin u \sin v), \cos^{-1}\left(\frac{\sin u \cos v}{\sqrt{\sin^2 u \cos^2 v + \cos^2 u}} \right) \right), \tag{A.2}$$

$$\tau_{16}(u,v) = \left(\cos^{-1}(\sin u \sin v), \cos^{-1}\left(\frac{\sin u \cos v}{\sqrt{\sin^2 u \cos^2 v + \cos^2 u}} \right) \right). \tag{A.3}$$

The transition maps for the next three patches are given below:

$$\tau_{21}(u,v) = (u, \pi + v), \tau_{23}(u,v) = (u, v - \frac{\pi}{2}), \tau_{24}(u,v) = (u, v + \frac{\pi}{2}), \tag{A.4}$$

$$\tau_{25}(u,v) = \left(\cos^{-1}(\sin u \sin v), \cos^{-1}\left(\frac{-\sin u \cos v}{\sqrt{\sin^2 u \cos^2 v + \cos^2 u}} \right) \right), \tag{A.5}$$

$$\tau_{26}(u,v) = \left(\cos^{-1}(-\sin u \sin v), \cos^{-1}\left(\frac{-\sin u \cos v}{\sqrt{\sin^2 u \cos^2 v + \cos^2 u}} \right) \right), \tag{A.6}$$

$$\tau_{31}(u,v) = (u, v - \pi/2), \tau_{32}(u,v) = (u, v + \frac{\pi}{2}), \tau_{34}(u,v) = (u, v + \pi), \tag{A.7}$$

$$\tau_{35}(u,v) = \left(\cos^{-1}(\sin u \cos v), \cos^{-1}\left(\frac{\sin u \sin v}{\sqrt{\sin^2 u \sin^2 v + \cos^2 u}} \right) \right), \tag{A.8}$$

$$\tau_{36}(u,v) = \left(\cos^{-1}(-\sin u \cos v), \cos^{-1}\left(\frac{\sin u \sin v}{\sqrt{\sin^2 u \sin^2 v + \cos^2 u}} \right) \right), \tag{A.9}$$

$$\tau_{41}(u,v) = (u, v + \pi/2), \tau_{42}(u,v) = (u, v - \frac{\pi}{2}), \tau_{43}(u,v) = (u, v + \pi), \tag{A.10}$$

$$\tau_{45}(u,v) = \left(\cos^{-1}(-\sin u \cos v), \cos^{-1}\left(\frac{-\sin u \sin v}{\sqrt{\sin^2 u \sin^2 v + \cos^2 u}} \right) \right), \tag{A.11}$$

$$\tau_{46}(u,v) = \left(\cos^{-1}(\sin u \cos v), \cos^{-1}\left(\frac{-\sin u \sin v}{\sqrt{\sin^2 u \sin^2 v + \cos^2 u}} \right) \right). \tag{A.12}$$

The transition maps for $\mathcal{P}_5^0$ and $\mathcal{P}_6^0$ are given as follows:

$$\tau_{51}(u,v) = \left(\cos^{-1}(\sin u \sin v), \cos^{-1}\left(\frac{\sin u \cos v}{\sqrt{\sin^2 u \cos^2 v + \cos^2 u}} \right) \right), \tag{A.13}$$

$$\tau_{52}(u,v) = \left(\cos^{-1}(\sin u \sin v), \cos^{-1}\left(\frac{-\sin u \cos v}{\sqrt{\sin^2 u \cos^2 v + \cos^2 u}} \right) \right), \tag{A.14}$$

$$\tau_{53}(u,v) = \left(\cos^{-1}(\sin u \sin v), \cos^{-1}\left(\frac{\cos u}{\sqrt{\sin^2 u \cos^2 v + \cos^2 u}} \right) \right), \tag{A.15}$$

$$\tau_{54}(u,v) = \left(\cos^{-1}(\sin u \sin v), \cos^{-1}\left(\frac{-\cos u}{\sqrt{\sin^2 u \cos^2 v + \cos^2 u}} \right) \right), \tag{A.16}$$

$$\tau_{54}(u,v) = (\pi - u, v), \tag{A.17}$$

$$\tau_{61}(u,v) = \left(\cos^{-1}(-\sin u \sin v), \cos^{-1}\left(\frac{\sin u \cos v}{\sqrt{\sin^2 u \cos^2 v + \cos^2 u}} \right) \right), \tag{A.18}$$

$$\tau_{62}(u,v) = \left(\cos^{-1}(-\sin u \sin v), \cos^{-1}\left(\frac{-\sin u \cos v}{\sqrt{\sin^2 u \cos^2 v + \cos^2 u}} \right) \right), \tag{A.19}$$

$$\tau_{63}(u,v) = \left(\cos^{-1}(-\sin u \sin v), \cos^{-1}\left(\frac{\cos u}{\sqrt{\sin^2 u \cos^2 v + \cos^2 u}} \right) \right), \tag{A.20}$$

$$\tau_{64}(u,v) = \left(\cos^{-1}(-\sin u \sin v), \cos^{-1}\left(\frac{-\cos u}{\sqrt{\sin^2 u \cos^2 v + \cos^2 u}} \right) \right), \tag{A.21}$$

$$\tau_{65}(u,v) = (\pi - u, v). \tag{A.22}$$

Table 5
Formulas used for computing various surface derivative quantities.

Symbol	Definition	Symbol	Definition
E	$\boldsymbol{x}_u \cdot \boldsymbol{x}_u$	M	$\boldsymbol{x}_{uv} \cdot \boldsymbol{n}$
F	$\boldsymbol{x}_u \cdot \boldsymbol{x}_v$	N	$\boldsymbol{x}_{vv} \cdot \boldsymbol{n}$
G	$\boldsymbol{x}_v \cdot \boldsymbol{x}_v$	H	$\frac{EN-2FM+GL}{2W^2}$
W	$\sqrt{EG-F^2}$	$\nabla_\gamma g$	$\frac{Gx_u-Fx_v}{W^2} g_u + \frac{Ex_v-Fx_u}{W^2} g_v$
$\boldsymbol{n}$	$\frac{\boldsymbol{x}_u \times \boldsymbol{x}_v}{W}$	$\nabla_\gamma \cdot \boldsymbol{g}$	$\frac{Gg_u-Fg_v}{W^2} \boldsymbol{x}_u + \frac{Eg_v-Fg_u}{W^2} \boldsymbol{x}_v$
L	$\boldsymbol{x}_{uu} \cdot \boldsymbol{n}$	K	$\frac{LN-M^2}{W^2}$

Table 6
Relative errors in surface derivatives. We compare the relative errors for different values of r_0.

m	r_0/π	$\epsilon_{\boldsymbol{n}}$	ϵ_H
8	5.5/12	2×10^{-4}	1.3×10^{-3}
8	5/12	2×10^{-4}	1.5×10^{-3}
8	4/12	4×10^{-4}	1.6×10^{-3}
16	5.5/12	8×10^{-6}	1.7×10^{-5}
16	5/12	7×10^{-6}	1.8×10^{-5}
16	4/12	7×10^{-6}	1.9×10^{-5}
32	5.5/12	4×10^{-7}	1.3×10^{-6}
32	5/12	4×10^{-7}	1.7×10^{-6}
32	4/12	4×10^{-7}	1.9×10^{-6}

Table 7
Relative error in computing surface normals n and the mean curvature H for unit sphere with and without the blending. ϵ^{b} are the errors with blending and ϵ^{nb} are the errors without the blending process in computing derivatives.

m	$\epsilon^{\text{b}}_{\boldsymbol{n}}$	ϵ^{b}_H	$\epsilon^{\text{nb}}_{\boldsymbol{n}}$	ϵ^{nb}_H
8	2×10^{-4}	1.3×10^{-3}	3×10^{-3}	2×10^{-2}
16	8×10^{-6}	1.7×10^{-5}	9×10^{-5}	2×10^{-3}
32	4×10^{-7}	1.3×10^{-6}	3×10^{-6}	3×10^{-4}

A.2. Surface derivative formulas

Here, we list the formulas for the first fundamental form coefficients E, F, G, the unit normal $\boldsymbol{n}$, the second fundamental form coefficients L, M, N, the mean curvature H, the Gaussian curvature K, the surface divergence (of a vector field $\boldsymbol{g}$) and the surface gradient (of a scalar function g) in terms of the local parameterization $\boldsymbol{x}(u,v) : D \subset \mathbb{R}^2 \longrightarrow \gamma$. We use these quantities in the computation of interfacial force and the verification of surface derivatives (see Table 5).

A.3. The choice of parameter r_0 for the partition of unity

Here, we briefly discuss our choice of parameter r_0 in the construction of partition of unity in Section 3.1. We note here that for the specific parameterization of the unit sphere we use (see Equation (3.8)), we require $r_0 > \frac{3\pi}{12}$ to ensure that every point on $\mathbb{S}^2$ belongs to the support of ψ_i^0 for at least one $i \in \{1,\dots,6\}$. If this is not the case, then $\{\psi_i^0\}_{i=1}^6$ ceases to be a partition of unity on the unit sphere. Additionally, we need $r_0 < \frac{\pi}{2}$ so that the support of each ψ_i^0 is compactly contained in $\mathcal{P}_i^0$. We tabulate some numerical results Table 6 for the relative errors in the surface derivatives on the unit sphere for different values of $\pi > r_0 > \frac{3\pi}{12}$. We find that $r_0 = \frac{5\pi}{12}$ gives optimal accuracy among the values of r_0 we experimented with.

A.4. Effect of the blending process in calculating derivatives

Here, we provide the relative errors in the computing normals n and the mean curvature H for the unit sphere with and without blending process discussed in Section 3.5. The results are tabulated in Table 7. The results show that blending improves the accuracy of derivatives.

Table 8
Relative errors in the derivative scheme and singular quadrature for the shapes in Fig. 13c and Fig. 16c using our scheme (at the top for different grid orders m with four times upsampling to compute quadrature). The shapes are upsampled to spherical harmonics degree $p = 64$ and the error is computed relative to those values. Surface divergence is computed for the smooth function $\boldsymbol{f}(x, y, z) = (x^2, y^2, z^2)$ on the surface.

m	N	$\epsilon^{up}_{S[f]}$	$\epsilon_{\boldsymbol{n}}$	ϵ_H	ϵ_K	ϵ_{div_γ}
8	294	5×10^{-2}	9×10^{-2}	5×10^{-1}	1	9×10^{-2}
16	1350	2×10^{-2}	6×10^{-2}	2×10^{-1}	2×10^{-1}	5×10^{-2}
32	5766	5×10^{-3}	9×10^{-3}	8×10^{-2}	9×10^{-2}	9×10^{-3}
64	23814	5×10^{-4}	9×10^{-4}	8×10^{-3}	1×10^{-2}	9×10^{-4}

(a) Relative error in singular quadrature and derivatives for the shape in Fig. 13c of reduced volume $v = 0.6$.

m	N	$\epsilon^{up}_{S[f]}$	$\epsilon_{\boldsymbol{n}}$	ϵ_H	ϵ_K	ϵ_{div_γ}
8	294	8×10^{-2}	9×10^{-2}	7×10^{-1}	8×10^{-1}	9×10^{-2}
16	1350	2×10^{-2}	5×10^{-2}	2×10^{-1}	3×10^{-1}	6×10^{-2}
32	5766	3×10^{-3}	8×10^{-3}	6×10^{-2}	7×10^{-2}	9×10^{-3}
64	23814	6×10^{-4}	9×10^{-4}	1×10^{-2}	1×10^{-2}	8×10^{-4}

(b) Relative error in singular quadrature and derivatives for the shape Fig. 16c of reduced volume $v = 0.60$.

Table 9
Relative errors in the derivative scheme and singular quadrature for ellipsoids of reduced volume (a) $v = 0.92$ and (b) $v = 0.78$ using our scheme (at the top for different grid orders m with four times upsampling to compute quadrature) and using spherical harmonics scheme [36] (at the bottom for different spherical harmonics degrees p). Spherical harmonics results for $p = 64$ are used as true values and the error is computed relative to those values. Surface divergence is computed for the smooth function $\boldsymbol{f}(x, y, z) = (x^2, y^2, z^2)$ on the surface.

m	N	$\epsilon^{up}_{S[f]}$	$\epsilon_{\boldsymbol{n}}$	ϵ_H	ϵ_K	ϵ_{div_γ}
8	294	7×10^{-3}	5×10^{-3}	4×10^{-3}	6×10^{-3}	4×10^{-2}
16	1350	4×10^{-4}	2×10^{-4}	5×10^{-4}	1×10^{-3}	3×10^{-3}
32	5766	2×10^{-5}	1×10^{-5}	3×10^{-5}	6×10^{-5}	1×10^{-4}
64	23814	1×10^{-6}	7×10^{-7}	2×10^{-6}	3×10^{-6}	1×10^{-5}
p	N	$\epsilon_{S[f]}$	$\epsilon_{\boldsymbol{n}}$	ϵ_H	ϵ_K	ϵ_{div_γ}
8	144	4×10^{-5}	3×10^{-3}	2×10^{-3}	6×10^{-3}	6×10^{-3}
16	544	3×10^{-8}	1×10^{-5}	1×10^{-5}	4×10^{-5}	2×10^{-5}
24	1200	5×10^{-11}	4×10^{-8}	6×10^{-8}	2×10^{-7}	1×10^{-7}
32	2112	6×10^{-13}	5×10^{-11}	8×10^{-10}	4×10^{-9}	3×10^{-9}

(a) Relative error in singular quadrature and derivatives for an ellipsoid of reduced volume $v = 0.9$, given by $\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1$, where $a = 0.6, b = 1, c = 1$.

m	N	$\epsilon^{up}_{S[f]}$	$\epsilon_{\boldsymbol{n}}$	ϵ_H	ϵ_K	ϵ_{div_γ}
8	294	7×10^{-3}	3×10^{-2}	2×10^{-2}	4×10^{-2}	4×10^{-2}
16	1350	4×10^{-4}	9×10^{-4}	3×10^{-3}	5×10^{-3}	9×10^{-4}
32	5766	2×10^{-5}	8×10^{-5}	3×10^{-4}	5×10^{-4}	1×10^{-4}
64	23814	1×10^{-6}	4×10^{-6}	1×10^{-5}	3×10^{-5}	9×10^{-6}
p	N	$\epsilon_{S[f]}$	$\epsilon_{\boldsymbol{n}}$	ϵ_H	ϵ_K	ϵ_{div_γ}
8	144	7×10^{-4}	2×10^{-2}	3×10^{-2}	6×10^{-2}	4×10^{-2}
16	544	2×10^{-6}	6×10^{-4}	1×10^{-3}	3×10^{-3}	1×10^{-3}
24	1200	3×10^{-8}	2×10^{-5}	5×10^{-5}	1×10^{-4}	5×10^{-5}
32	2112	5×10^{-10}	1×10^{-7}	6×10^{-7}	9×10^{-6}	4×10^{-6}

(b) Relative error in singular quadrature and derivatives for an ellipsoid of reduced volume $v = 0.78$, given by $\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1$, where $a = 0.4, b = 1, c = 1$.

Table 10
Relative errors in the derivative scheme and singular quadrature for ellipsoids of reduced volume (a) $v = 0.5$ and (b) $v = 0.3$ using our scheme (at the top for different grid orders m with four times upsampling to compute quadrature) and using spherical harmonics scheme [36] (at the bottom for different spherical harmonics degrees p). Spherical harmonics results for $p = 64$ are used as true values and the error is computed relative to those values. Surface divergence is computed for the smooth function $\boldsymbol{f}(x, y, z) = (x^2, y^2, z^2)$ on the surface.

m	N	$\epsilon^{up}_{S[f]}$	$\epsilon_{\boldsymbol{n}}$	ϵ_H	ϵ_K	ϵ_{div_γ}
8	294	2×10^{-2}	2×10^{-1}	2×10^{-1}	3×10^{-1}	4×10^{-2}
16	1350	4×10^{-3}	9×10^{-3}	3×10^{-2}	6×10^{-2}	3×10^{-3}
32	5766	7×10^{-4}	8×10^{-4}	4×10^{-3}	6×10^{-3}	1×10^{-4}
64	23814	6×10^{-5}	9×10^{-5}	4×10^{-4}	7×10^{-4}	1×10^{-5}
p	N	$\epsilon_{S[f]}$	$\epsilon_{\boldsymbol{n}}$	ϵ_H	ϵ_K	ϵ_{div_γ}
8	144	9×10^{-3}	9×10^{-2}	3×10^{-1}	4×10^{-1}	2×10^{-3}
16	544	2×10^{-4}	1×10^{-2}	6×10^{-2}	9×10^{-2}	4×10^{-2}
24	1200	1×10^{-5}	3×10^{-3}	1×10^{-2}	2×10^{-2}	7×10^{-3}
32	2112	2×10^{-6}	1×10^{-4}	1×10^{-3}	5×10^{-3}	6×10^{-4}

(a) Relative error in singular quadrature and derivatives for an ellipsoid of reduced volume $v = 0.5$, given by $\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1$, where $a = 0.2, b = 1, c = 1$.

m	N	$\epsilon^{up}_{S[f]}$	$\epsilon_{\boldsymbol{n}}$	ϵ_H	ϵ_K	ϵ_{div_γ}
8	294	4×10^{-2}	4×10^{-1}	5×10^{-1}	6×10^{-1}	4×10^{-2}
16	1350	1×10^{-2}	9×10^{-2}	2×10^{-1}	3×10^{-1}	3×10^{-3}
32	5766	4×10^{-3}	2×10^{-2}	4×10^{-2}	7×10^{-2}	2×10^{-3}
64	23814	1×10^{-3}	2×10^{-3}	8×10^{-3}	9×10^{-3}	9×10^{-4}
p	N	$\epsilon_{S[f]}$	$\epsilon_{\boldsymbol{n}}$	ϵ_H	ϵ_K	ϵ_{div_γ}
8	144	4×10^{-2}	2×10^{-1}	1	1	4×10^{-1}
16	544	3×10^{-3}	1×10^{-1}	3×10^{-1}	4×10^{-1}	2×10^{-1}
24	1200	5×10^{-4}	4×10^{-2}	1×10^{-1}	1×10^{-1}	6×10^{-2}
32	2112	4×10^{-5}	5×10^{-3}	3×10^{-2}	4×10^{-2}	5×10^{-3}

(b) Relative error in singular quadrature and derivatives for an ellipsoid of reduced volume $v = 0.3$ given by $\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1$, where $a = 0.1, b = 1, c = 1$.

A.5. Surface derivative and singular integration errors for shear flow and Poiseuille flow terminal shapes

To further verify the convergence and accuracy of our derivative and integration schemes, we upsample the low reduced volume shapes obtained in Fig. 13c and Fig. 16c to $p = 64$ spherical harmonics and use the spherical harmonics derivatives and Graham-Sloan quadrature [36] as the reference values to study the convergence of our schemes. We report the relative errors for these shapes in Table 8.

A.6. Numerical errors for different reduced volume v

Here, we tabulate errors in computing the singular quadrature and surface derivatives using our numerical scheme. We report these errors for ellipsoids of varying reduced volumes $v \in \{0.92, 0.78, 0.5, 0.3\}$ in Table 9 and Table 10. The tables demonstrate that the accuracy deteriorates with decreasing reduced volumes. In general, it is harder to do long time horizon simulations of capsules with lower reduced volumes due to this reason.

Table 11
Sensitivity of singular quadrature with respect to the regularization parameter δ. First three columns show the relative errors for the patch-dependent $\delta = C\delta^*$ where C is a constant. The last three columns show the relative errors for fixed global regularization parameter $\delta = Ch_m$.

m	$\delta = 0.5\delta^*$	$\delta = \delta^*$	$\delta = 2\delta^*$	$\delta = 0.5h_m$	$\delta = h_m$	$\delta = 2h_m$
8	2×10^{-2}	7×10^{-3}	1×10^{-2}	2×10^{-2}	7×10^{-3}	1×10^{-2}
16	6×10^{-3}	4×10^{-4}	1×10^{-3}	6×10^{-3}	5×10^{-4}	5×10^{-3}
32	3×10^{-3}	2×10^{-5}	6×10^{-5}	4×10^{-3}	1×10^{-4}	1×10^{-3}
64	2×10^{-4}	1×10^{-6}	6×10^{-6}	7×10^{-4}	6×10^{-5}	6×10^{-4}

(a) Relative error in the computation of Stokes single layer potential (with density $\boldsymbol{f} = (x^2, y^2, z^2)$) on the ellipsoid $\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1$, where $a = 0.4, b = 1, c = 1$, for different choices of regularization parameter δ. Reference solution is computed using $p = 64$ spherical harmonics.

Table 12
Wall clock time per time step (denoted by T_{step} in *seconds*) for our code with 4x upsampling (on the left) vs the spherical harmonics scheme (on the right). N denotes the number of discretization points. N_{up} is the upsampled number of discretization points used for the singular quadrature in our scheme.

m	N	N_{up}	T_{step}	p	N	T_{step}
8	294	4704	0.61	8	144	0.75
16	1350	21600	1.26	16	544	1.92
32	5766	92256	10.10	32	2112	10.05
48	13254	381024	46.46	64	8320	87.86

A.7. Sensitivity of the singular quadrature scheme to the regularization parameter δ

Here, we discuss the effect of choice of regularization parameter δ on the singular quadrature accuracy. We experiment with different values of the constant in C in Equation (3.20). We tabulate the error in singular quadrature for different values of C in Table 11. We find out that $C = 1$ works best for the parameterization we have chosen. We also tabulate in the same table the results for fixed global regularization parameter $\delta = Ch_m$ and show that our patch-dependent regularization parameter gives better accuracy and convergence compared to the fixed regularization parameter.

A.8. Wall clock times for our GPU accelerated code vs *the spherical harmonics CPU code*

We report the wall clock time per time step required for our GPU-accelerated code compared to the spherical harmonics CPU code in Table 12. We note that the spherical harmonics discretization of degree p contains $2p(p+1)$ points compared to $6(m-1)^2$ for our scheme with m_{th}-order grid. Since, we use four times upsampling, the number of discretization points for evaluating singular quadrature is even higher at $N_{\text{up}} = 6(4m-1)^2$ in our scheme. Even though our discretization has much higher points than the spherical harmonics, our scheme has lower runtime for higher order grids and hence, scales better than the spherical harmonics as the number of points increase. Since the quadrature scheme requires most of the time in our simulations Fig. 17b, our scheme can be further accelerated using fast multipole methods (FMMs) [35] and can be used to do faster simulations at higher resolutions. We leave the FMM acceleration as the subject for future work.

References

[1] L. af Klinteberg, A.-K. Tornberg, A fast integral equation method for solid particles in viscous flow using quadrature by expansion, J. Comput. Phys. 326 (2016) 420–445.
[2] P. Bagchi, Mesoscale simulation of blood flow in small vessels, Biophys. J. 92 (6) (2007) 1858–1877.
[3] J. Beale, W. Ying, J. Wilson, A simple method for computing singular or nearly singular integrals on closed surfaces, Commun. Comput. Phys. 20 (3) (2016) 733–753.
[4] T. Biben, A. Farutin, C. Misbah, Three-dimensional vesicles under shear flow: numerical study of dynamics and phase diagram, Phys. Rev. E 83 (2011) 031921.
[5] G. Boedec, M. Leonetti, M. Jaeger, 3d vesicle dynamics simulations with a linearly triangulated surface, J. Comput. Phys. 230 (2011) 1020–1034.
[6] O. Bruno, L. Kunyansky, A fast, high-order algorithm for the solution of surface scattering problems: basic implementation, tests, and applications, J. Comput. Phys. 169 (2001) 80–110.
[7] D. Cordasco, A. Yazdani, P. Bagchi, Comparison of erythrocyte dynamics in shear flow under different stress-free configurations, Phys. Fluids 26 (2014) 041902.
[8] C. de Boor, A Practical Guide to Splines, Springer-Verlag, 1978.
[9] M. Delfour, J. Zolésio, Shapes and Geometries: Metrics, Analysis, Differential Calculus, and Optimization, 2011.
[10] A. Farutin, T. Biben, C. Misbah, 3D numerical simulations of vesicle and inextensible capsule dynamics, J. Comput. Phys. 275 (2014) 539.
[11] D.A. Fedosov, W. Pan, B. Caswell, G. Gompper, G.E. Karniadakis, Predicting human blood viscosity in silico, Proc. Natl. Acad. Sci. 108 (2011) 11772.
[12] D.A. Fedosov, M. Peltomäki, G. Gompper, Deformation and dynamics of red blood cells in flow through cylindrical microchannels, Soft Matter 10 (2014) 4258.
[13] E. Fehlberg, Classical fifth-, sixth-, seventh-, and eighth-order Runge-Kutta formulas with stepsize control, NASA Technical Report, 1968.

[14] E. Fehlberg, Low-order classical Runge-Kutta formulas with stepsize control and their application to some heat transfer problems, National Aeronautics and Space Administration, 1969, p. 315.
[15] Bengt Fornberg, Generation of finite difference formulas on arbitrarily spaced grids, Math. Comput. 51 (184) (1988) 699–706.
[16] Z. Gimbutas, S. Veerapaneni, A fast algorithm for spherical grid rotations and its application to singular quadrature, SIAM J. Sci. Comput. 35 (6) (2013) A2738–A2751.
[17] I. Graham, I. Sloan, Fully discrete spectral boundary integral methods for Helmholtz problems on smooth closed surfaces in $\mathbb{R}^3$, Numer. Math. 92 (2002).
[18] L. Greengard, M. O'Neil, M. Rachh, F. Vico, Fast multipole methods for the evaluation of layer potentials with locally-corrected quadratures, J. Comput. Phys. X 10 (2021) 100092.
[19] A. Guckenberger, A. Kihm, T. John, C. Wagner, S. Gekle, Numerical-experimental observation of shape bistability of red blood cells flowing in a microchannel, Soft Matter 14 (2018) 2032.
[20] T. Kruger, M. Gross, D. Raabe, F. Varnik, Predicting human blood viscosity in silico, Soft Matter 9 (2013) 9008.
[21] E. Lac, D. Barthès-Biesel, N. Pelekasis, J. Tsamopoulos, Spherical capsules in three-dimensional unbounded Stokes flows: effect of the membrane constitutive law and onset of buckling, J. Fluid Mech. 516 (2004) 303–334.
[22] R. MacMeccan, J. Clausen, G. Neitzel, C. Aidun, Simulating deformable particle suspensions using a coupled lattice-Boltzmann and finite-element method, J. Fluid Mech. 618 (2009) 13–39.
[23] J. Mauer, S. Mendez, L. Lanotte, F. Nicoud, M. Abkarian, G. Gompper, D.A. Fedosov, Flow-induced transitions of red blood cell shapes under shear, Phys. Rev. Lett. 121 (2018) 118103.
[24] H.-N. Nguyen, R. Cortez, Reduction of the regularization error of the method of regularized stokeslets for a rigid object immersed in a three dimensional Stokes flow, Commun. Comput. Phys. 15 (2014) 126–152.
[25] L. Nylons, M. Harris, J. Prins, Fast N-Body Simulation with CUDA. GPU Gems 3, Addison Wesley, 2007, pp. 677–795.
[26] S. Orszag, Fourier series on spheres, Mon. Weather Rev. 102 (1974) 56–75.
[27] C. Pozrikidis, Boundary Integral and Singularity Methods for Linearized Viscous Flow, Cambridge University Press, Cambridge, 1992.
[28] C. Pozrikidis, Effect of membrane bending stiffness on the deformation of capsules in simple shear flow, J. Fluid Mech. 440 (2001) 269–291.
[29] C. Pozrikidis, Interfacial dynamics for Stokes flow, J. Comput. Phys. 169 (2001) 250–301.
[30] A. Rahimian, S.K. Veerapaneni, D. Zorin, G. Biros, Boundary integral method for the flow of vesicles with viscosity contrast in three dimensions, J. Comput. Phys. 298 (2015) 766–786.
[31] D. Rossinelli, Y.-H. Tang, K. Lykov, D. Alexeev, M. Bernaschi, P. Hadjidoukas, M. Bisson, W. Joubert, C. Conti, G. Karniadakis, et al., The in-silico lab-on-a-chip: petascale and high-throughput simulations of microfluidics at cell resolution, in: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, 2015, pp. 1–12.
[32] R. Skalak, A. Tozeren, R.P. Zarda, S. Chien, Strain energy function of red blood cell membranes, Biophys. J. 13 (1973) 245.
[33] S. Sukumaran, U. Seifert, Influence of shear flow on vesicles near a wall: a numerical study, Phys. Rev. E 64 (2001) 011916.
[34] S. Tlupova, J. Beale, Regularized single and double layer integrals in 3D Stokes flow, J. Comput. Phys. 386 (2019) 568.
[35] A. Tornberg, L. Greengard, A fast multipole method for the three-dimensional Stokes equations, J. Comput. Phys. 227 (2008) 1613–1619.
[36] S.K. Veerapaneni, A. Rahimian, G. Biros, D. Zorin, A fast algorithm for simulating vesicle flows in three dimensions, J. Comput. Phys. 230 (2011) 5610.
[37] V. Vitkova, M.-A. Mader, B. Polack, C. Misbah, T. Podgorski, Micro-macro link in rheology of erythrocyte and vesicle suspensions?, Biophys. J. 95 (2008) L33.
[38] M. Wala, A. Klöckner, A fast algorithm for quadrature by expansion in three dimensions, J. Comput. Phys. 388 (2019) 655–689.
[39] F.W. Warner, Foundations of Differentiable Manifolds and Lie Groups, Springer-Verlag, New York, 1983.
[40] J. Weiner, On a problem of Chen, Willmore et al., Indiana Univ. Math. J. 27 (1978) 19–35.
[41] B. Wu, P.-G. Martinsson, A unified trapezoidal quadrature method for singular and hypersingular boundary integral operators on curved surfaces, SIAM J. Numer. Anal. 61 (5) (2023) 2182–2208.
[42] L. Ying, G. Biros, D. Zorin, A kernel-independent adaptive fast multipole algorithm in two and three dimensions, J. Comput. Phys. 196 (2004) 591–626.
[43] L. Ying, G. Biros, D. Zorin, A high-order 3d boundary integral equation solver for elliptic pdes in smooth domains, J. Comput. Phys. 219 (1) (2006) 247–275.
[44] H. Zhao, A. Isfahani, L. Olson, J. Freund, A spectral boundary integral method for flowing blood cells, J. Comput. Phys. 229 (10) (2010).
[45] H. Zhao, A. Spann, E. Shaqfeh, The dynamics of a vesicle in a wall-bound shear flow, Phys. Fluids 23 (2011) 121901.