Isogeometric FEM-BEM simulations of drop, capsule and vesicle dynamics in Stokes flow

Gwenn Boedec, Marc Leonetti, Marc Jaeger

Isogeometric FEM-BEM simulations of drop, capsule and vesicle dynamics in Stokes flow

Gwenn Boedec[a], Marc Leonetti[a], Marc Jaeger[b]

[a] *Aix Marseille Univ, CNRS, Centrale Marseille, IRPHE UMR 7342, 13384, Marseille, France*

[b] *Aix Marseille Univ, CNRS, Centrale Marseille, M2P2 UMR 7340, 13451, Marseille, France*

Abstract

We develop an algorithm for the three dimensional simulation of the dynamics of soft objects (drops, capsules, vesicles) under creeping flow conditions. Loop elements are used to describe the shape of the soft objects. This surface representation is used both for membrane solver based on finite element method (FEM) and for the fluid solver based on the boundary element method (BEM). This isogeometric analysis of the low Reynolds fluid-structure interaction problem is then coupled to high-order explicit time stepping or second-order implicit time stepping algorithm. For vesicles simulation, a preconditioner is designed for the resolution of the surface velocity incompressibility constraint, which is treated by the use of a local Lagrange multiplier. A mesh quality preserving algorithm is introduced to improve the control mesh quality over long simulation times. We test the proposed algorithm on capsule and vesicle dynamics in various flows, and study its convergence properties, showing a second-order convergence $O(N^{-2})$ with mesh number of elements.

Keywords: Drops, Capsules, Vesicles, Stokes flow, Boundary element method, Finite element method

Email addresses: gwenn.boedec@univ-amu.fr (Gwenn Boedec), leonetti@irphe.univ-mrs.fr (Marc Leonetti), marc.jaeger@centrale-marseille.fr (Marc Jaeger)

1. Introduction

Dynamics of small soft objects in viscous flows have been the subject of numerous studies in the past decades. One fundamental motivation is to better understand the behavior of complex fluids, down to the microscopic level of their elementary constituents: polymers, flexible fibers, drops and bubbles are the most studied soft particles. Over the two last decades however, attention has started to shift towards more complex artificial particles such as capsules and vesicles, which can be thought of as drops enclosed by a complex interface. Vesicles are drops enclosed by an incompressible fluid lipid membrane which resists bending, while capsules are drops enclosed by a solid polymerized membrane resisting shear and area deformation (and bending).

The studies of these particles were first motivated by the desire to model blood flow down to individual cells that compose it: red blood cells (RBCs), platelets, white blood cells. The common model for RBCs from a fluid dynamics point of view is to consider it as a combination of a vesicle and a capsule: an elastic network of polymers (cytoskeleton) is anchored into a lipid membrane, giving the RBCs membrane the properties of surface incompressibility, shear elasticity, as well as bending resistance [35, 24]. Thus both capsules and vesicles makes excellent candidates as biomimetic models of RBCs, allowing to focus on the influence of specific mechanical properties.

The numerical simulation of the dynamics of these particles requires a fluid solver to describe internal and external flows, a membrane solver to describe interfacial forces, as well as a proper description and tracking of the interface position to couple those solvers. Several computational models have been developed for capsules, vesicles and RBCs, and we refer to recent reviews [54, 35, 24, 2] for a complete overview of the existing literature.

For low Reynolds number flows typical of these objects, the Boundary Element Method (BEM) simplifies greatly the coupling of fluid and membrane solvers since only the interface needs to be meshed, and thus the same interface representation/mesh can be used for both solvers. However, BEM usually uses C^0 meshes while a direct computation of membrane bending forces requires at least a C^4 representation of the position, since fourth-order derivatives of the position appears in the bending forces. To circumvent this difficulty, special methods can be designed to compute membrane bending forces on C^0 meshes, using either a local reconstruction of the interface [57, 19, 39], or designing computations of Laplace-Beltrami operator

on C^0 based on discrete geometry [6, 7, 52]. On the other hand, methods based on a spectral representation have been developed [58, 53, 59, 47], but their global basis function implies that the whole interface must be refined even if the deformation is highly localized. It would thus be desirable to design a method allowing for a local representation of the interface that could be used both in fluid and membrane solver. It is possible to lift the C^4 requirement for interface representation if the forces are computed in the weak sense, using a finite element method, which only requires the approximation of the position to be in H^2. [10] have shown that subdivision surfaces meet this requirement and can be used to accurately model thin shells. Recently, schemes based on subdivision surfaces have been developed for capsules [33, 34, 29] or vesicles [22, 38, 61, 50], but [33, 34, 29] use immersed boundary method where the forces (and thus the interface representation) are spread to fluid solver nodes via approximate delta-function, while in [61, 50] bending forces are computed by first variation of the bending energy, computed from a surrogate surface based on Loop subdivision, but every other physical quantity is computed on linear basis, especially the boundary integral fluid solver. Finally, [22, 38] computes equilibrium shape of vesicles but do not consider coupled dynamics under flow.

On the other hand, the rapid development of isogeometric analysis [30] has shown that using a consistent geometry representation throughout the whole computation can greatly improve the quality of the results obtained. The idea stems from the fact that in engineering analysis CAD softwares used for designing pieces use a different surface representation like Bézier curves, splines, or Non-Uniform Rational Basis Spline (NURBS) than the software used for finite element analysis of these pieces (generally linear and quadratic Lagrange elements). Thus the standard finite element analysis starts with a modeling error in the sense that shape from CAD software cannot be represented exactly in the FEM model: before doing any computation, it is necessary to discretize the shape with a mesh which is typically based on Lagrange elements. In two dimensions, the simplest mesh would use linear triangle which will generally represent only imperfectly the real geometry of the computational domain (e.g. a disk-shape domain). Only in the limit in infinite refinement the mesh will converge to the true geometry. On the other hand, representing simple geometrical shape such as circle or ellipsoids is possible with NURBS, even with a finite number of elements. The idea behind isogeometric analysis is to use the same approximation for both the geometry discretization and the solution discretization. This leads

to a significant improvement of solution quality [30]. While this approach has been used in a boundary element context for elastostatics, aeroacoustics, electrostatics, to our knowledge, its use for viscous flow simulations has been so far limited to [28] where no deformation of the interface is involved, and [31] for drops and inextensible membranes.

In this paper, we propose an algorithm based on the idea of using Loop subdivision surfaces to represent the shape of deformable objects under flow. Starting from this assumption, a natural question is the following: how to evolve dynamically the control values of Loop elements such that the resulting evolution of the (limit) physical surface they control is a discretized version of the fluid structure interaction problem. To do so, we derive a consistent framework to discretize the problem: the boundary element method extends naturally to Loop elements to compute viscous flows, while a finite element framework can be used for membrane forces computations with a variety of physical properties of the interface. Main originality of this work is that Loop elements are used to discretize every physical quantity: position, velocity, forces, tension field. This algorithm has been already used to simulate capsule deformation in extensional flow [15] and drop dynamics in shear flow [25] with surface viscosity effects considered. Here, we describe in greater details the algorithm which was briefly outlined in [25], especially discretization of Loop boundary element as well as update of control values (including a mesh quality preserving algorithm). We also discuss extension of this algorithm to vesicle simulations which requires treatment of surface incompressibility, and introduce an implicit-time stepping algorithm. To accelerate the resolution of surface incompressibility constraint, a preconditioner based on a linear discretization is introduced. The paper is organized as follows : the system that we aim to simulate is described in section 2, the numerical description of this system is presented in section 3. The algorithm is then validated with comparison to literature results in section 4, together with a discussion of its properties.

2. Physical system description

2.1. Fluid flows

The internal flow (denoted with superscript int) and the external flow (ext) are described by the Stokes equation:

$$\begin{aligned} \eta^{\text{int,ext}} \Delta \mathbf{v}^{\text{int,ext}} - \nabla p^{\text{int,ext}} &= 0 \\ \nabla \cdot \mathbf{v}^{\text{int,ext}} &= 0 \end{aligned} \tag{1}$$

where η is the viscosity of the fluid, $\mathbf{v}$ the velocity and p the pressure.

2.2. Membrane mechanics

The membrane is a closed surface S, which is described by its position $\mathbf{x}(t)$ at time t, as well as its reference position $\mathbf{x^0}$ (surface S^0) if any. The surface density of force exerted by the membrane $\mathbf{f}$ onto surrounding fluids is given by the first variation of its surface energy

$$\mathbf{f} = \mathbf{f}(\mathbf{x}) = -\frac{1}{\sqrt{a}}\frac{\delta E}{\delta \mathbf{x}} \quad , \quad E = \int_S w_s \mathrm{d}S \tag{2}$$

where w_s is the surface energy density which completely describes the mechanical properties of the interface, and a is the determinant of the local metric.

For example, a drop with surface tension γ can be modeled with a surface energy $w_s = \gamma$, which upon application of equation (2) leads to surface density of forces

$$\mathbf{f} = \mathbf{f}^{\gamma}(\mathbf{x}) = \nabla_{\mathbf{s}}\gamma + 2\gamma H \mathbf{n} \tag{3}$$

where $\nabla_{\mathbf{s}}$ is the surface gradient operator, H is the mean curvature of the interface, and $\mathbf{n}$ is the outward pointing normal vector.

For a vesicle membrane, the most used model for surface energy is Helfrich energy, together with a local Lagrange multiplier γ for local surface incompressibility constraint ($\nabla_s \cdot \mathbf{v} = 0$):

$$w_s = \gamma + w_s^H \quad , \quad w_s^H = \frac{\kappa}{2}(2H)^2 \tag{4}$$

Note that the Helfrich model may contain a spontaneous curvature H_0 which is set to zero here. It also contains a Gaussian curvature term which can be ignored here since its integral is constant by Gauss-Bonnet theorem (no topological changes considered).

For a capsule membrane, several modeling based on thin-shell equations [27, 26, 41] have been developed [4, 5, 44]. We refer to recent reviews [45, 2] for more details and simply note that the most commonly used model for numerical simulations of capsules is a membrane model, neglecting bending resistance, and modeling the interface as a two-dimensional elastic material. Within this framework, specific interfacial properties are described by the choice of an appropriate constitutive law [3] for the membrane, for which there is two common alternatives: either a strain-softening (Neo-Hookean law, noted NH) or strain-hardening (Skalak [49] law, noted Sk). For these

two laws the surface density of energy is defined on the reference configuration S^0 as:

$$w_{s^0}^{NH} = \frac{G_s}{2}\left[I_1 - 1 + \frac{1}{I_2 + 1}\right] \quad , \quad w_{s^0}^{Sk} = \frac{G_s}{4}\left[I_1^2 + 2I_1 - 2I_2 + CI_2^2\right] \tag{5}$$

with G_s the surface shear modulus, C a parameter controlling the area dilatation resistance in Skalak law, and I_1, I_2 strain invariants related to the principal extension ratios λ_1, λ_2 by [49] :

$$\begin{aligned} I_1 &= \lambda_1^2 + \lambda_2^2 - 2 \\ I_2 &= \lambda_1^2\lambda_2^2 - 1 \end{aligned} \tag{6}$$

2.3. Coupling conditions

Membrane and fluids surrounding it are coupled by the following conditions

- *Quasi-static mechanical equilibrium:* Membrane's inertia is negligible, thus, the surface density of forces due to membrane deformation is at equilibrium with fluid stresses, leading to the condition

$$[[\boldsymbol{\sigma}]] \cdot \mathbf{n} + \mathbf{f} = 0 \tag{7}$$

  where $[[\boldsymbol{\sigma}]] = [\boldsymbol{\sigma}^{ext} - \boldsymbol{\sigma}^{int}]$ is the jump of fluid stresses at the interface and $\mathbf{f}$ is a density of force exerted by the membrane, see equation (2).

- *Continuity of the velocities at the interface:*

$$\mathbf{v}^{int}(\mathbf{x}) = \mathbf{v}^{ext}(\mathbf{x}) = \mathbf{v}^{S} \quad \forall \mathbf{x} \in S \tag{8}$$

  where $\mathbf{v^S}$ is the velocity of fluids at the interface.

- *Advection of the membrane by the flow:* The membrane is assumed to be impermeable, thus the velocity of the membrane is exactly the velocity of the fluid at the interface

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{v^S} \quad \forall \mathbf{x} \in S \tag{9}$$

- *Surface incompressibility constraint (vesicles):* The velocity field of a vesicle membrane is required to satisfy a surface divergence-free constraint:

$$\boldsymbol{\nabla}_s \cdot \mathbf{v^S} = 0 \tag{10}$$

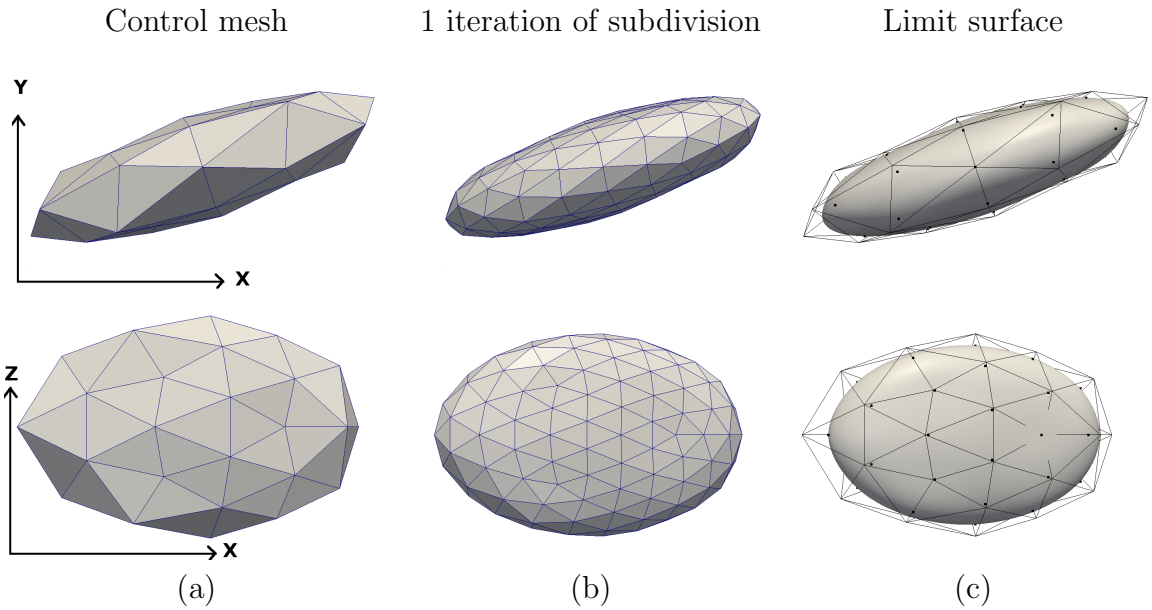
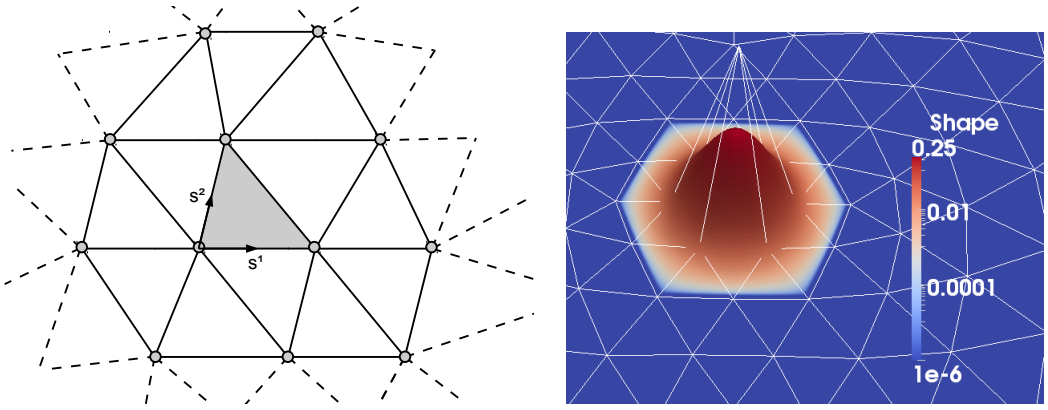


Figure 1: Simulation of a Skalak capsule in shear flow $\mathbf{v} = \dot{\gamma} y \mathbf{e_x}$, showing (a) the control mesh, composed of 80 elements, (b) the mesh obtained after one iteration of Loop subdivision and (c) the limit surface corresponding to an infinite number of iterations of Loop subdivision. The original control mesh is also represented in black wireframe, and the limit position of control vertices on the limit surface is represented by black dots. First row is the view in the xy plane, and second row the view in the xz plane.

3. Numerical discretization

3.1. Surface representation : Subdivision surfaces

The surface is discretized using Loop subdivision elements [37], which are assembly of linear triangles refined using a subdivision process. Starting from an initial coarse mesh, called control mesh, new elements are created and vertices positions are updated according to Loop subdivision rules [37]. In the limit of infinite subdivisions, the mesh converges toward a limit surface, which is C^2 continuous, except at some extraordinary vertices where it is C^1. Because of the regularity of the limit surface, numerical computation of geometrical quantities such as normal vector or curvature tensors is more straightforward and accurate that when using classical linear or quadratic Lagrange elements which are only C^0. The ordinary or regular vertices are linked to six elements, while any vertex linked to a different number of element is termed "extraordinary" since the subdivision rule must be adapted



(a) (b)

Figure 2: (a) Scheme of an element (gray triangle) with its local parametrization. The position inside the element depends on the nodal values of position of every node connected to the central element by at most one edge, denoted the one-ring. (b) Support of the shape function associated with one node extends up to every triangle which is at most one edge afar from the node: the control mesh is composed of a flat mesh, except one node whose height is set to 1.

here. The number of extraordinary vertices depends on the initial mesh (level 0 of subdivision). Starting from an icosahedron, there are 12 vertices linked to only 5 elements. This number does not change with refinement of mesh, since any vertex introduced by Loop subdivision is regular. Note also that it is not possible to discretize a spherical surface without introducing irregular vertices : the reason behind this is that the Euler characteristic of a sphere is constant and can related to discretization parameters (number of faces, vertices, edges) by Euler's formula. An example of a simulation of a capsule under shear flow using such elements is presented in figure 1.

An interesting property of Loop subdivision surface is that it is possible to analytically compute the limit position of the original vertices composing the control mesh. Moreover, since the seminal work of [51], it is also possible to compute the limit position of any point inside the initial triangles by using shape functions N_i:

$$\mathbf{x^e}(s^1, s^2) = \sum_{p \in \text{one-ring}} N_p(s^1, s^2)\mathbf{X^p} \tag{11}$$

where (s^1, s^2) is a local parametrization of the element. The shape functions are polynomials of the local parametrization (see [51]), and $\mathbf{X^p}$ are the

nodal values of the position field. Note that these values have no real physical meaning : they are neither the real position of the vertex, nor the limit position of this vertex. They are just numerical values controlling the approximation of the shape. Contrary to classical Lagrange triangular elements, the support of the shape function for a given node p extends outside the triangles containing p as a vertex, up to every triangle which is linked to p by at most an edge (see figure 2 (b)). Thus, the position inside a given triangle depends not only on nodal values of the vertices of this triangle, but also of nodal values $\mathbf{X^p}$ of the nodes related to this triangle by at most one edge. These elements are called the one-ring of element e. We refer to Stam [51] and Cirak et al. [10] for a detailed discussion of the evaluation of the shape functions inside an element. In this work, this local basis $(N_1, N_2, ...)$ is used to represent every function f which is defined on the interface, e.g. membrane force density or membrane velocity. Thus, f is written as:

$$f(\mathbf{x}) = f(\mathbf{x}(s^1, s^2)) = f(s^1, s^2) = \sum_{p \in \text{one-ring}} F^p N_p(s^1, s^2) \tag{12}$$

where F^p is the p-th nodal value of the field representing the function f.The series on the right-hand side of equation (12) is an approximation of the function f using Loop subdivision functions. While equation (12) gives a way to compute f at any point of the membrane if the nodal values F^p are known, one also needs to convert a field defined onto the interface $f(x)$ into its nodal values F^p , that is, given f find the approximation under the form (12) such that the approximation error is minimized. Two main alternatives could be considered : Galerkin formulation in which the approximation error is measured in the L_2-norm over the surface, or a collocation formulation in which the approximation error is measured pointwise. In this work, we choose to collocate the known field at vertices. This choice will be discussed in section 3.2 when considering velocity evaluation. Note that the two formulations were compared in the context of drop isogeometric simulations by [31], which found that while the Galerkin formulation has better convergence properties, collocation is both more stable and more fast. Thus, assuming f is known at each point of the interface, we search the nodal values F^p of a field $\hat{f}$ such that:

$$f^p = f(\mathbf{x} = \mathbf{x^p}) = \hat{f}(\mathbf{x^p}) = \sum_{q \in \text{one-ring}} F^q N_q(s^1(\mathbf{x^p}), s^2(\mathbf{x^p})) \quad \forall p \in \{1, ..., N_{nodes}\} \tag{13}$$

where $\mathbf{x^p}$ is the position of the p-th control vertex on the limit surface, and $N_q(\mathbf{x^p})$ are the shape functions evaluated at parameters corresponding to $\mathbf{x^p}$. Assembling all the nodal values of the field $\hat{f}$ into a global vector $\{F^p\}$, equation (13) is rewritten in matrix form as:

$$\{f^p\} = \left\{\hat{f}(\mathbf{x^p})\right\} = \mathbf{C}\,\{F^p\} \tag{14}$$

where $\mathbf{C}$ is a collocation matrix in the sense that it enforces pointwise the relationship between the known limit values f^p and the unknown nodal values F^p of the field $\hat{f}$ which interpolates f at positions $\mathbf{x^p}$. This matrix is sparse and thus the linear system is solved with the sparse solver UMFPACK [13, 14]. This linear system is solved each time a conversion from limit value to nodal values is needed. However, for system sizes considered here, the computational cost is always a small fraction (typically, less than 10%) of the total cost of one time step. In the following, the notation $\{f\}$ will be used to designate the N-dimensional vector containing the values of f evaluated at control vertices limit positions $\{\mathbf{x}\}$ while the notation $\{F\}$ will designate the N-dimensional vector containing the nodal values defined on the control mesh.

3.2. Fluid solver : Boundary element method

For a soft liquid particle suspended in a viscous fluid, the velocity at a point $\mathbf{y}$ of the particle interface S is given by the integral relation [43]:

$$\begin{aligned}\mathbf{v}(\mathbf{y}) = \frac{2}{1+\lambda}\mathbf{v}^{\infty}(\mathbf{y}) + \frac{1}{4\pi\eta_{ext}(1+\lambda)}\int_S G_{ij}(\mathbf{x},\mathbf{y})f_i(\mathbf{x})\mathrm{d}S(\mathbf{x})\mathbf{e_j} \\ + \frac{(1-\lambda)}{4\pi(1+\lambda)}\int_S v_i(\mathbf{x})T_{ijk}(\mathbf{x},\mathbf{y})n_k(\mathbf{x})\mathrm{d}S(\mathbf{x})\mathbf{e_j}\end{aligned} \tag{15}$$

where $\mathbf{v}^{\infty}$ is the imposed far-field velocity, $\boldsymbol{G}, \boldsymbol{T}$ are the free space Stokeslet and Stresslet, $\mathbf{f}$ is the surface density of force due to the membrane, $\mathbf{e_i}$ is the unit i-th Cartesian vector and $\lambda = \eta_{int}/\eta_{ext}$ is the viscosity ratio. Roman indices (i, j, k, l) can vary between 1 and 3 and denote the Cartesian components of vectors and tensors. To lighten the expressions, Einstein summation convention is used e.g. $G_{ij}f_i = G_{1j}f_1 + G_{2j}f_2 + G_{3j}f_3$.

The integrals in (15) needs special numerical treatment to deal with singularities of $\boldsymbol{G}, \boldsymbol{T}$ occurring when $\mathbf{x} \to \mathbf{y}$. A powerful method to treat such integrals consists in subtracting the singularities by using analytical integral identities [43]. This has been largely used for Stresslet as well as

Stokeslet when the membrane force was purely normal (as is the case for a clean drop). However, more general expression for the forces (with tangential components) required different treatment like singularity cancellation by special quadratures. Only recently was derived a new integral identity allowing to treat general expression of membranes forces [19] without resorting to special integration rule for singular elements. Using both singularity subtraction for Stresslet [43] and Stokeslet [19], the boundary integral relation is rewritten as:

$$\begin{aligned}\mathbf{v}(\mathbf{y}) = \mathbf{v}^{\infty}(\mathbf{y}) &+ \frac{1}{8\pi\eta_{ext}} \int_S G_{ij}(\mathbf{y},\mathbf{x}) \tilde{f}_j(\mathbf{y},\mathbf{x}) \mathrm{d}S(\mathbf{x}) \mathbf{e_i} \\ &+ \frac{1}{4\pi} \left(\int_S \mathbf{R}(\mathbf{y},\mathbf{x}) \mathrm{d}S(\mathbf{x}) \right) \wedge [\mathbf{n}(\mathbf{y}) \wedge \mathbf{f}(\mathbf{y})] \\ &+ \frac{(1-\lambda)}{8\pi} \int_S [v_i(\mathbf{x}) - v_i(\mathbf{y})] T_{ijk}(\mathbf{x},\mathbf{y}) n_k(\mathbf{x}) \mathrm{d}S(\mathbf{x}) \mathbf{e_i}\end{aligned} \tag{16}$$

where the expression for the modified force density $\tilde{\mathbf{f}}$ and $\mathbf{R}$ are [19]:

$$\begin{aligned}\tilde{\mathbf{f}}(\mathbf{y},\mathbf{x}) &= \mathbf{f}(\mathbf{x}) + [-\mathbf{n}(\mathbf{x})(\mathbf{f}(\mathbf{y}) \cdot \mathbf{n}(\mathbf{y})) + \mathbf{n}(\mathbf{x}) \wedge (\mathbf{n}(\mathbf{y}) \wedge \mathbf{f}(\mathbf{y}))] \\ \mathbf{R}(\mathbf{y},\mathbf{x}) &= \frac{[(\mathbf{y}-\mathbf{x}) \cdot \mathbf{n}(\mathbf{x})]}{|\mathbf{y}-\mathbf{x}|^3} (\mathbf{y}-\mathbf{x})\end{aligned} \tag{17}$$

In the expression (16) every integrands is now non singular and can thus be integrated by standard quadratures. In this work, Gauss points are used. Note that while there is an additional integral in equation (17) and thus an additional computational cost when compared to classical formulation, this doesn't change the overall $O(N^2)$ computational complexity of the algorithm, only the prefactor. On the other hand, since all the integrals can be treated with standard quadratures without distinction between singular and regular elements, it eases the parallelization of the computation, when compared to using special quadratures on singular elements (e.g. Duffy quadrature or introducing polar coordinates).

If the internal and external viscosities are equal ($\lambda = 1$), the integral relation (16) is simplified and directly gives the velocity if the membrane forces are known. If $\lambda \neq 1$, the integral relation (16) needs to be solved for the velocity. This equation is enforced at each node of the mesh. Equation (16) is a linear integral relation between physical velocities $\mathbf{v}$ and forces $\mathbf{f}$, which depends linearly on the nodal values of velocities $\mathbf{V}$ and forces $\mathbf{F}$ via interpolation (12). Equation (16) is thus rewritten as :

$$\left[\mathbf{I} + (\lambda - 1)\mathbf{T}\mathbf{C}^{-\mathbf{1}}\right] \{v\} = \{v^{\infty}\} + \mathbf{G}\{F\} \tag{18}$$

where $\mathbf{I}$ is the identity matrix, $\mathbf{G}, \mathbf{T}$ are the matrix corresponding to the discretization of (16) and $\{v\}, \{v^\infty\}$ are vectors containing the limit values of velocities, and $\{F\}$ contains the nodal values of forces. This linear system is solved by GMRES [48] without explicitly computing the $\mathbf{G}, \mathbf{T}$ matrices, but only the matrix-vector products. These matrix-vectors products corresponds to linear integral operators e.g.:

$$\int_S G_{ij}(\mathbf{y},\mathbf{x})\tilde{f}_j(\mathbf{y},\mathbf{x})\mathrm{d}S(\mathbf{x}) = \sum_{e=1}^{N_{el}} \int_{S^e} G_{ij}(\mathbf{y},\mathbf{x})\tilde{f}_j(\mathbf{y},\mathbf{x})\mathrm{d}S^e(\mathbf{x}) \tag{19}$$

where S^e is the surface of element e. Integrals are then transformed in the parameter space (ξ, η) and then performed by numerical quadrature, using Gauss-Hammer quadrature points on triangular elements:

$$\begin{aligned}\int_{S^e} G_{ij}(\mathbf{y},\mathbf{x})\tilde{f}_j(\mathbf{y},\mathbf{x})\mathrm{d}S^e(\mathbf{x}) &= \int_{s^1=0}^{1}\int_{s^2=0}^{1-s^1} G_{ij}(\mathbf{y},\mathbf{x})\tilde{f}_j(\mathbf{y},\mathbf{x})\sqrt{a}\mathrm{d}s^1\mathrm{d}s^2 \\ &= \sum_{k=1}^{N^{quad}} w_k G_{ij}(\mathbf{y},\mathbf{x}_k)\tilde{f}_j(\mathbf{y},\mathbf{x_k})\sqrt{a(s_k^1,s_k^2)}\end{aligned} \tag{20}$$

with $\mathbf{x}_k = \mathbf{x}(s_k^1, s_k^2)$ the position of k-th quadrature point. In this work, 12 Gauss quadrature points per element are used for the computation of boundary element integrals.

After obtaining of limit values of velocities, the resulting values of velocities on the limit surface are converted into nodal values of the velocity by resolution of the collocation equation (14). Note that a Galerkin formulation of equation (16) would require an additional integration over the surface, meaning that the boundary integral equation would need to be evaluated at each quadrature point instead of mesh vertices. Since there is always more elements than vertices in the mesh, even using a one-point quadrature rule would require more integral evaluations than in the collocation case, and the gap only widens if one uses higher-order quadrature rules. Thus, in terms of computational complexity, it seems advantageous to use a collocation formulation.

3.3. Membrane solver : Finite element method

Starting from a reference shape $\mathbf{x^0}(s^1, s^2)$, the membrane is deformed into its current shape $\mathbf{x}(s^1, s^2)$. To describe elastic deformations, it is useful to introduce a local basis on the deformed configuration composed of

the tangent vectors $\mathbf{a_1}, \mathbf{a_2}$ defined by the parametrization and the outward pointing normal vector $\mathbf{n}$:

$$\mathbf{a_1} = \mathbf{x}_{,\mathbf{1}} = \frac{\partial \mathbf{x}}{\partial s^1} \quad , \quad \mathbf{a_2} = \mathbf{x}_{,\mathbf{2}} = \frac{\partial \mathbf{x}}{\partial s^2} \quad , \quad \mathbf{n} = \frac{\mathbf{a_1} \wedge \mathbf{a_2}}{|\mathbf{a_1} \wedge \mathbf{a_2}|} \tag{21}$$

The local basis on reference configuration $(\mathbf{a_1^0}, \mathbf{a_2^0}, \mathbf{n}^0)$ is defined similarly, with $\mathbf{x^0}$ replacing $\mathbf{x}$. In the following, Greek indices $(\alpha, \beta, \gamma, \delta)$ will be used to denote components of vectors and tensors in the local basis, and can take the value 1 or 2. With this notation, the deformed configuration metric $a_{\alpha\beta}$ and curvature $b_{\alpha\beta}$ tensors are defined by :

$$a_{\alpha\beta} = \mathbf{a}_\alpha \cdot \mathbf{a}_\beta \quad , \quad b_{\alpha\beta} = \mathbf{a}_{\alpha,\beta} \cdot \mathbf{n} = \frac{\partial \mathbf{a}_\alpha}{\partial s^\beta} \cdot \mathbf{n} \tag{22}$$

with a similar definition for reference configuration metric $a^0_{\alpha\beta}$ and curvature $b^0_{\alpha\beta}$. The determinant of the metric is noted $a = \det(a_{\alpha\beta})$ and is related to the elementary area by $\mathrm{d}S = \sqrt{a}\mathrm{d}s^1\mathrm{d}s^2$. Finally, it is useful to introduce the inverse metric $a^{\alpha\beta}$ defined by $a^{\alpha\gamma}a_{\gamma\beta} = \delta^\alpha_\beta$ with δ^α_β the Kronecker symbol.

Following [10], we write the principle of virtual work for the membrane as :

$$-\int_S \left[\sigma^{\alpha\beta}\delta(E_{\alpha\beta}) + \mu^{\alpha\beta}\delta(B_{\alpha\beta})\right] \mathrm{d}S + \int_S \mathbf{f^{ext}} \cdot \delta\mathbf{x}\mathrm{d}S = 0 \tag{23}$$

The first integral is the internal virtual work for a virtual displacement $\delta\mathbf{x}$, where $\sigma^{\alpha\beta}, \mu^{\alpha\beta}$ are the effective membrane and bending stresses. The second integral is the external virtual work. $E_{\alpha\beta}$ is the Green-Lagrange strain tensor measuring membrane deformations

$$E_{\alpha\beta} = \frac{1}{2}\left(\mathbf{a}_\alpha \cdot \mathbf{a}_\beta - \mathbf{a^0_\alpha} \cdot \mathbf{a^0_\beta}\right) = \frac{1}{2}\left(a_{\alpha\beta} - a^0_{\alpha\beta}\right) \tag{24}$$

and $B_{\alpha\beta}$ is a measure of bending strains defined as

$$B_{\alpha\beta} = \left(\mathbf{a}_{\alpha,\beta} \cdot \mathbf{n} - \mathbf{a^0_{\alpha,\beta}} \cdot \mathbf{n^0}\right) = \left(b_{\alpha\beta} - b^0_{\alpha\beta}\right) \tag{25}$$

Noting that $\delta(E_{\alpha\beta}) = \frac{1}{2}\delta(a_{\alpha\beta})$, $\delta(B_{\alpha\beta}) = \delta(b_{\alpha\beta})$ and $\mathbf{f^{ext}} = -\mathbf{f}$, equation (23) writes:

$$\int_S \left[\frac{1}{2}\sigma^{\alpha\beta}\delta(a_{\alpha\beta}) + \mu^{\alpha\beta}\delta(b_{\alpha\beta})\right] \mathrm{d}S + \int_S \mathbf{f} \cdot \delta\mathbf{x}\mathrm{d}S = 0 \tag{26}$$

The above equality holds whether it is a drop, vesicle or capsule (with Kirchoff-Love hypothesis), and thus constitutes an interesting unified framework to study both types of objects. Before describing the specific interfacial properties that comes into play via membrane ($\sigma^{\alpha\beta}$) and bending stresses ($\mu^{\alpha\beta}$), we describe how this equality is discretized. From (21) and (22) we have

$$\begin{aligned}
\delta(a_{\alpha\beta}) &= [\delta\mathbf{a}_\alpha \cdot \mathbf{a}_\beta + \mathbf{a}_\alpha \cdot \delta\mathbf{a}_\beta] = [\delta\mathbf{x}_{,\alpha} \cdot \mathbf{x}_{,\beta} + \mathbf{x}_{,\alpha} \cdot \delta\mathbf{x}_{,\beta}] \\
\delta(b_{\alpha\beta}) &= \mathbf{n} \cdot \delta(\mathbf{a}_{\alpha,\beta}) + \mathbf{a}_{\alpha,\beta} \cdot \delta(\mathbf{n}) \\
&= \mathbf{n} \cdot \delta(\mathbf{a}_{\alpha,\beta}) + \left[b_{\alpha\beta}\mathbf{n} + \Gamma^\gamma_{\alpha\beta}\mathbf{a}_\gamma \cdot\right] \delta(\mathbf{n}) \\
&= \mathbf{n} \cdot \delta(\mathbf{a}_{\alpha,\beta}) - \Gamma^\gamma_{\alpha\beta}\mathbf{n} \cdot \delta(\mathbf{a}_\gamma)
\end{aligned} \tag{27}$$

where $\Gamma^\gamma_{\alpha\beta} = a^{\gamma\delta}\mathbf{a}_{\alpha,\beta} \cdot \mathbf{a}_\delta$ are the Christoffel symbols. We used the identities $\delta(||\mathbf{n}||^2) = 2\mathbf{n} \cdot \delta\mathbf{n} = 0$ and $\delta(\mathbf{a}_\gamma \cdot \mathbf{n}) = \delta(\mathbf{a}_\gamma) \cdot \mathbf{n} + \mathbf{a}_\gamma \cdot \delta(\mathbf{n}) = 0$ to express $\delta(b_{\alpha\beta})$ as a function of $\delta\mathbf{x}$.

Using the same basis functions (Loop elements) for the Cartesian components of membrane force, position and virtual displacement, we write:

$$\begin{aligned}
f_i(s^1, s^2) &= \sum_{p \in \text{one-ring}} F_i^p N_p(s^1, s^2) \\
x_i(s^1, s^2) &= \sum_{p \in \text{one-ring}} X_i^p N_p(s^1, s^2) \\
\delta x_i(s^1, s^2) &= \sum_{p \in \text{one-ring}} \delta X_i^p N_p(s^1, s^2)
\end{aligned} \tag{28}$$

The principle of virtual work (26) is thus discretized as:

$$\sum_{e=1}^{N_{el}} \int_{S^e} [I_\sigma + I_M + I_\mathbf{f}] \sqrt{a} \mathrm{d}s^1 \mathrm{d}s^2 = 0 \tag{29}$$

with

$$\begin{aligned}
I_\sigma &= \sum_{p \in \text{one-ring}} \frac{1}{2}\sigma^{\alpha\beta} \left(N_{p,\alpha}x_{i,\beta} + N_{p,\beta}x_{i,\alpha}\right) \delta X_i^p \\
I_M &= \sum_{p \in \text{one-ring}} \mu^{\alpha\beta} \left(n_i N_{p,\alpha\beta} - \Gamma^\gamma_{\alpha\beta} n_i N_{p,\gamma}\right) \delta X_i^p \\
I_\mathbf{f} &= \sum_{p \in \text{one-ring}} \sum_{q \in \text{one-ring}} N_p N_q F_i^q \delta X_i^p
\end{aligned} \tag{30}$$

where $N_{p,\alpha} = \frac{\partial N_p}{\partial s^\alpha}$. Note that a double sum appears explicitly in $I_{\mathbf{f}}$ because nodal values F are unknowns, while a double sum is "hidden" in I_σ, I_M because nodal values of positions are known and thus can be used to evaluate every terms inside the triangle (see discussion below).

Equation (29) can then be written in matrix vector form as

$$\{\mathbf{RHS}\} + \mathbf{M}\{\mathbf{F}\} = \mathbf{0} \tag{31}$$

The right hand side vector $\{\mathbf{RHS}\}$ and the mass matrix $\mathbf{M}$ are formed by numerical integration of equation (29) using Gauss quadrature points, for instance, for membrane terms :

$$\int_{S^e} I_\sigma \sqrt{a} \mathrm{d}s^1 \mathrm{d}s^2 \approx \sum_{k=1}^{N^{quad}} w_k I_\sigma(s_k^1, s_k^2) \sqrt{a(s_k^1, s_k^2)} \tag{32}$$

where w_k and (s_k^1, s_k^2) are weights and parameters values of the k-th quadrature point, N^{quad} is the total number of quadrature points, and the dependency of I_σ and $\sqrt{a}$ on parameters has been made explicit for clarity. To evaluate I_σ and $\sqrt{a}$ at parameters values (s_k^1, s_k^2) one needs to evaluate shape functions and their derivatives, membrane stresses and moments, and also tangent and normal vectors. In this work, 12 Gauss quadrature points are used.

The evaluation of shape functions and their derivatives inside regular and irregular elements follows the strategy of [10]. The evaluation of the membrane stresses and moments and of virtual variations of membrane and bending strains is discussed in the following.

For drops and zero-thickness capsule model (for example, Neo-Hookean or Skalak law), the surface energy does not depend on curvatures, thus

$$\mu^{\alpha\beta} = 0 \tag{33}$$

It then remains to compute $\boldsymbol{\sigma}$, which for drops takes the form

$$\sigma^{\alpha\beta} = \gamma a^{\alpha\beta} \tag{34}$$

while for capsules it takes the form [55]:

$$\sigma^{\alpha\beta} = \frac{2}{J_s} \frac{\partial w_s}{\partial I_1} a^{0,\alpha\beta} + 2J_s \frac{\partial w_s}{\partial I_2} a^{\alpha\beta} \tag{35}$$

with $J_s = \frac{\sqrt{a}}{\sqrt{a^0}}$ the Jacobian of the transformation from reference to deformed position. Derivatives of the energy w_s with respect to invariants is done analytically, and the invariants of the deformation are computed as:

$$\begin{aligned} I_1 &= a_{\alpha\beta}a^{0,\alpha\beta} - 2 \\ I_2 &= a/a^0 - 1 \end{aligned} \tag{36}$$

For a membrane obeying the Helfrich energy (4), the membrane and bending stresses write [8]:

$$\begin{aligned} \sigma^{\alpha\beta} &= \frac{2}{\sqrt{a}}\frac{\partial(\sqrt{a}w_s^H)}{\partial a_{\alpha\beta}} = \frac{\kappa}{2}\left[4H^2a^{\alpha\beta} - 8Hb^{\alpha\beta}\right] + \gamma a^{\alpha\beta} \\ \mu^{\alpha\beta} &= \frac{\partial w_s^H}{\partial b_{\alpha\beta}} = \frac{\kappa}{2}\left[4Ha^{\alpha\beta}\right] \end{aligned} \tag{37}$$

In all cases, one needs to compute inverse metrics and curvature tensors and local coefficients ($\gamma, H, J_s, \frac{\partial w_s}{\partial I_1}, \frac{\partial w_s}{\partial I_2}$) at quadrature points.

The metric tensors in the reference state $a^0_{\alpha\beta}$ and the deformed state $a_{\alpha\beta}$ are readily computed from the interpolation of the position, e.g.:

$$a_{\alpha\beta}(s^1, s^2) = \mathbf{x}_{,\alpha} \cdot \mathbf{x}_{,\beta} = \sum_{p\in\text{one-ring}} \sum_{q\in\text{one-ring}} X_i^p X_i^q N_{p,\alpha} N_{q,\beta} \tag{38}$$

The inverses tensors ($a^{0,\alpha\beta}, a^{\alpha\beta}$) are computed explicitly at each quadrature point.

Since the Loop subdivision scheme guarantees C^2 continuity everywhere except at extraordinary vertices, it is possible to evaluate the curvature tensor at any quadrature point by direct differentiation of the shape functions, to yield:

$$b_{\alpha\beta}(s^1, s^2) = \mathbf{x}_{,\alpha\beta} \cdot \mathbf{n} = \left(\sum_p X_i^p N_{p,\alpha\beta}\mathbf{e_i}\right) \cdot \mathbf{n} \tag{39}$$

where the normal vector is computed by

$$\mathbf{n}(s^1, s^2) = \frac{\mathbf{a_1} \wedge \mathbf{a_2}}{|\mathbf{a_1} \wedge \mathbf{a_2}|} = \frac{\left(\sum_p X_i^p \frac{\partial N_p}{\partial s^1}\mathbf{e_i}\right) \wedge \left(\sum_q X_j^q \frac{\partial N_q}{\partial s^2}\mathbf{e_j}\right)}{|\mathbf{a_1} \wedge \mathbf{a_2}|} \tag{40}$$

For drops and vesicles, the local value of the tension (or Lagrange multiplier) is computed from the interpolation of the tension field:

$$\gamma(s^1, s^2) = \sum_{p\in\text{one-ring}} \Gamma^p N_p(s^1, s^2) \tag{41}$$

For a drop, the surface tension is a physical property of the interface, which may depends on other variables such as temperature or surfactant concentration. For a vesicle, the tension is a Lagrange multiplier ensuring the constraint of surface divergence free velocity field, and is thus an additional unknown of the system. The determination of tension in this case is discussed in the following section.

3.4. Surface incompressibility constraint solver

For a vesicle, the surface incompressibility constraint is satisfied by the use of a local Lagrange multiplier γ, which is the solution of :

$$\boldsymbol{\nabla}_s \cdot \mathbf{v}^\gamma(\mathbf{x}) + \boldsymbol{\nabla}_s \cdot \mathbf{v}^\infty(\mathbf{x}) + \boldsymbol{\nabla}_s \cdot \mathbf{v}^\kappa(\mathbf{x}) = 0 \tag{42}$$

where $\mathbf{v}^\gamma$ is the velocity due solely to tension, and $\mathbf{v}^\infty, \mathbf{v}^\kappa$ are the imposed external velocity and bending induced velocity. This equation is actually linear in the unknown tension field γ, and is thus rewritten as:

$$\mathbf{D}^\gamma\{\Gamma\} = -\{\boldsymbol{\nabla}_s \cdot \mathbf{v}^\infty(\mathbf{x}) + \boldsymbol{\nabla}_s \cdot \mathbf{v}^\kappa(\mathbf{x})\} \tag{43}$$

where $\mathbf{D}^\gamma$ is a linear operator transforming the nodal values of tension $\{\Gamma\}$ to the limit value of the surface divergence of the velocity field due to tension. Equation (43) is thus a collocation at nodes of surface incompressibility constraint (10). Equation (43) is solved by GMRES [48]. The operator $\mathbf{D}^\gamma$ is thus never build explicitly, only its action on $\{\Gamma\}$ is computed as follows. Given a particular vector $\{\Gamma\}$, the nodal values of the forces due to tension are solutions of the system (31), with the right-hand side computed with equation (34). The limit values of velocity field $\{v\}$ are then obtained by application of the BEM equation (18) without $\mathbf{v}^\infty$. After resolution of the system (18), the nodal values $\{V\}$ of the velocity field are obtained by application of the collocation equation (14). Finally, the limit value of surface divergence is obtained by using the following formula

$$\nabla_\mathbf{s} \cdot \mathbf{v} = a^{\alpha\beta} \frac{\partial x_i}{\partial s^\alpha} \frac{\partial v_i}{\partial s^\beta} \tag{44}$$

This equation is computed at each node. For regular vertices, the application is straightforward by injection of the interpolation of position and velocities in (44). For irregular vertices, first derivatives are computed by using the tangent masks of Loop subdivision scheme [10]. The inverse metric is obtained by analytic inversion of metric.

Note that equation (43) is however badly conditioned and leads to an increasing number of GMRES iteration while decreasing the mesh size, as can be seen in table 1. To overcome this problem, a preconditioner has been designed. The idea is to use a coarse approximation of the operator $\mathbf{D}^\gamma$ by approximating tension forces, velocity and surface divergence computation onto a C^0 mesh composed of linear triangles. The construction of these operators is based on our previous work using linear triangles for vesicle simulations [7] and is briefly recalled here for completeness. The basic idea is to interpolate linearly fields such as tension and velocity on the triangles describing the shape

$$\gamma(s^1, s^2) = \gamma^0(1 - s^1 - s^2) + \gamma^1 s^1 + \gamma^2 s^2 \tag{45}$$

where (s^1, s^2) are local element coordinates and γ^i is the nodal value at node i. Then, it is possible to compute gradients for tension or divergence for velocity with respect to local element coordinates:

$$\nabla_{\mathbf{s}}\gamma = \frac{\partial \gamma}{\partial s^\alpha}\mathbf{a}^\alpha = (\gamma^1 - \gamma^0)\mathbf{a^1} + (\gamma^2 - \gamma^0)\mathbf{a^2} \tag{46}$$

where $\mathbf{a}^\alpha = a^{\alpha\beta}\mathbf{a}_\beta$. Equation (46) gives a constant value by element, which is then converted into a nodal value by a weighted-area average, and then projected on tangent plane:

$$(\nabla_{\mathbf{s}}\gamma)_n = [\mathbf{I} - \mathbf{n} \otimes \mathbf{n}] \cdot \frac{1}{A} \sum_{e \in E^n} A_e \left[(\gamma^1 - \gamma^0)\mathbf{a^1} + (\gamma^2 - \gamma^0)\mathbf{a^2}\right] \tag{47}$$

Similar construction is applied for the computation of surface divergence of the velocity field, and the normal component of the tension force is obtained by computing mean curvature at nodes using a discrete surface divergence theorem (see [7] for details). It is then straightforward to construct matrix form of the relations between on the one hand limit values of tension ($\{\gamma\}$) and limit values of tension forces ($\{\mathbf{f}^\gamma\}$), and on the other hand, between limit values of velocities ($\{v\}$) and limit values of surface divergence ($\{\boldsymbol{\nabla}_s \cdot \mathbf{v}\}$). To finish the approximation of $\mathbf{D}^\gamma$, the linear relation between limit values of forces ($\{f\}$) and limit values of velocities ($\{v\}$) expressed by (16) without viscosity contrast ($\lambda = 1$) is discretized over the linear mesh as in [7] by using Gauss-Hammer quadrature for non-singular elements and Gauss-Legendre quadrature with polar coordinates for singular elements. Thus, with the matrices $\mathbf{F}^\gamma, \mathbf{D}, \mathbf{G}$ defined by

$$\{\mathbf{f}^\gamma\} = \mathbf{F}^\gamma\{\gamma\} \quad , \quad \{\boldsymbol{\nabla}_s \cdot \mathbf{v}\} = \mathbf{D}\{v\} \quad , \quad \{v\} = \mathbf{G}\{f\} \tag{48}$$

	No preconditioner					preconditioner				
N_{el}	$\lambda = 1$			$Ca = 1$		$\lambda = 1$			$Ca = 1$	
	Ca			λ		Ca			λ	
	0.1	1	10	0.1	10	0.1	1	10	0.1	10
320	18	18	18	18	18	10	12	12	13	13
1280	39	42	44	46	43	13	15	16	17	16
5120	68	76	78	81	77	14	17	18	19	13

Table 1: Tension solver preconditioning : number of GMRES iterations needed to ensure a $\epsilon = 10^{-9}$ residual for the surface divergence free constraint (equation 43), for a vesicle under shear flow $\mathbf{v} = \dot{\gamma} x \mathbf{e_y}$, under various flow conditions measured by capillary number $Ca = \frac{\dot{\gamma}\eta_{ext}R^3}{\kappa}$ and viscosity contrast $\lambda = \frac{\eta_{int}}{\eta_{ext}}$.

the preconditioner $\widehat{\mathbf{D}^\gamma}$ is defined as

$$\widehat{\mathbf{D}^\gamma} = \mathbf{D}\mathbf{C^{-1}}\mathbf{G}\mathbf{F}^\gamma \tag{49}$$

and we solve:

$$\mathbf{D}^\gamma \left(\widehat{\mathbf{D}^\gamma}\right)^{-1} \{y\} = -\{\boldsymbol{\nabla}_s \cdot \mathbf{v}^\infty + \boldsymbol{\nabla}_s \cdot \mathbf{v}^\kappa\} \quad , \quad \widehat{\mathbf{D}^\gamma}\{\Gamma\} = \{y\} \tag{50}$$

with $\{y\}$ an intermediate variable of calculus. In theory, the C^{-1} is not necessary since for preconditioning, every variable is interpolated on the linear mesh. However, numerical tests have shown that including this matrix in the definition of $\widehat{\mathbf{D}^\gamma}$ lowered the number of iteration. While we lack a rigorous explanation for this, we propose the following heuristic argument: Note that on a linear mesh, the value of the surface divergence of velocity depends only on the values of velocity at adjacent nodes. This is not the case with Loop subdivision, where the value of surface divergence on one node depends on the values of velocity at each node separated by at most two edges. There is thus an important loss of information in the C^0 approximation of $\mathbf{F}^\gamma, \mathbf{D}, \mathbf{G}$, which is balanced by the inclusion of the inverse of the collocation matrix ($\mathbf{C^{-1}}$). Note that the inverse of collocation matrix ($\mathbf{C^{-1}}$) depends only on mesh connectivity and not on the real shape of the mesh. Thus, it has to be computed only once at the beginning of the simulation.

This preconditioner is computed once at the beginning of each time step with the current position $\mathbf{x_n}$ of the interface, an then an LU-factorization is performed using LAPACK [1]. This factorization is then used for all the

intermediate steps if Runge-Kutta scheme is used or for all the Newton iterations if trapezoidal scheme is used. This renders simulations of vesicles much costlier than drop or capsule simulations, since precondition factorization can represent as much as 50% of the computational cost of a time step. Also, viscosity contrast was not included in the $\mathbf{G}$ matrix to avoid the need to solve an additional linear system. However, as shown in table 1, the preconditioning is still efficient even with large viscosity constrast. For the coupled system given by (58), the preconditioner is simply

$$\mathbf{P} = \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{D}\mathbf{C}^{-1}\mathbf{G}\mathbf{F}^{\gamma} \end{pmatrix} \tag{51}$$

3.5. Time stepping algorithm

Two solvers are used for integrating the time evolution of the interface given by equation (9) : an explicit high-order solver and an implicit solver. Note that, for a given position of the interface $\mathbf{x_n}$, membrane forces can be computed directly from equation (31) and then the velocity can be computed by equation (18), possibly with a projection stage described in section 3.4 in order to satisfy the surface incompressibility constraint. We note by $\mathbf{v}(\mathbf{x_n})$ the velocity computed at position $\mathbf{x_n}$.

The explicit solver is a Runge-Kutta Fehlberg fourth-fifth order solver [21], whose advantages are high-order and adaptive time stepping, allowing for very good preservation of invariants such as enclosed volume of fluid. Given a position and tension $(\mathbf{x_n}, \gamma_n)$ at time t_n, the position and tension $(\mathbf{x_{n+1}}, \gamma_{n+1})$ at time t_{n+1} are such that:

$$\mathbf{x_{n+1}} = \mathbf{x_n} + \Delta t \sum_{i=0}^{4} c_i \mathbf{v_n^{(i)}} \tag{52}$$

where the intermediate velocities are computed by:

$$\begin{aligned} \mathbf{v_n^{(0)}} &= \mathbf{v}(\mathbf{x_n}, \gamma_n) \quad , \quad \boldsymbol{\nabla}_s \cdot \mathbf{v_n^{(0)}} = 0 \\ \mathbf{v_n^{(k)}} &= \mathbf{v}\left(\mathbf{x_n^{(k)}}, \gamma_n^{(k)}\right) \quad , \quad \boldsymbol{\nabla}_s \cdot \mathbf{v_n^{(k)}} = 0 \\ \text{with } \mathbf{x_n^{(k)}} &= \mathbf{x_n} + \Delta t \sum_{i=0}^{k-1} \beta_{ik} v_n^{(i)} \end{aligned} \tag{53}$$

and c_i, β_{ik} the coefficients of the Runge-Kutta Fehlberg method [21]. The intermediate velocities $\mathbf{v_n^{(k)}}$ are computed by moving the interface position

to $\mathbf{x_n^{(k)}}$ and using this intermediate position to compute all integrals needed to build systems given by (31) and (18). Note that if this scheme is used for vesicles, the surface incompressibility constraint is enforced at each Runge-Kutta stage (see section 3.4 for details). A fifth order combination is also used to form another solution $\hat{\mathbf{x}}_{\mathbf{n+1}}$

$$\hat{\mathbf{x}}_{\mathbf{n+1}} = \hat{\mathbf{x}}_{\mathbf{n}} + \Delta t \sum_{i=0}^{5} \hat{c}_i \mathbf{v}_{\mathbf{n}}^{(\mathbf{i})} \tag{54}$$

which permits to estimate numerical error and thus select adaptively the time step.

For capsules without bending, the stability condition $\Delta t \leq O(\Delta x / G_s)$ allows reasonable time steps to be selected, while for vesicles, the stability condition $\Delta t \leq O(\Delta x^3 / \kappa)$ prevents the use of such an explicit scheme in practical conditions.

The implicit solver is the trapezoidal time scheme. Given a position and tension $(\mathbf{x_n}, \gamma_n)$ at time t_n, the position and tension $(\mathbf{x_{n+1}}, \gamma_{n+1})$ at time t_{n+1} are such that:

$$\begin{aligned} &\mathbf{x_{n+1}} = \mathbf{x_n} + \frac{\Delta t}{2} \left[\mathbf{v}(\mathbf{x_n}, \gamma_\mathbf{n}) + \mathbf{v}(\mathbf{x_{n+1}}, \gamma_\mathbf{n+1}) \right] \\ &\boldsymbol{\nabla}_s \cdot \mathbf{v}(\mathbf{x_{n+1}}, \gamma_\mathbf{n+1})) = 0 \end{aligned} \tag{55}$$

To solve these implicit equations, we proceed by iteration, and let

$$\begin{aligned} &{\mathbf{x_{n+1}}}^{(0)} = \mathbf{x_n} + \Delta t (\mathbf{v}(\mathbf{x_n}, \gamma_n)) \\ &\gamma_{n+1}^{(0)} = \gamma_n \end{aligned} \tag{56}$$

For the first time step, the tension γ_0 is found by solving $\boldsymbol{\nabla}_s \cdot \mathbf{v}(\mathbf{x_0}, \gamma_\mathbf{0}) = 0$. Now, given a value of $({\mathbf{x_{n+1}}}^{(r)}, \gamma_{n+1}^{(r)})$ at the r-th iteration, the residual writes:

$$\begin{aligned} res_x &= \mathbf{x_{n+1}^{(r)}} - \left[\mathbf{x_n} + \frac{\Delta t}{2} (\mathbf{v}(\mathbf{x_n}, \gamma_n) + \mathbf{v}(\mathbf{x_{n+1}^{(r)}}, \gamma_{n+1}^{(r)})) \right] \\ res_\gamma &= \boldsymbol{\nabla}_s \cdot \mathbf{v}(\mathbf{x_{n+1}^{(r)}}, \gamma_{n+1}^{(r)}) \end{aligned} \tag{57}$$

If both residuals are less than a prescribed tolerance, the iteration finishes. If not, we seek a correction $(\delta \mathbf{x}, \delta\gamma)$ such that $(\mathbf{x_{n+1}^{(r)}} + \delta\mathbf{x}, \gamma_{n+1}^{(r)} + \delta\gamma)$ satisfies the linearization of (55):

$$\begin{aligned} &\left[\mathbf{I} - \frac{\Delta t}{2} J^x \right] \delta\mathbf{x} - \frac{\Delta t}{2} \mathbf{v}^\gamma(\mathbf{x_{n+1}^{(r)}}, \delta\gamma) = -res_x \\ &\boldsymbol{\nabla}_s \cdot (J^x \delta\mathbf{x}) + \boldsymbol{\nabla}_s \cdot \left(\mathbf{v}^\gamma(\mathbf{x_{n+1}^{(r)}}, \delta\gamma) \right) = -res_\gamma \end{aligned} \tag{58}$$

	Δt		
N_{el}	5.10^{-4}	10^{-3}	2.10^{-3}
320	6.3 (2)	6.4 (2)	7.4 (2)
1280	7.3 (2.1)	9.2 (2.3)	10.5 (2.8)
5120	16.3 (3)	18.8 (4)	33.1 (4)

Table 2: Average GMRES iteration count for the resolution of equation (58), the average number of Newton iterations (r) necessary to solve (57) is indicated in parenthesis. The average is performed over the 10 first time steps of a simulation of a vesicle in shear flow with $Ca = 1$.

where J^x is the Jacobian with respect to $\mathbf{x}$, and $\mathbf{v}^\gamma$ is the velocity due solely to tension, that is, the velocity given by equation (16) with $\mathbf{f} = \mathbf{f}^\gamma$ and $\mathbf{v}^\infty = 0$. Note that tension-induced velocity is a linear operator in γ and thus $\mathbf{v}^\gamma(\mathbf{x}, \delta\gamma)$ is exactly the Jacobian with respect to γ applied to $\delta\gamma$, that is $J^\gamma \delta\gamma = \mathbf{v}^\gamma(\mathbf{x}, \delta\gamma)$. Equation (58) is solved with the Jacobian-free Newton-Krylov method [32]. Thus, in the GMRES solver, the Jacobian is not constructed explicitly, but only evaluated numerically through its action on vectors :

$$J^x \delta\mathbf{x} \approx \frac{\mathbf{v}(\mathbf{x}^{(\mathbf{r})}_{\mathbf{n+1}} + \epsilon\delta\mathbf{x}, \gamma_{n+1}) - \mathbf{v}(\mathbf{x}^{(\mathbf{r})}_{\mathbf{n+1}}, \gamma_{n+1})}{\epsilon} \tag{59}$$

with ϵ a small parameter [32]. Again, intermediate velocities are computed by moving the interface position to e.g. $\mathbf{x}^{(\mathbf{r})}_{\mathbf{n+1}} + \epsilon\delta\mathbf{x}$ and using this intermediate position to compute all integrals for (31) and (18).

The average number of GMRES iteration required to solve (58) and the average number of Newton iteration required to solve (57) are reported in table 2. It can be noticed that, due to the ill-conditioning of J^x, keeping the time step fixed while increasing the mesh resolution increases the number of iterations required. A possible remedy would be to design a preconditioner for the J^x operator, as has been done in [53] for vesicle simulations with spectral methods. Note that both the explicit and implicit solver are described for vesicle, but can be applied for capsules or drops, in which case the equations for tension γ are not solved.

3.6. Mesh quality preserving algorithm

For capsules, the shear elasticity prevents severe mesh distortion while for drops or vesicles, the membrane fluidity allows velocity fields that may

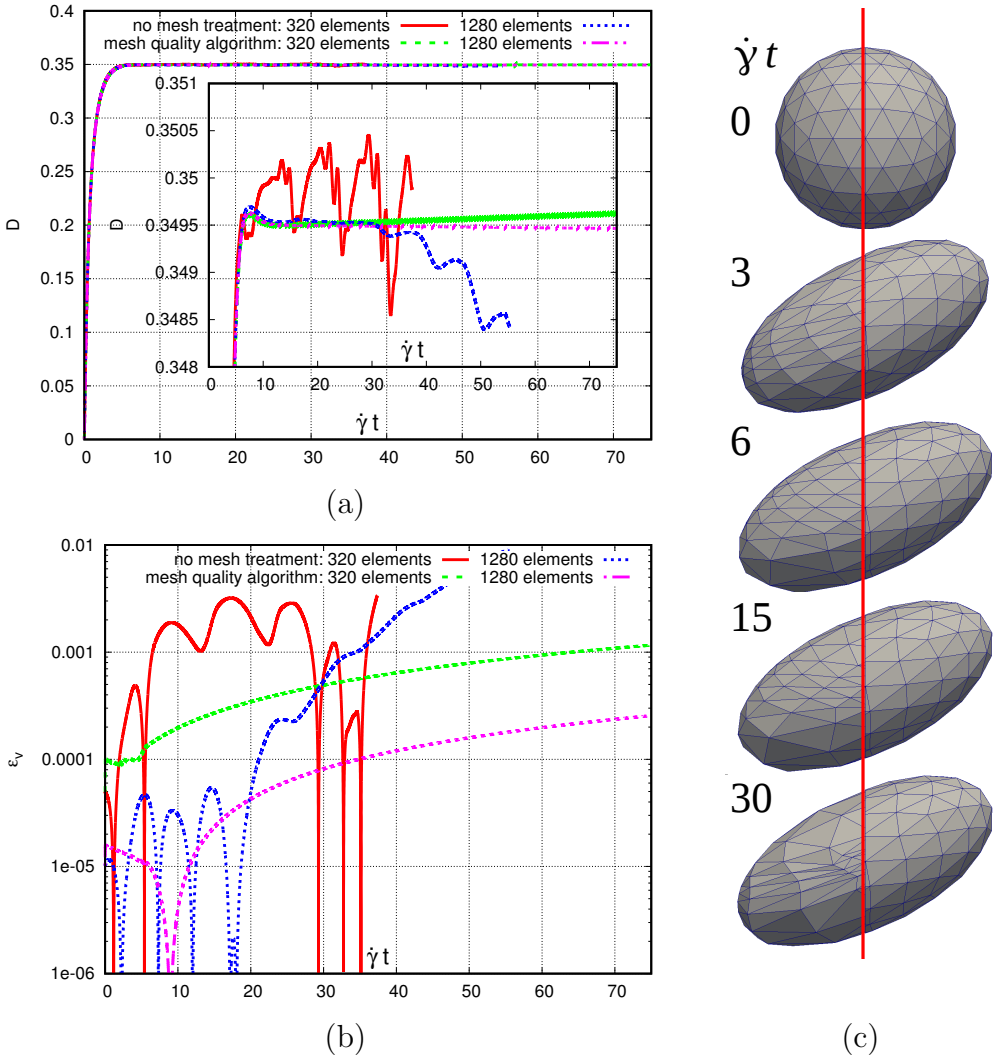
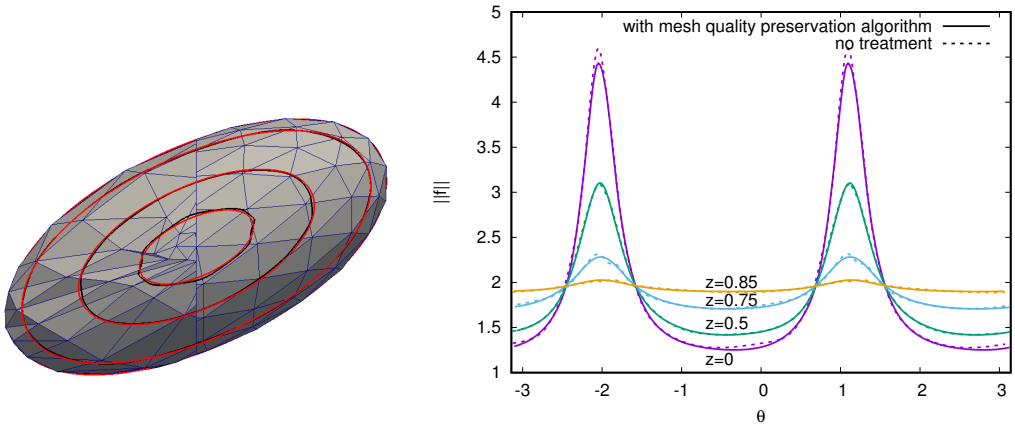


Figure 3: (color online) Effect of mesh quality preserving algorithm on a tank-treading drop $Ca = 0.3$: (a) evolution of Taylor parameter as a function of dimensionless time. (b) relative error on volume conservation (c) Snapshots of the shape and mesh at different dimensionless times (from top to bottom) : $\dot{\gamma}t = 0; 3; 6; 15; 30$. Left (respectively right) part of the shape is a simulation without (respectively with) mesh quality preservation algorithm.

strongly distort the mesh, as can be seen in figure 3 (c). Interestingly, even with badly shaped elements, the simulation is still stable for some time, and gives acceptable results. The most noticeable effect of poor mesh quality is an order of magnitude increase in the enclosed volume variation as shown in figure 3 (b), and increased variations in the Taylor parameter and inclination angle as shown in figure 3 (a), for a clean drop submitted to simple shear flow ($\mathbf{v}^\infty = \dot{\gamma} y \mathbf{e_x}$). Note that the oscillation amplitude in the Taylor parameter is less than 0.5% and thus that the global picture is still correct, even without any treatment on the control mesh. One may argue that Taylor deformation parameter is a global quantity which thus hides the local details. To check the impact of the control mesh quality on local quantities we take four cross-section of the shape at $z = 0; 0.5; 0.75; 0.85$ as shown in figure 4 (a) and compare results for a control mesh without any treatment (left part of the shape) and a control mesh with a mesh quality preservation algorithm (described in the following) applied (right part of the shape). One can first notice that the cross-sections almost overlap, showing that the local limit shapes are very similar. Then, we measure the membrane force norm (thus essentially the curvature) along these cross sections and plot the results as a function of the polar angle in figure 4 (b). Again, results are almost identical, with the most noticeable difference being that the shape without mesh treatment overestimates the curvature at the tip of the drop. This example illustrates that control mesh quality seems not too critical to obtain acceptable results, and that Loop elements are certainly more robust than linear or quadratic Lagrange elements. A more systematic study would be needed to investigate this point further. Moreover, it is still worthwhile to preserve mesh quality, as it improves volume conservation, reduces parasitic oscillations of the Taylor parameter, and more importantly, allows to keep a reasonable time step : indeed, if mesh elements become to small, the stable time step will decrease dramatically, leading to increased computational cost.

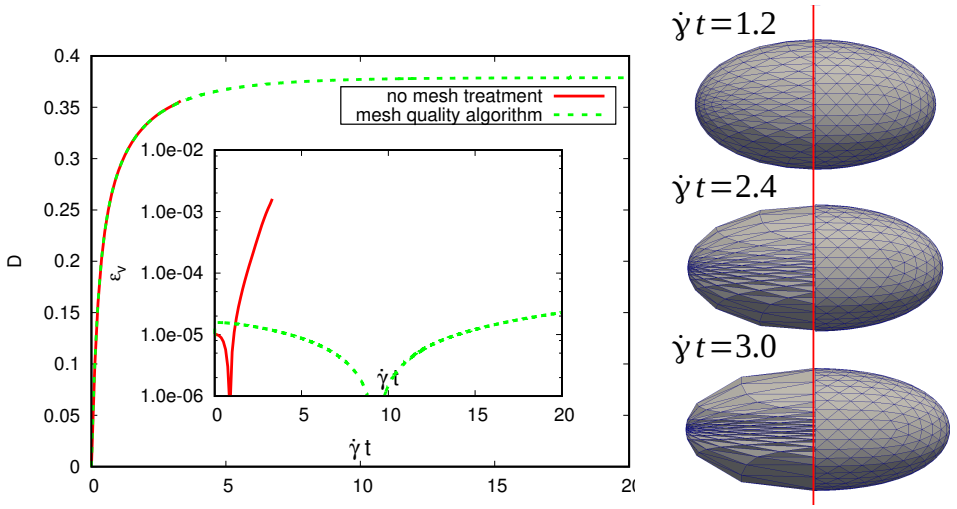
A mesh quality preserving algorithm has thus been designed to redistribute mesh nodes onto the interface.

This algorithm is similar in principle to existing algorithms such as [36, 63, 11, 56, 62] : using the fact that, for vesicles or drops, there is no tangential reference configuration means that the tangent velocities can be described in an Eulerian manner, that is, decoupled from mesh velocities. In other words, it is possible to move grid points along the surface in order to preserve/improve mesh quality, without affecting the physics. In



(a) (b)

Figure 4: Influence of control mesh quality on the solution at $\dot{\gamma}t = 37.5$ (parameters are the same as in figure 3. (a) control mesh nodes projected onto the limit shape, and cross section of the shape in the $z = 0; 0.5; 0.75; 0.85$ planes. Red (resp. black) curve is with (respectively without) mesh quality preserving algorithm. (b) comparison of the membrane force norm on the different cross-section. The curves are plotted as a function of θ, the polar angle defined with respect to x-axis.



(a) (b)

Figure 5: (a) Evolution of Taylor parameter for a clean drop in extensional flow as a function of dimensionless time. (Inset) Evolution of relative error on enclosed volume as function of dimensionless time. (b) Snapshots of the shape of a clean drop in extensional flow at different dimensionless times.

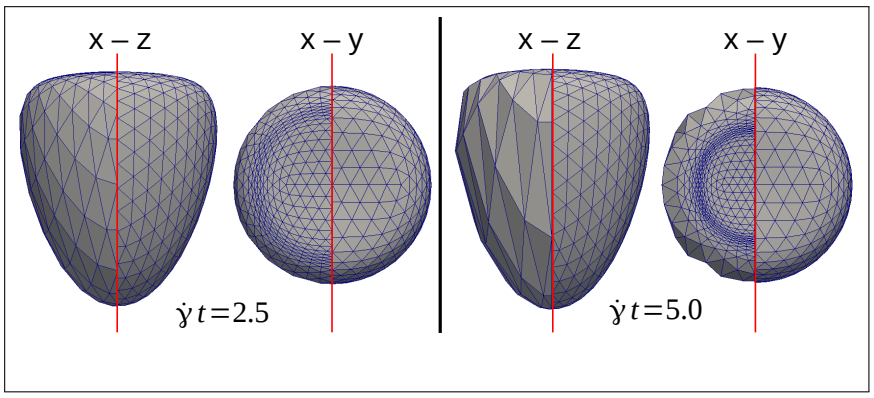


Figure 6: Effect of mesh quality preserving algorithm on a drop in Poiseuille flow ($Ca = 0.5$). Left part of the drop comes from a simulation without any node redistribution, right part comes from a simulation with application of algorithm described in section 3.6.

their pioneering work, [36] update the shape with only normal component of the physical velocity, and introduces a tangential "velocity" field aiming to preserve mesh quality. Subsequent developments [63, 11, 56, 62] are all based on introducing somehow a "mesh energy", either "kinetic energy" [63, 62], or "elastic energy" [11, 56]. In [63], passive stabilization algorithm seeks to preserve the edges current length, while introducing an elastic energy [56] may permit to improve mesh quality over time, for instance when starting with a not-optimal mesh. Adaptivity to local features [11, 62] is done by introducing local reference length scales based on e.g. curvature or element area. The idea of the present algorithm is to treat the mesh as a network of springs, similar to elastic mesh approach, but with varying stiffness aiming to take into account local properties. Doing so allows to keep the network under tension and avoids numerical instabilities that we found using a local reference length approach when this length scales varies too rapidly in the mesh or when the current configuration is too far from optimal. Elastic energy is then minimized by tangential displacements of the nodes along the shape that relax forces in the network. This is done by overdamped dynamics, that is computing on each node the velocity $\mathbf{u}^{\mathrm{MQ}}$ such that $\mathbf{f}^{\mathrm{MQ}} - \mu\mathbf{u}^{\mathrm{MQ}} = \mathbf{0}$ where μ is a fictitious viscosity and $\mathbf{f}^{\mathrm{MQ}}$ is the sum of forces due to network of springs. Since we do not want the redistribution of nodes to change the shape, only the tangent part of the mesh quality velocity is kept, thus $\mathbf{u}^{\mathrm{MQ}} = (\mathbf{I} - \mathbf{n} \otimes \mathbf{n}) \cdot \mathbf{f}^{\mathrm{MQ}}/\mu$. On a given node

of position $\mathbf{x_i}$, the mesh quality force is computed as:

$$\mathbf{f}^{\mathrm{MQ}}{}_i = \sum_{j \in \text{neighbours}} \left[(k_i + k_j) \left(l_{ij} - l_{ij}^0 \right) \frac{\mathbf{t_{ij}}}{l_{ij}} \right] \tag{60}$$

where $\mathbf{t_{ij}} = \mathbf{x_j} - \mathbf{x_i}$, $l_{ij} = ||\mathbf{t_{ij}}||$ and l_{ij}^0 is an equilibrium length, and the sum is taken over all "neighbors" nodes, that is, nodes connected by one edge to node i. This equilibrium length is defined as $\alpha \times L_{ij}$, where L_{ij} was the distance between $\mathbf{x_i}$ and $\mathbf{x_j}$ the last time that the mesh quality algorithm was applied. In practice, mesh quality algorithm is generally applied at the end of each time step, in which case L_{ij} is thus simply the distance between $\mathbf{x_i}$ and $\mathbf{x_j}$ at the beginning of the time step. $\alpha \leq 1$ is a parameter of the mesh quality algorithm algorithm to allow nodes displacement even if nodes have not moved between two time steps (e.g. a stationary state has been reached). The constants k_j are defined such that the mesh quality algorithm tries to maintain an uniform distribution of area of elements (variable A_e in 61), while also having a more refined mesh in highly curved regions (variable $(2H)_e^2$ in 61). Thus, k_j is computed as:

$$k_j = \frac{c_A}{< A >} \sum_{e \in E_j} \frac{A_e}{3} + \frac{c_H}{< (2H)^2 >} \sum_{e \in E_j} \frac{(2H)_e^2}{3} \tag{61}$$

where c_A, c_H are constants selecting the relative importance of the terms. A_e is the area of the element, $(2H)_e^2$ is the value of the curvature energy integrated over the element, and

$$< A >= \frac{1}{N} \int_S \mathrm{d}S \quad , \quad < (2H)^2 >= \frac{1}{N} \int_S (2H)^2 \mathrm{d}S \tag{62}$$

are the average value of area and curvature energy by node, with N the number of nodes in the mesh. The nodes positions are then evolved following:

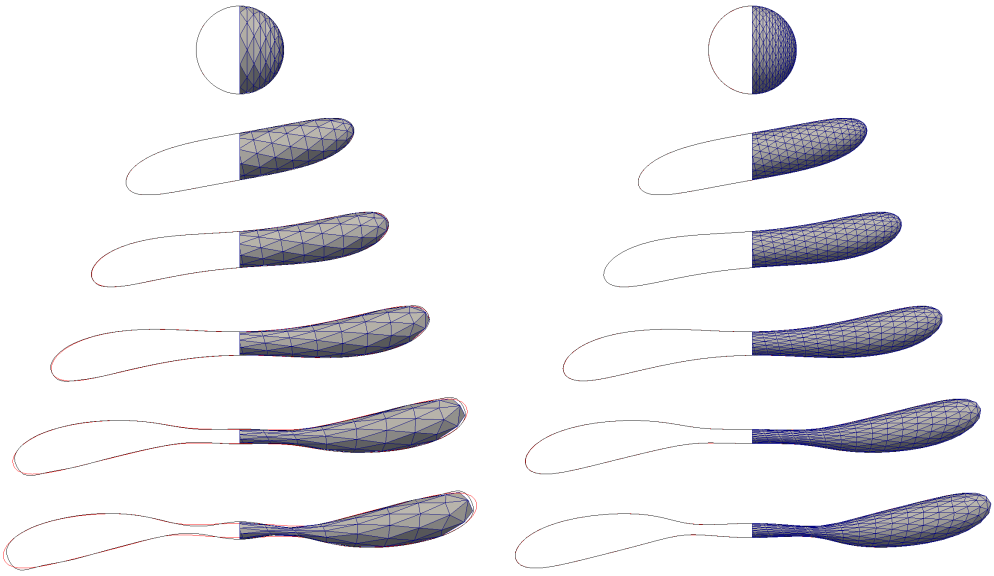
$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t^*} = \mathbf{u}^{\mathrm{remesh}} \tag{63}$$

Equation (63) is integrated in fictitious time t^* by an adaptive third order Runge-Kutta time integrator, until the residual of forces $\sum_i ||\mathbf{f_i}||$ is below a prescribed parameter ϵ. The effect of this mesh quality algorithm algorithm is illustrated with parameters ($\alpha = 0.9, \epsilon/c_A = 0.05, c_H = c_A$) which yielded good results for all cases simulated so far, as exemplified for a drop in: shear flow (see 3 (c)), planar extensional flow (see figure 3.6 (b)) and Poiseuille flow (see figure 3.6 (b)). Note that the value of the fictitious viscosity μ only changes the time scale necessary to reach the equilibrium.

4. Numerical validations and applications of the coupled algorithm

In this section, the algorithm is used to simulate several test cases, in order to validate its implementation and demonstrate possible applications. Since an extensive validation of the numerical algorithm applied to drops has been performed in [25], only an example of drop simulation is presented in section 4.1 and then the numerical algorithm and its implementation are tested with respect to reference results of the literature for capsules and vesicles dynamics.

4.1. Drops



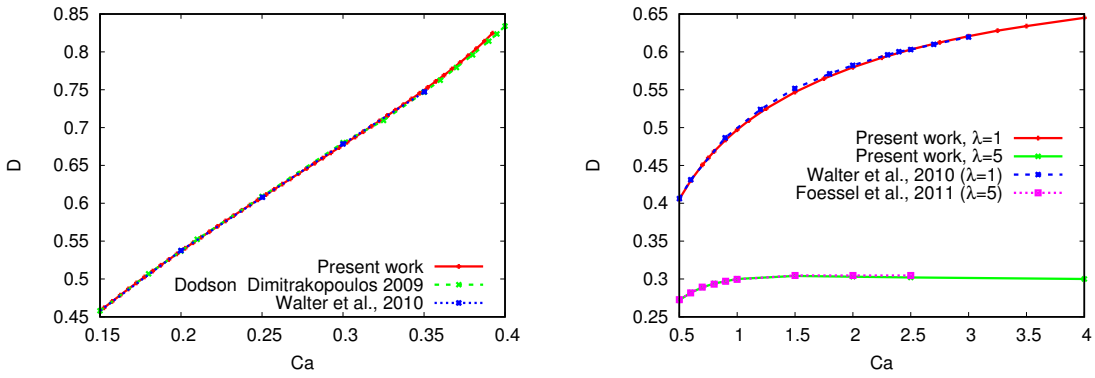
(a) 320 elements (b) 1280 elements

Figure 7: (color online) Evolution of a drop in shear flow with $Ca = 0.47$, with (a) 320 and (b) 1280 elements. From top to bottom, times are : $t = 0; 25; 50; 75; 95; 100$. The slice of the shape in the $x - y$ plane is shown, as well as the control mesh vertices projected onto their limit position. For comparison, the slice of the simulation with 5120 elements is superposed (red curve).

To illustrate the application of Loop subdivision elements to drop dynamics, we consider the evolution of a drop in simple shear flow ($\mathbf{v}^\infty = \dot{\gamma} y \mathbf{e_x}$), with a capillary number $Ca = 0.47$ above the critical capillary number for breakup: numerous works have considered similar simulations (in

particular, in the context of BEM methods, [11, 62]), but it is still a good illustration of the potential of the method. Starting from an initially spherical shape, the drop elongates and develops a thin neck connecting two bulges, which will then leads to breakup by neck pinching. Since the mesh used here is of fixed topology, the breakup process is not described. For such elongated shapes, a sphero-cylindrical mesh [19, 50] is more adapted than the classical spherical mesh obtained by regular subdivision of an icosahedron, and is used here. Simulations are performed with three different number of elements : 320, 1280 and 5120 elements. The evolution of the shape at different times is presented in figure 7. The difference between coarse mesh and fine mesh starts to be visible only in the latter stages of the elongation, while shapes obtained with medium and fine resolution cannot be distinguished for the whole duration of the simulation. Note also that at $t = 100$, the volume drift is around 0.7% for 320 elements and 0.03% for 1280 elements. Thus, even with low resolution and elongated shapes, it is possible to obtain acceptable results, which could be interesting for simulations of suspensions.

4.2. Capsules



(a) (b)

Figure 8: Comparison of present work with literature results for Taylor parameter D as a function of capillary number Ca for (a) neo-Hookean capsule in extensional flow [18, 55] (b) Skalak capsule in shear flow [55, 23]

Capsules dynamics and steady state deformation in simple flows have been studied by several independent groups and thus constitute an excellent validation case. We thus consider the steady state deformation of a Neo-Hookean (NH) capsule and of a Skalak (Sk) capsule, placed either in planar

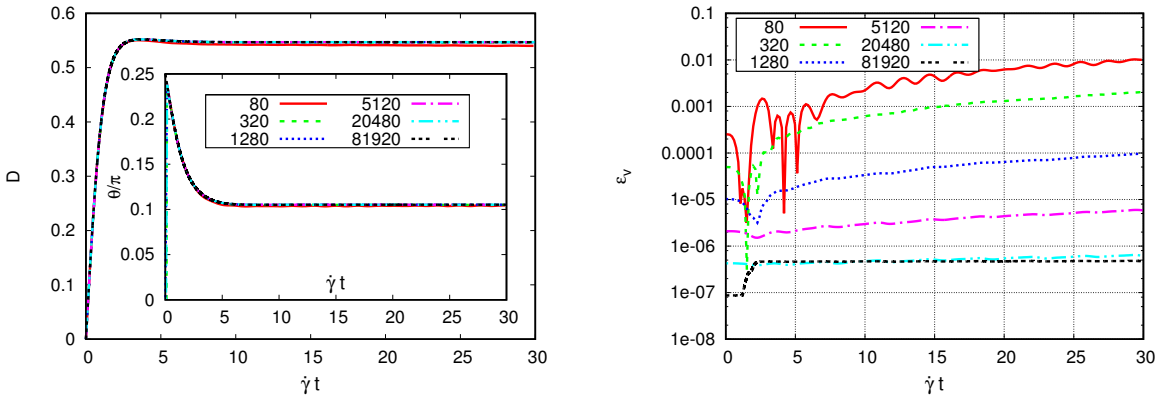
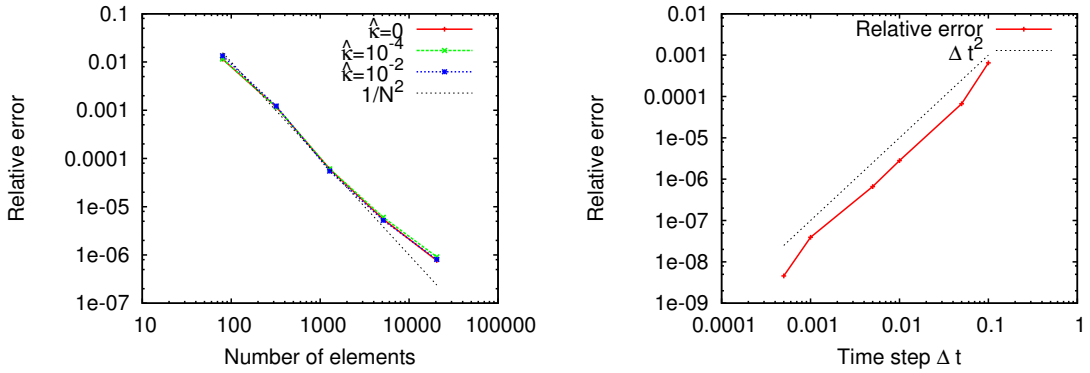


Figure 9: (a) Evolution of Taylor parameter and inclination angle of a Skalak capsule in shear flow ($Ca = 1.5$) as a function of time, for different number of elements. (b) Corresponding evolution of the relative error on enclosed volume as a function of time, for different number of elements.

extensional flow ($\mathbf{v}^\infty = \dot{\gamma}[x\mathbf{e_x} - y\mathbf{e_y}]$) or in simple shear flow ($\mathbf{v}^\infty = \dot{\gamma}y\mathbf{e_x}$). In both cases, an initially spherical capsule of radius R_0 will deform into a roughly ellipsoidal shape. The deformation of the capsule is measured by the Taylor parameter $D = \frac{L-l}{L+l}$, where L and l are semi-axis of the ellipsoid having the same tensor of inertia, and is represented as a function of the capillary number $Ca = \frac{\dot{\gamma}R_0\eta_{ext}}{G_s}$ in figures 8 (a) for NH capsule in extensional flow and 8 (b) for Sk capsule in shear flow. As can be seen in these figures, the agreement with results of [18, 55, 23] is excellent. The meshes used to obtain these results contains between 320 and 20480 elements.

A convergence study is carried out for a Sk capsule in shear flow ($Ca = 1.5, \lambda = 1$) to assess the dependency on the number of elements, which is varied from 80 to 81920. As shown in figure 9 (a), when the Taylor parameter D or the inclination angle θ are plotted as a function of dimensionless time $\dot{\gamma}t$, the results from the different simulations overlap almost perfectly, the only noticeable difference being seen with the coarsest mesh (80 elements). A more important effect of mesh refinement is seen on the preservation of enclosed volume: the relative error $\epsilon_v = \frac{V_0 - V(t)}{V_0}$ is represented as a function of dimensionless time in figure 9 (b), where $V_0 = \frac{4}{3}\pi R_0^3$ is the volume of the initially spherical capsule. This figure shows that one mesh refinement (multiplying the number of elements by four) leads to roughly gaining one order of magnitude in the volume preservation. However, one can also no-



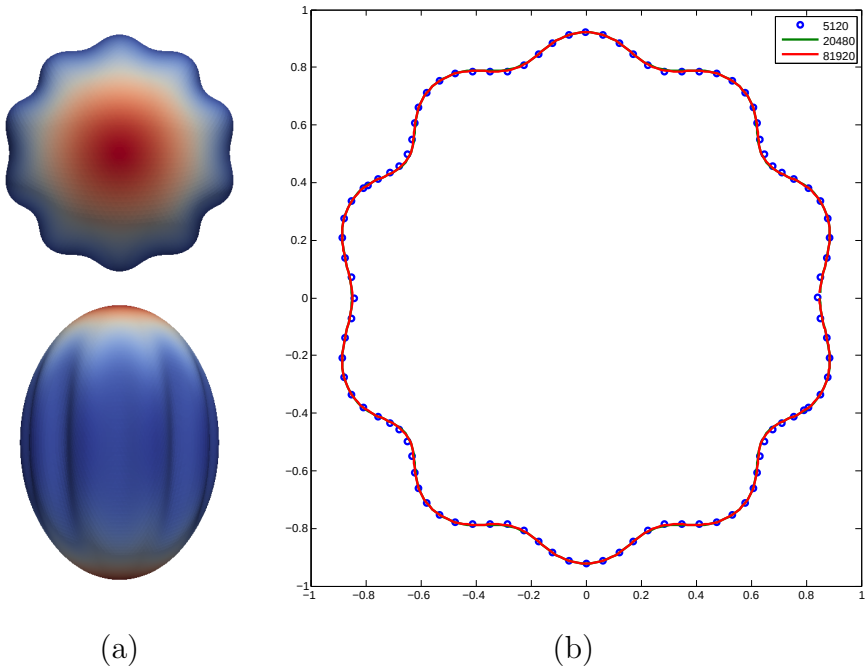
(a) (b)

Figure 10: Convergence study of the numerical algorithm : relative error on the steady state value (taken at $t = 26.25$) of the Taylor parameter of a Skalak capsule under shear flow ($Ca = 1.5$) with respect to (a) number of elements (b) time step size for trapezoidal scheme. For spatial convergence (a), the RK45 scheme is used and the reference solution is computed with 81920 elements. The spatial convergence of a capsule with bending resistance modeled by Helfrich membrane is also included for two values of the dimensionless bending modulus $\hat{\kappa} = 10^{-4}, 10^{-2}$. For temporal convergence of trapezoidal scheme, a mesh with 5120 elements is used, and the reference solution is computed with RK45 scheme.

tice that even with the coarsest mesh (80 elements) volume preservation is still acceptable : the volume drift after a dimensionless time of $\dot{\gamma}t = 30$ is only 1%.

To estimate the convergence order of the numerical method, the relative error $= \frac{D_{ref}-D}{D_{ref}}$ on Taylor parameter D at a dimensionless time $\dot{\gamma}t = 26.25$ (thus in stationary state) is represented as a function of the number of elements in figure 10 (a), with the reference solution computed with 81920 elements. As can be seen in the figure 10 (a), the relative error is second order in number of elements $O(N^{-2})$. Note that in [19], the authors mention that using the singularity-subtraction with standard quadrature will leads asymptotically to a $O(N^{-1})$ convergence because the gradient of the Green function is still singular. For the number of elements used here, this seems not to affect the convergence rate of our method, but could possibly degrade it for finer resolutions, in which case adaptive integration scheme as described in [20] could be used to restore the $O(N^{-2})$ convergence. To check whether inclusion of bending rigidity affects the convergence order of the method, we also run the same convergence test with two dimensionless bending modulus $\hat{\kappa} = \frac{\kappa}{GR_0^2} = 10^{-4}, 10^{-2}$. Note that in this test case, the

bending rigidity is added by simply taking into account Helfrich energy for the membrane. Results displayed in figure 10 (a) shows that the convergence order of the method is not affected by inclusion of bending rigidity, still displaying a $O(N^{-2})$ convergence rate. Finally, we also use this test case to check the time convergence order of the trapezoidal time scheme as described in section 3.5 : a mesh containing 5120 elements is used to compute the solution, both with a Runge-Kutta scheme (52) and with trapezoidal scheme (55) for various time step. As expected, the relative error on Taylor parameter converges as $O(\Delta t^2)$ when using the trapezoidal rule.



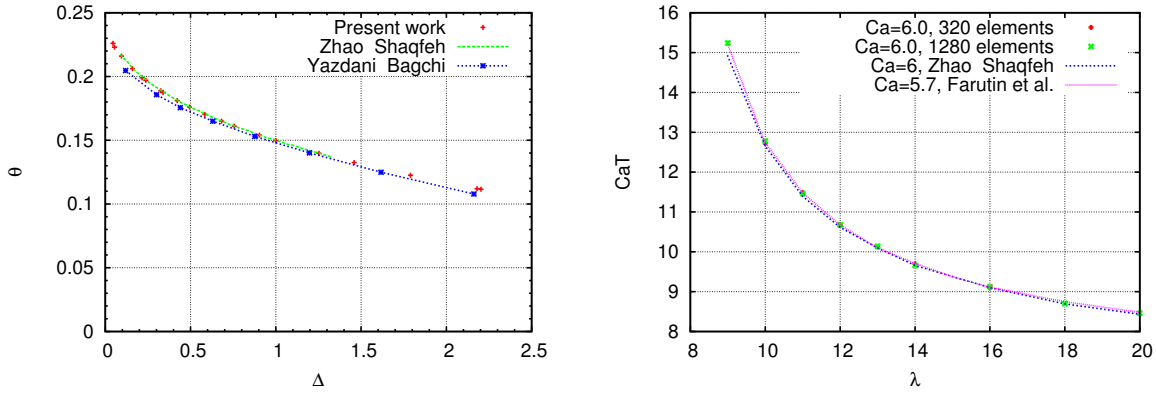
(a) (b)

Figure 11: Neo-Hookean capsule in uniaxial extensional flow ($Ca = 0,07; \hat{\kappa} = 10^{-4}$) : (a) Top - view in the x-y plane . Bottom - view in the y-z plane (b) Cross section in the $z = 0$ plane.

A possible application of our algorithm is to simulate capsules with bending resistance, which is necessary to describe capsule dynamics in the parameter region where compressive stresses are found. The simplest inclusion of bending resistance is to describe the interface energy as the simple sum of a two-dimensional membrane model (e.g., NH, Sk) and a Helfrich membrane. Note that more elaborate model of thin-shells can easily be implemented within the framework presented here, but we choose this ap-

proach for its simplicity. We thus simulate the stretching by an uniaxial extensional flow ($\mathbf{v}^\infty = \dot{\gamma}\left[-\frac{x}{2}\mathbf{e_x} - \frac{y}{2}\mathbf{e_y} + z\mathbf{e_z}\right]$) of a NH membrane supplemented with Helfrich energy. After an initial transient, the capsule reach a steady stated showed in figure 4.2. Runs at three different resolutions ($N = 5120, 20480, 81920$ elements) have been performed, and the comparison of the cross-section in the $z = 0$ plane is shown in figure 4.2 (b), showing an excellent agreement for all resolutions.

4.3. Vesicles



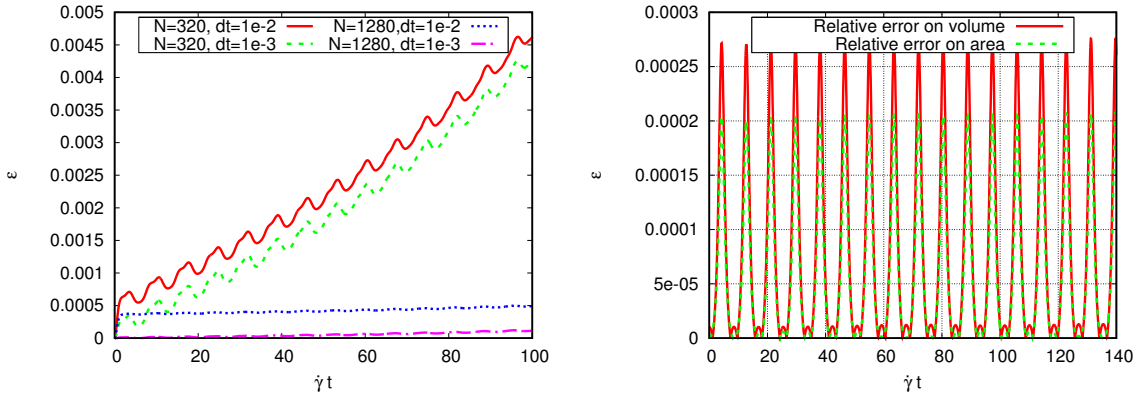
(a) (b)

Figure 12: (a) Steady state inclination angle θ for a tank-treading vesicle in simple shear flow ($\lambda = 1, Ca = 10$) as a function of excess area Δ. (b) Dimensionless VB-TR/TU period (T) as a function of viscosity contrast λ for a quasi-spherical vesicle ($v = 0.95, Ca = 6.0$)

We then test our numerical algorithm applied to vesicles on the prototypical configuration of a vesicle immersed in shear flow : $\mathbf{v}^\infty = \dot{\gamma} y \mathbf{e_x}$. In this configuration, several dynamics may be observed, the simplest being a tank-treading configuration where the vesicle reach a steady deformation and inclination angle similarly to drops or capsules. This steady angle depends on the viscosity contrast λ and on the excess area Δ which measures the deformability of the vesicle. Indeed, since for a vesicle both enclosed volume and total surface must be preserved, an initially spherical vesicle will not deform, whatever the flow strength is. To be deformed, the vesicle must thus be deflated, and a measure of this deflation is provided either by the reduced volume $v = \frac{V}{\frac{4}{3}\pi\left(\frac{A}{4\pi}\right)^{3/2}}$ or by the excess area $\Delta = A\left(\frac{4\pi}{3V}\right)^{2/3} - 4\pi$.

The ratio between external flow time scale and bending time scale defines the capillary number : $Ca = \frac{\dot{\gamma}\eta_{ext}R_0^3}{\kappa}$. The evolution of the inclination angle as a function of excess area for a vesicle without viscosity contrast is reported in figure 12 (a) where it is compared to spectral simulations of [59] and simulations of [57]. The agreement with [59] is excellent, even with 320 elements.

If one includes viscosity contrast, different dynamics can be observed, such as tumbling, where the vesicle inclination axis undergoes a full rotation. In between tank-treading and tumbling, for sufficiently high capillary number, a more complex dynamics termed vacillating-breathing [40] or trembling [17, 59] can be observed, where the inclination axis undergoes oscillation without full rotations, while the shape also oscillates. The evolution of the VB-TR / TU period is plotted as a function of viscosity contrast in figure 12 (b) and compared to spectral simulations of [59] and simulations of [19], showing an excellent agreement with both, except for $\lambda = 9$ where results deviates slightly from [59], but seems consistent with [19] in doing so.

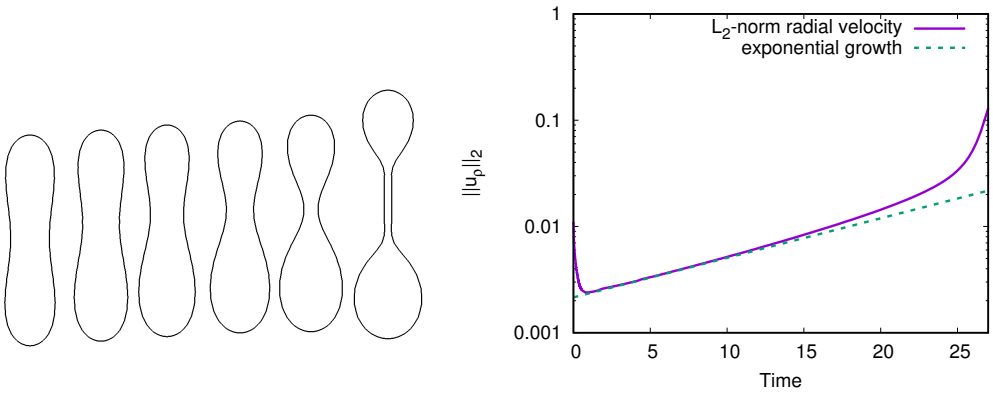


(a) (b)

Figure 13: (a) Relative error on reduced volume for a tank treading vesicle in simple shear flow ($Ca = 10, \lambda = 1, v = 0.95$), comparison of two meshes and two different time steps. (b) Relative error on volume and area for a tumbling vesicle in shear flow ($Ca = 6, \lambda = 20$), for 1280 elements and a time step of $\Delta t = 10^{-2}$.

For vesicles, both the enclosed volume and the total surface should be preserved. The relative error for a tank treading vesicle is shown in figure 13 (a), and for a tumbling vesicle in figure 13 (b). In figure 13 (a), influence of

mesh size and time step on reduced volume preservation is also studied. The reduced volume drifts for long simulation times (stationary state is reached around $\dot{\gamma}t \approx 5$), but even with the coarsest mesh the relative error is below 0.5% after several full tank-treading period. As in figure 9 (a), refining the mesh once leads to an order of magnitude decrease in the variation of reduced volume, because the slope of the drift depends on the mesh used. There is also an important effect of the time step used, because for 1280 elements, the major part of the variation of the enclosed volume occurs in the transient part where the vesicles deforms from its initial ellipsoidal shape towards the stationary tank treading shape. Significant gains in reduced volume preservation could thus be obtained by using an adaptive implicit time scheme, as has been done recently by [46].

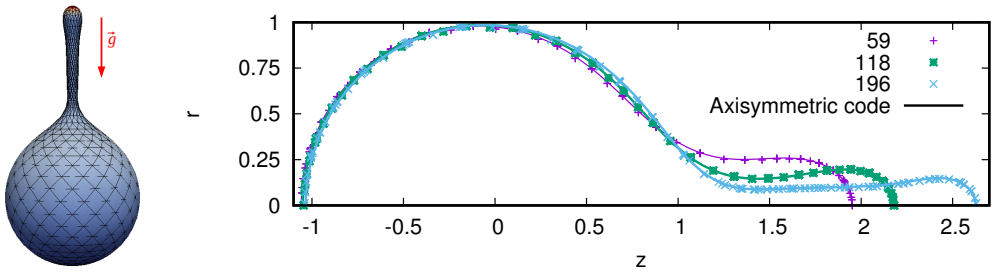


(a) (b)

Figure 14: (a) Snapshots of shape at different dimensionless times $\dot{\gamma}t = 0; 18.1; 23.3; 25.9; 27.2; 28.5$ (b) Evolution of L_2 norm of the radial velocity. The dashed line corresponds to an exponential growth with a growth rate of $\omega = 8.57 \times 10^{-2}$ as computed by [60, 50].

We also simulate a deflated $v = 0.65$ vesicle in uniaxial extensional flow ($\mathbf{v}^\infty = \dot{\gamma}\left[-\frac{x}{2}\mathbf{e_x} - \frac{y}{2}\mathbf{e_y} + z\mathbf{e_z}\right]$), which is known [60, 50, 42, 12] to exhibit instability. For comparison, we follow the procedure described in [60, 50]: starting from the equilibrium shape at $v = 0.65$, we first simulate the stationary state under flow, with $Ca_A = 4$. Here, $Ca_A = \frac{\dot{\gamma}\eta}{\kappa}\left(\frac{A}{4\pi}\right)^{\frac{3}{2}}$ is the capillary number with a reference length based on the area. After reaching the stationary state, the shape is then perturbed with a sinusoidal perturbation, and the L_2-norm of the radial velocity is monitored as a function of

time, as in [60, 50]. Snapshots of the shape at different dimensionless times $\dot{\gamma}t$ as well as evolution of L_2 norm of radial velocity are shown in figure 14, and show an excellent agreement with reference results of [60, 50].



(a) (b)

Figure 15: (a) Stationary shape of a settling deflated vesicle (= 0.93, $Bo = 216$) (b) Stationary profiles for different Bond numbers ($Bo = 59; 118; 196$). Symbols are present results and solid lines are comparison with axisymmetric code [9, 52].

Finally, we simulate vesicles settling due to a density contrast between the inner and outer fluids, and compare with results obtained with an axisymmetric code using linear elements [9, 52]. The dimensionless parameter controlling the shape is the Bond number, defined as $Bo = \frac{\Delta \rho g R_0^4}{\kappa}$. Starting with a prolate ellipsoid of excess area $\Delta = 0.93$, shape evolves toward a rounded vesicle at the front with a protrusion at the rear, whose length reaches a plateau at stationary state. Depending on the strength of the forcing, shapes varies continuously from pear-like shapes to tethered shapes. Comparison of profiles showed in figure 15 show an excellent agreement with [9, 52].

5. Conclusion

In this paper, we have presented an algorithm to simulate the dynamics of soft objects such as drops, capsules or vesicles under Stokes flow. The algorithm is based on the use of Loop subdivision surfaces to describe the interface of these objects. This surface representation is used both for the finite element solver for membrane forces and for the boundary element method for the fluid solver. The surface incompressibility constraint for vesicles is treated by the introduction of a local Lagrange multiplier which is solved by an iterative method. To avoid the growth of the number of

iterations, a preconditioner based on a linear approximation of the surface [7] is introduced. The whole algorithm spatial discretization error decreases as $O(N^{-2})$, where N is the typical mesh number of elements (or vertices), and is coupled with either a second order implicit time scheme for simulation with bending rigidity, or with an adaptive RK45 time scheme for capsule or drop simulations.

The combination of an high order spatial scheme with second order time algorithm allows a very good preservation of invariants such as enclosed volume or area for vesicles, without any correction on the velocity field, introduction of a reference/target volume or area, or rescaling of the shape after each time step. Note that it is of course possible to include these corrections into the proposed algorithm without difficulties to eliminate possible drift of the solution at long simulation times, but these corrections cannot be used for simulations of multiple objects where the rescaling of the shape will lead to a modification of inter-objects distances, which might crucially affect the dynamics. We plan to improve the conservation of invariants by using adaptive quadratures to limit volume drift and adaptive time stepping with implicit algorithm to limit both volume and area variations, such as the recent algorithm of [46] designed for 2D vesicles suspensions.

Finally, the algorithm presented here can be extended to account for more complex interfacial rheology (e.g. surface viscosity effects, Marangoni effects). Its extension to drops with viscous interfaces has been discussed in [25]. The inclusion of surface viscosity for other soft objects (capsules, vesicles) is likely to strongly influence the dynamics, and might be necessary to have a more accurate representation of real capsules, as recent experiments suggest [16].

6. Acknowledgements

The authors would like to thank Paul Gang Chen for careful reading of the manuscript. This work has benefited from financial support from the ANR Polytransflow (grant no.13-BS09-0015-01), from Labex MEC (grant no. ANR-11-LABX-0092), from A*MIDEX (grant no. ANR-11-IDEX-0001-02) and from CNES. This work was granted access to the HPC resources of Aix-Marseille Université financed by the project Equip@Meso (ANR-10-EQPX-29-01).

[1] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. *LAPACK Users'*

Guide. Society for Industrial and Applied Mathematics, Philadelphia, PA, third edition, 1999.

[2] Dominique Barthès-Biesel. Motion and deformation of elastic capsules and vesicles in flow. *Annual Review of Fluid Mechanics*, 48(1):25–52, 2016.

[3] DOMINIQUE BARTHS-BIESEL, ANNA DIAZ, and EMMANUELLE DHENIN. Effect of constitutive laws for two-dimensional membranes on flow-induced capsule deformation. *Journal of Fluid Mechanics*, 460:211222, Jun 2002.

[4] D. Barths-Biesel. Motion of a spherical microcapsule freely suspended in a linear shear flow. *Journal of Fluid Mechanics*, 100(4):831853, Oct 1980.

[5] D. Barths-Biesel and J. M. Rallison. The time-dependent deformation of a capsule freely suspended in a linear shear flow. *Journal of Fluid Mechanics*, 113:251267, Dec 1981.

[6] Thierry Biben, Alexander Farutin, and Chaouqi Misbah. Three-dimensional vesicles under shear flow: Numerical study of dynamics and phase diagram. *Phys. Rev. E*, 83:031921, Mar 2011.

[7] G. Boedec, M. Leonetti, and M. Jaeger. 3d vesicle dynamics simulations with a linearly triangulated surface. *J. Comp. Phys.*, 230:1020–1034, 2011.

[8] G. Boedec, M. Leonetti, and M. Jaeger. Mechanics of a helfrich membrane as a two-dimensional cosserat medium. *in preparation*, 2016.

[9] Gwenn Boedec, Marc Jaeger, and Marc Leonetti. Sedimentation-induced tether on a settling vesicle. *Phys. Rev. E*, 88:010702, Jul 2013.

[10] Fehmi Cirak, Michael Ortiz, and Peter Schröder. Subdivision surfaces: a new paradigm for thin-shell finite-element analysis. *International Journal for Numerical Methods in Engineering*, 47(12):2039–2072, 2000.

[11] Vittorio Cristini, Jerzy Bawzdziewicz, and Michael Loewenberg. An adaptive mesh algorithm for evolving surfaces: Simulations of drop breakup and coalescence. *Journal of Computational Physics*, 168(2):445 – 463, 2001.

[12] Joanna B. Dahl, Vivek Narsimhan, Bernardo Gouveia, Sanjay Kumar, Eric S. G. Shaqfeh, and Susan J. Muller. Experimental observation of the asymmetric instability of intermediate-reduced-volume vesicles in extensional flow. *Soft Matter*, 12:3787–3796, 2016.

[13] Timothy Davis. Algorithm 832: Umfpack, an unsymmetric-pattern multifrontal method. *ACM Transactions on Mathematical Software*, 30(2):196–199, 2004.

[14] Timothy Davis. *Direct Methods for Sparse Linear Systems*. SIAM, 2006.

[15] C. de Loubens, J. Deschamps, G. Boedec, and M. Leonetti. Stretching of capsules in an elongation flow, a route to constitutive law. *Journal of Fluid Mechanics*, 767, 3 2015.

[16] C. de Loubens, J. Deschamps, F. Edwards-Levy, and M. Leonetti. Tank-treading of microcapsules in shear flow. *Journal of Fluid Mechanics*, 789, 2016.

[17] J. Deschamps, V. Kantsler, and V. Steinberg. Phase diagram of single vesicle dynamical states in shear flow. *Phys. Rev. Lett.*, 102:118105, Mar 2009.

[18] W. R. Dodson and P. Dimitrakopoulos. Dynamics of strain-hardening and strain-softening capsules in strong planar extensional flows via an interfacial spectral boundary element algorithm for elastic membranes. *Journal of Fluid Mechanics*, 641:263–296, 12 2009.

[19] Alexander Farutin, Thierry Biben, and Chaouqi Misbah. 3d numerical simulations

of vesicle and inextensible capsule dynamics. *Journal of Computational Physics*, 275(0):539 – 568, 2014.

[20] Alexander Farutin and Chaouqi Misbah. Exact singularity subtraction from boundary integral equations in modeling vesicles and red blood cells. *Numerical Mathematics: Theory, Methods Applications*, 7:413–434, 2014.

[21] E. Fehlberg. Low-order classical runge-kutta formulas with step size control and their application to some heat transfer problems. Technical report, NASA, 1969.

[22] Feng Feng and William S. Klug. Finite element modeling of lipid bilayer membranes. *J. Comput. Phys.*, 220(1):394–408, December 2006.

[23] É. Foessel, J. Walter, A.-V. Salsac, and D. Barthès-Biesel. Influence of internal viscosity on the large deformation and buckling of a spherical capsule in a simple shear flow. *Journal of Fluid Mechanics*, 672:477–486, 4 2011.

[24] Jonathan B. Freund. Numerical simulation of flowing blood cells. *Annual Review of Fluid Mechanics*, 46(1):67–95, 2014.

[25] J. Gounley, G. Boedec, M. Jaeger, and M. Leonetti. Influence of surface viscosity on droplets in shear flow. *Journal of Fluid Mechanics*, 791:464–494, 3 2016.

[26] A. E. Green and J.E. Adkins. *Large Elastic Deformations.* Clarendon, Oxford, 1970.

[27] A. E. Green and W. Zerna. *Theoretical Elasticity.* Dover, New York, 1968.

[28] Luca Heltai, Marino Arroyo, and Antonio DeSimone. Nonsingular isogeometric boundary element method for stokes flows in 3d. *Computer Methods in Applied Mechanics and Engineering*, 268(0):514 – 539, 2014.

[29] Wei-Xi Huang, Cheong Bong Chang, and Hyung Jin Sung. Three-dimensional simulation of elastic capsules in shear flow by the penalty immersed boundary method. *Journal of Computational Physics*, 231(8):3340 – 3364, 2012.

[30] T.J.R. Hughes, J.A. Cottrell, and Y. Bazilevs. Isogeometric analysis: Cad, finite elements, nurbs, exact geometry and mesh refinement. *Computer Methods in Applied Mechanics and Engineering*, 194(3941):4135 – 4195, 2005.

[31] A.A. Joneidi, C.V. Verhoosel, and P.D. Anderson. Isogeometric boundary integral analysis of drops and inextensible membranes in isoviscous flow. *Computers Fluids*, 109:49 – 66, 2015.

[32] D.A. Knoll and D.E. Keyes. Jacobian-free newtonkrylov methods: a survey of approaches and applications. *Journal of Computational Physics*, 193(2):357 – 397, 2004.

[33] Duc-Vinh Le. Subdivision elements for large deformation of liquid capsules enclosed by thin shells. *Computer Methods in Applied Mechanics and Engineering*, 199(3740):2622 – 2632, 2010.

[34] Duc-Vinh Le and Sum Thai Wong. A front-tracking method with catmullclark subdivision surfaces for studying liquid capsules enclosed by thin shells in shear flow. *Journal of Computational Physics*, 230(9):3538 – 3555, 2011.

[35] Xuejin Li, Petia M. Vlahovska, and George Em Karniadakis. Continuum- and particle-based modeling of shapes and dynamics of red blood cells in health and disease. *Soft Matter*, 9:28–37, 2013.

[36] M. Loewenberg and E. J. Hinch. Numerical simulation of a concentrated emulsion in shear flow. *Journal of Fluid Mechanics*, 321:395–419, 8 1996.

[37] Charles Teorell Loop. Smooth subdivision surfaces based on triangles. Master's thesis, University of Utah, 1987.

[38] Lin Ma and William S. Klug. Viscous regularization and r-adaptive remeshing for finite element analysis of lipid membrane mechanics. *Journal of Computational Physics*, 227(11):5816 – 5835, 2008.

[39] S. Mendez, E. Gibaud, and F. Nicoud. An unstructured solver for simulations of deformable particles in flows at arbitrary reynolds numbers. *Journal of Computational Physics*, 256:465 – 483, 2014.

[40] Chaouqi Misbah. Vacillating breathing and tumbling of vesicles under shear flow. *Phys. Rev. Lett.*, 96:028104, Jan 2006.

[41] P. M. Naghdi. *The Theory of Shells and Plates*, pages 425–640. Springer Berlin Heidelberg, Berlin, Heidelberg, 1973.

[42] Vivek Narsimhan, Andrew P. Spann, and Eric S. G. Shaqfeh. The mechanism of shape instability for a vesicle in extensional flow. *Journal of Fluid Mechanics*, 750:144190, Jul 2014.

[43] C. Pozrikidis. *Boundary integral and singularity methods for linearized viscous flow*. Cambridge university press, 1992.

[44] C. POZRIKIDIS. Effect of membrane bending stiffness on the deformation of capsules in simple shear flow. *Journal of Fluid Mechanics*, 440:269291, Aug 2001.

[45] C. Pozrikidis. Interfacial dynamics for stokes flow. *Journal of Computational Physics*, 169(2):250 – 301, 2001.

[46] Bryan Quaife and George Biros. Adaptive time stepping for vesicle suspensions. *Journal of Computational Physics*, 306:478 – 499, 2016.

[47] Abtin Rahimian, Shravan K. Veerapaneni, Denis Zorin, and George Biros. Boundary integral method for the flow of vesicles with viscosity contrast in three dimensions. *J. Comput. Phys.*, 298(C):766–786, October 2015.

[48] Youcef Saad and Martin H Schultz. Gmres: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 7(3):856–869, July 1986.

[49] R Skalak, A Tozeren, R P Zarda, and S Chien. Strain energy function of red blood cell membranes. *Biophysical Journal*, 1973.

[50] Andrew P. Spann, Hong Zhao, and Eric S. G. Shaqfeh. Loop subdivision surface boundary integral method simulations of vesicles at low reduced volume ratio in shear and extensional flow. *Physics of Fluids*, 26(3), 2014.

[51] Jos Stam. Evaluation of loop subdivision surfaces. *SIGGRAPH'99 Course Notes*, 1999.

[52] R. Trozzo, G. Boedec, M. Leonetti, and M. Jaeger. Axisymmetric boundary element method for vesicles in a capillary. *Journal of Computational Physics*, 289:62 – 82, 2015.

[53] Shravan K. Veerapaneni, Abtin Rahimian, George Biros, and Denis Zorin. A fast algorithm for simulating vesicle flows in three dimensions. *Journal of Computational Physics*, 230(14):5610 – 5634, 2011.

[54] Petia M. Vlahovska, Thomas Podgorski, and Chaouqi Misbah. Vesicles and red blood cells in flow: From individual dynamics to rheology. *Comptes Rendus Physique*, 10(8):775 – 789, 2009. Complex and biofluidsFluides complexes et biologiques.

[55] J. Walter, A.-V. Salsac, D. Barthès-Biesel, and P. Le Tallec. Coupling of finite element and boundary integral methods for a capsule in a stokes flow. *International*

Journal for Numerical Methods in Engineering, 83(7):829–850, 2010.

[56] C. Wang, B.C. Khoo, and K.S. Yeo. Elastic mesh technique for 3d {BIM} simulation with an application to underwater explosion bubble dynamics. *Computers Fluids*, 32(9):1195 – 1212, 2003.

[57] Alireza Yazdani and Prosenjit Bagchi. Three-dimensional numerical simulation of vesicle dynamics using a front-tracking method. *Phys. Rev. E*, 85:056308, May 2012.

[58] Hong Zhao, Amir H.G. Isfahani, Luke N. Olson, and Jonathan B. Freund. A spectral boundary integral method for flowing blood cells. *Journal of Computational Physics*, 229(10):3726 – 3744, 2010.

[59] Hong Zhao and Eric S. G. Shaqfeh. The dynamics of a vesicle in simple shear flow. *Journal of Fluid Mechanics*, 674:578–604, 5 2011.

[60] Hong Zhao and Eric S. G. Shaqfeh. The shape stability of a lipid vesicle in a uniaxial extensional flow. *Journal of Fluid Mechanics*, 719:345361, Mar 2013.

[61] Hong Zhao, Andrew P. Spann, and Eric S. G. Shaqfeh. The dynamics of a vesicle in a wall-bound shear flow. *Physics of Fluids*, 23(12), 2011.

[62] Alexander Z. Zinchenko and Robert H. Davis. Emulsion flow through a packed bed with multiple drop breakup. *Journal of Fluid Mechanics*, 725:611663, Jun 2013.

[63] Alexander Z. Zinchenko, Michael A. Rother, and Robert H. Davis. A novel boundary-integral algorithm for viscous interaction of deformable drops. *Physics of Fluids*, 9(6):1493–1511, 1997.