Fully implicit Finite Element method for the modeling of free surface flows with surface tension effect

A. Laadhari*, G. Székely

Institut für Bildverarbeitung, Department of Information Technology and Electrical Engineering, Swiss Federal Institute of Technology Zürich (Eidgenössische Technische Hochschule Zürich), CH-8092 Zürich, Switzerland

SUMMARY

A new approach based on the use of the Newton and level set methods allows to follow the motion of interfaces with surface tension immersed in an incompressible Newtonian fluid. Our method features the use of a high order fully implicit time integration scheme that circumvents the stability issues related to the explicit discretization of the capillary force when capillary effects dominate. A strategy based on a consistent Newton-Raphson linearization is introduced, and performances are enhanced by using an exact Newton variant which guarantees a third-order convergence behavior without requiring second order derivatives. The problem is approximated by mixed finite elements, while the anisotropic adaptive mesh refinements enable us to increase the computational accuracy. Numerical investigations of the convergence properties and comparisons with benchmark results provide evidence regarding the efficacy of the methodology. The robustness of the method is tested with respect to the standard explicit method, and stability is maintained for significantly larger time steps compared to those allowed by the stability condition. Copyright © 2016 John Wiley & Sons, Ltd.

KEY WORDS: Free surface flow; Surface tension; Cubically-convergent Newton method; Finite element method; Anisotropic mesh adaptation; Adaptive time stepping.

1. INTRODUCTION

This framework is concerned with the numerical modeling of the dynamics of interfacial flows with surface tension in a surrounding incompressible Newtonian fluid. The influence of the interfacial
surface tension is significant in several scientific, engineering and geophysical applications such as ink-jet printing [1], atomization processes [2] and gases transfer between the atmosphere and seas [3].

Over the past decade, a multitude of computational strategies have been dedicated to this problem [4, 5, 6, 7, 8, 9]. Typically, they can be roughly sorted according to the way they track the interface motion. We can distinguish between the interface tracking approaches and the interface capturing approaches.

Regarding the first group, a moving mesh explicitly fits the fluid interface and follows its movement. This Lagrangian representation results in separate conservation equations for the fluid inside and outside the interface, while the interfacial condition allows to set the appropriate boundary conditions. For a detailed description of such an Arbitrary Lagrangian Eulerian (ALE) formulation, see e.g. [10, 11]. The boundary integral method, which expresses the flow in terms of distributions of singularities over the interfaces, can also be classified in this category [12].

In the second group, the flow in each fluid domain and on the interface is considered as a single continuum model, inherently taking into account the interfacial condition. In such an Eulerian description, the movement of the interface is followed in an implicit manner. An additional equation describing the transport of the interface is then considered. Several methods are used in the published literature such as the level set method [13, 14], the Volume-of-Fluid method (VOF) [15, 16, 17] and the phase field method [18]. Remark that there exists other hybrid methods between the interface tracking and interface capturing methods, see e.g. [19, 20].

While the Lagrangian approaches features the sharpness of the interface representation, they may exhibit problems related for instance to the large displacement of the interface, the mesh distortion and the changes of topology such as interface merging or breaking. The Eulerian approaches feature the ease of implementation. They avoid problems related to the mesh distortion and naturally handle the topological changes of the interface. Although an additional transport equation of the interface needs to be solved, the corresponding computational cost is in essentially all cases insignificant compared to the cost of solving the Navier-Stokes equations.

In an Eulerian framework, a fixed mesh is usually used along the simulation period. Recently, few mesh adaptation techniques have been employed together with Eulerian methods. In [21, 4], a VOF method combined with an adaptive and structured quad/octree mesh refinement technique is used for the numerical simulations of droplets, showing that the mesh refinement enables more computational accuracy. In [22], adapted unstructured meshes are employed together with the immersed boundary method and the level set method, resulting in a more accurate flow description especially near the wall. In [23], an anisotropic mesh adaptation technique is presented and allows to adapt a general unstructured mesh in the vicinity of the interface. Numerical results in the cases of the Leveque’s test and the mean curvature motion depict that the rate of convergence of the method is improved by the mesh adaptation.

The surface tension is a major component in free surface flows and the numerical coupling of hydrodynamical and capillary effects remains challenging. Indeed, the capillary force lies at the interface and is consequently singular therein. A crucial difficulty is due to the numerical discretization of the surface tension term. For such a problem, the capillary number Ca is used to compare the relative effect of viscous forces to the surface tension acting across the interface between immiscible fluids. Most existing approaches have used explicit decoupling strategies which
are conditionally stable, yielding a severe capillary stability condition for the temporal resolution of type $\Delta t < C/Ca h^{3/2}$ especially for small capillary numbers (i.e. high surface tension) [24], where $h$ stands for the mesh size and $C$ is a constant value. For small velocities, the time step limit is rather given by the viscous time step limit, which corresponds to the characteristic diffusion time. For such a slow flow, this time step limitation of type $\Delta t < C_\nu h^2$, where $C_\nu$ is a constant depending on the kinematic viscosities (ratio of the dynamic viscosity to the density) of both inner and outer fluids, becomes more stringent than the previous convective time step restriction, see e.g. [25, 7]. Therefore, exceeding this time step restriction may lead to an unphysical evolution of the interface or the divergence of the numerical algorithm. Regarding the time discretization, almost exclusively first order schemes for problems of interfaces with capillary force have been used, whereas only few works have used second order schemes such as [26, 27] or third order schemes for Differential-Algebraic formulation of ALE approach [28]. However, no works that used an adaptive time scheme of second order is known to us.

In [29], it was suggested that an implicit treatment of the surface tension would mitigate the time step restriction significantly. Although limited works have been devoted to the time integration strategies, some works introduced the semi-implicit schemes in order to lift the time step restrictions. Following the seminal work of Dziuk [30], Hysing described a semi-implicit method for the treatment of the surface tension term by introducing an additional implicit term that represents a diffusion of velocities tangential to the interface and induced by the surface tension [31]. The method was implemented in a finite element framework, and results show that the method proved to remain stable up to almost 80 times the capillary time step constraint. In [32], the previous method was implemented in a VOF-based finite volume framework, showing that this time step restriction can be exceeded by at least a factor of five without destabilizing the numerical solution. In [33], a fully coupled numerical framework on arbitrary meshes in a finite volume framework is presented, including a novel method to evaluate the curvature from volume fractions. More stability is observed for larger density ratios.

In the present paper we intend to give a new computational way to solve the time dependent coupled Navier-Stokes and level set equations involving interfacial flows with surface tension. The proposed method can be somehow placed in the interface capturing group, as we use a fully Eulerian description of the interface in which the interface condition is incorporated by the level set method. A second order scheme with adaptive time step sizes is introduced. We describe two fully implicit and monolithic approaches where we attempt to solve the entire coupled system using Newton-type strategies. To the knowledge of the authors, the Newton-Raphson linearization of this kind of problems is a novel concept. Our approach avoids the usual stability issues and time step limitations characterizing explicit strategies. Indeed, we achieve that stability is maintained for considerably larger time steps than those predicted by the explicit surface tension time step constraint [24]. Two variants of the exact Newton method featuring second-order and third-order convergence behaviors, respectively, are described. The quantitative comparison of their performances is presented subsequently. Our method is entirely based on a finite element discretization on arbitrary unstructured meshes, whereas an anisotropic metric-based mesh adaptation procedure allows to refine and enhance the computational accuracy on the interface and in zones with complex flow patterns. The present fully-coupled implicit strategy is used to study and compare the maximum temporal resolution with respect to an explicit method, showing that the
capillary time step constraint can be significantly exceeded. We numerically validate our framework in the context of the rising bubble benchmark presented in [34]. The method is further tested in the case of the oscillating two-dimensional bubble and in a simple three-dimensional case.

We have arranged the remainder of this paper as follows. Section 2 outlines the mathematical model, including the level set formulation and the Navier-Stokes equations governing the fluid dynamics. The derivation of the tangent problem and the linearization procedure are presented in Section 3, where we also present two alternative Newton strategies. We also provide details about the numerical discretization and the mesh adaptive procedure. In Section 4 various numerical simulations are performed to illustrate the efficiency and robustness of the proposed method. We close with some remarks and current extensions in Section 5.

2. GOVERNING EQUATIONS

Let $T > 0$ represent the period of the experiment. For any time $t \in (0, T)$, let $\Omega(t) \subset \mathbb{R}^d$, $d = 2, 3$ denote the internal domain having a Lipschitz continuous boundary $\Gamma(t) = \partial \Omega(t)$. The interface $\Gamma(t)$ is embedded in a larger computational domain $\Lambda$ such that $\Gamma(t) \cap \partial \Lambda = \emptyset$, for all $t \in (0, T)$. Let $\mathbf{n}$ and $\mathbf{v}$ denote the unit outer normal vector on the interface $\Gamma(t)$ and the external boundary $\partial \Lambda$, respectively. Let $\mathbf{t}$ be a unit tangent vector on the boundary $\partial \Lambda$. We introduce the mean curvature $H = \text{div} \mathbf{n}$ as the sum of the principle curvatures on $\Gamma(t)$.

From now, the explicit dependance of $\Omega$ and $\Gamma$ from $t$ will be understood.

2.1. Level set formulation

An Eulerian framework is adopted to follow the interfacial motion in such a way that $\Gamma$ represents an iso-surface $\varphi = 0$ of a level set function $\varphi$. A time-dependent partial differential equation, initialized by a signed distance function $\varphi(t = 0)$ to $\Gamma(0)$, describes its motion

$$\partial_t \varphi + \mathbf{u} \cdot \nabla \varphi = 0 \quad \text{in } (0, T) \times \Lambda.$$ 

Geometrical quantities, e.g. $\mathbf{n}$ and $H$, are encoded in terms of $\varphi$ and are consequently extended to the entire domain $\Lambda$. It is well-known that the signed distance property is lost after the advection of $\varphi$, giving sometimes rise to difficult scenarios where $|\nabla \varphi|$ becomes very large or very small near $\Gamma$ and deteriorating the accuracy of computations over $\Gamma$. To reestablish the signed distance property, a redistancing problem is commonly solved to initialize the level set function as a signed distance function [35]. However, it has been revealed in the literature that the zero level set may be unphysically shifted during the redistancing process leading to substantial errors [36]. To alleviate this difficulty, several approaches were developed based on some mesh refinement techniques [37, 38] or by introducing explicit forcing term to fix the iso-surface $0$ during the redistancing process [39, 40, 41]. In the present work, we use the redistancing approach presented in [42, 43] where a Lagrange multiplier allows to enforce a zero-displacement on the interface $\Gamma$. Let $\tau$ be a pseudo-time variable. At every time $t \in (0, T)$, we initialize $\phi(\tau = 0; \cdot; t) = \varphi(t, \cdot)$ and we
solve until convergence the problem:

$$\partial_{\tau} \phi(\tau, \cdot; t) + s_\varepsilon(\varphi(t, \cdot)) \left| \nabla \phi(\tau, \cdot; t) \right| = s_\varepsilon(\varphi(t, \cdot)) + \lambda(\varphi(t, \cdot); t), \quad \text{in } (0, +\infty) \times \Lambda,$$

where $s_\varepsilon(\cdot)$ is a regularized sign function that helps us impose a zero-displacement on $\Gamma$, while $\lambda$ is a forcing term that prevents the unphysical shifting of $\Gamma$ during the redistancing process. A full description of the approach is provided in [43]. We thereafter update $\varphi(t, \cdot)$ with $\phi(\infty, \cdot; t)$.

Let $\varepsilon$ be a regularization parameter proportional to the local mesh size $h$. The sharp Heaviside function $\mathcal{H}$, the Dirac measure $\delta_{\Gamma}$ and the sign function $s$ are regularized within a banded strip of width $2\varepsilon$ such that

$$\mathcal{H}_\varepsilon(\varphi) = \frac{1}{2} \left( 1 + \frac{\min(\max(\varphi, 0), \varepsilon)}{\varepsilon} + \frac{1}{\pi} \sin \left( \frac{\pi \min(\max(\varphi, 0), \varepsilon)}{\varepsilon} \right) \right),$$

$$\delta_{\varepsilon}(\varphi) = \frac{d}{d\varphi} \mathcal{H}_\varepsilon(\varphi) \quad \text{and} \quad s_\varepsilon(\varphi) = 2 \mathcal{H}_\varepsilon(\varphi) - 1.$$

In such an Eulerian-based framework, all integrals over $\Gamma$ are transformed into integrals over $\Lambda$. For any given function $\eta(\cdot)$ defined on $\Gamma$, let $\tilde{\eta}(\cdot)$ be an extension over the domain $\Lambda$. We have

$$\int_{\Gamma} \eta(x) \; ds = \int_{\Lambda} |\nabla \varphi| \delta_{\Gamma}(\varphi) \; dx = \int_{\Lambda} |\nabla \varphi| \delta_{\varepsilon}(\varphi) \; \tilde{\eta}(x) \; dx. \quad (2.1)$$

### 2.2. Statement of the nonlinear capillary problem

To model the surface tension effects in the case of immiscible and incompressible fluids, we consider the instationary Navier-Stokes equations with a free capillary boundary. We assume constant fluid density $\rho_{i/o}$ and viscosity $\mu_{i/o}$ in both sides of the interface $\Gamma$, where the subscripts “$i$” and “$o$” stand for the internal and external domains, respectively. The regularized global viscosity and density functions are given by:

$$\mu_\varepsilon(\varphi) = \mu_i + (\mu_o - \mu_i) \mathcal{H}_\varepsilon(\varphi) \quad \text{and} \quad \rho_\varepsilon(\varphi) = \rho_i + (\rho_o - \rho_i) \mathcal{H}_\varepsilon(\varphi).$$

Let $\sigma(u, p, \varphi) = 2\mu(\varphi)D(u) - p \mathbf{Id}$ and $D(u) = \left( \nabla u + \nabla u^T \right)/2$ be the fluid Cauchy stress tensor and the strain tensor, respectively. Let $[\cdot]^-$ design the jump across $\Gamma$ and $\gamma$ be the surface tension coefficient. The capillary problem is characterized by the continuity of the velocity across the interface, while the stress discontinuity across the interface is calibrated according to the membrane tension. We also introduce an external body force $g$.

We first present the non-dimensionalized problem. Let us introduce the Reynolds number $\text{Re} = \rho_o U_g D/\mu_o$, where $D$ designs the diameter of the interface at $t = 0$ and $U_g = \sqrt{Dg}$ represents the velocity induced by the external force. We also consider the Eötvös number $\text{Eo} = \rho_o U_g^2 D/\gamma$, which compares the gravitational forces to the surface tension effects. The capillary number is given by the ratio $\text{Ca} = \text{Eo}/\text{Re}$. Let us denote by $\mu^* = \mu_i/\mu_o$ and $\rho^* = \rho_i/\rho_o$ the viscosity ratio and the density ratio between both sides of $\Gamma$, respectively. From now, the non-dimensionalized counterpart of the external force is also denoted $g$ and all quantities and domains are non-dimensionalized. In addition, the same notations will be used for ease of exposition.
Let \( \text{div}(\cdot) \) and \( \text{div}(\cdot) \) denote the divergence of tensor fields and vector fields, respectively. The non-dimensionalized problem reads:

\[
\begin{align*}
\mathcal{P}: \text{find the velocity } u &= u(t, x) \text{ and pressure } p = p(t, x) \text{ such that} \\
\text{Re } \rho(\varphi)(\partial_t u + u \cdot \nabla u) - \text{div}(2\mu(\varphi)D(u)) + \nabla p &= \text{Re } \rho(\varphi)g, & \text{in } (0, T) \times \Lambda, \\
\text{div } u &= 0, & \text{in } (0, T) \times \Lambda, \\
[u]_- = 0 \quad \text{and} \quad [\sigma n]_-^+ &= \frac{1}{Ca} H n, & \text{on } (0, T) \times \Gamma.
\end{align*}
\]

### 3. NUMERICAL APPROXIMATION AND IMPLEMENTATION DETAILS

#### 3.1. Semi-discretization in time

Let us divide \([0, T]\) into \(N\) subintervals \([t_n, t_{n+1})\), \(n = 0, \ldots, N - 1\) of variable step size \(\Delta t_n\). We consider an adaptive time stepping scheme to properly ensure the convergence of the Newton subiterations. Indeed, the problem is solved again with a decreased \(\Delta t_n\) by a factor \(1/2\) if the Newton loop does not converge after 6 iterations, whereas we increase \(\Delta t_{n+1}\) for the next time step with an amplification factor 1.2 if the convergence holds. For any \(n \geq 1\), the unknowns \(u_n, p_n\) and \(\varphi_n\) at time step \(n\) are computed by induction. We use the adapted backward differentiation scheme of second order, referred to as BDF2, to approximate the time derivative terms \([44\). The scheme is bootstrapped by the initial conditions \(u_{-1} = u_0 = u(0)\) and \(\varphi_{-1} = \varphi_0 = \varphi(0)\), where \(u_{-1}\) and \(\varphi_{-1}\) only represent suitable notations. Let us introduce the ratio \(\vartheta_n = \Delta t_n / \Delta t_{n-1}\).

Concerning the boundary conditions for the velocity, let \(\Sigma_D \in \partial \Lambda\) be the subset on which essential boundary conditions are assigned, while free stresses are imposed on the remaining boundary \(\partial \Lambda / \Sigma_D\). We introduce the functional spaces:

\[
\nabla(u_b) = \left\{ v \in \left(H^1(\Lambda)\right)^d : v = u_b \text{ on } \Sigma_D \right\} \quad \text{and} \quad Q = \left\{ q \in L^2(\Lambda) : \int_{\Omega} q \, dx = 0 \right\}.
\]

Let \(\Sigma_- = \left\{ x \in \partial \Lambda : u \cdot \nu(x) < 0 \right\} \) represent the upstream boundary. We introduce the space of admissible level set functions:

\[
\mathcal{X}(\varphi_b) = \left\{ \psi \in W^{1,\infty}(\Lambda) \cap H^1(\Lambda) : \psi = \varphi_b \text{ on } \Sigma_- \right\}.
\]

At every \(t_n\), \(\varphi_b = \varphi_{n-1}\) is the steady-state solution of the redistancing problem and approximates a signed distance function; it allows to set the boundary condition for \(\varphi_n\) on \(\Sigma_-\). Considering the adaptive advancing formula of the BDF2 \([44\), the semi-discrete approximation in time of the
capillary problem $\mathcal{P}$ reads:

$$
\mathcal{P}_n: \text{find } (u_n, p_n, \varphi_n) \in V(u_0) \times Q \times \mathcal{X}(\varphi_{n-1}) \text{ such that }
\begin{align*}
\text{Re} \int_{\Omega} \rho(\varphi_n) \frac{(1 + 2 \theta_n) u_n - (1 + \theta_n)^2 u_{n-1} + \theta_n^2 u_{n-2}}{(1 + \theta_n) \Delta t_n} \cdot v + \text{Re} \int_{\Omega} \rho(\varphi_n) (u_n \cdot \nabla u_n) \cdot v
+ \int_{\Omega} 2 \mu(\varphi_n) D(u_n) : D(v) - \int_{\Omega} p_n \text{div} v - \frac{1}{Ca} \int_{\Gamma_n} H_n (\nabla \varphi_n / |\nabla \varphi_n|) \cdot v - \text{Re} \int_{\Omega} \rho(\varphi_n) g \cdot v = 0,
\end{align*}
$$

(3.1)

$$
\int_{\Omega} q \text{div} u_n = 0,
$$

(3.2)

$$
\int_{\Omega} (1 + 2 \theta_n) \varphi_n - (1 + \theta_n)^2 \varphi_{n-1} + \theta_n^2 \varphi_{n-2} \psi + \int_{\Omega} (u_n \cdot \nabla \varphi_n) \psi = 0,
$$

(3.3)

for all test functions $v \in V(0)$, $q \in Q$ and $\psi \in \mathcal{X}(0)$.

Following the approach described in [43, Section 3.3.2], we additionally consider an a posteriori mass correction term that enforces at every time step the condition $|\Omega_n| = |\Omega|_{\text{exact}}$.

Regarding the redistancing problem, we solve it at every time step after solving the system (3.1-3.2-3.3). At the numerical level, we use a first order combined characteristics and finite difference discretization method to approximate the advection term. In addition, we use the Gauss-Lobatto quadrature formula that guarantees further stability for the characteristics method [45]. Further details about the numerical scheme and the method of characteristics are available in [46, 47, 48, 43].

### 3.2. Consistent linearization and Newton-Raphson method

For a given functional $\mathcal{F}(\varphi(x))$, let $D\mathcal{F}(\varphi)[\delta \varphi]$ denote the Gâteaux derivative of $\mathcal{F}$ at $\varphi$ along the direction $\delta \varphi$. Let $\nabla_s = (\text{Id} - n \otimes n)\nabla = \nabla - n \cdot (n \cdot \nabla)$ denote the surface gradient operator, and $\otimes$ represent the tensor product between vectors.

We first provide some useful directional derivatives in the direction of the level set increment $\delta \varphi$:

$$
D\nabla \varphi[\delta \varphi] = \nabla \delta \varphi,
$$

$$
D\nabla / \nabla |\varphi| = D\sqrt{\nabla \varphi \cdot \nabla \varphi} [\delta \varphi] = \nabla \delta \varphi \otimes n,
$$

$$
D1 / |\nabla \varphi| = -\nabla \delta \varphi \cdot \nabla \varphi / |\nabla \varphi|^3,
$$

$$
Dn[\delta \varphi] = \nabla \delta \varphi / |\nabla \varphi| - \frac{(\nabla \varphi \cdot \nabla \delta \varphi)}{|\nabla \varphi|^3} \nabla \varphi = \nabla \delta \varphi / |\nabla \varphi| - \frac{(n \cdot \nabla \delta \varphi) n}{|\nabla \varphi|} = \frac{\nabla s \delta \varphi}{|\nabla \varphi|},
$$

$$
D\mu_c(\varphi)[\delta \varphi] = (1 - \mu^*) \delta_s(\varphi) \delta \varphi
$$

and

$$
D\rho_c(\varphi)[\delta \varphi] = (1 - \rho^*) \delta_c(\varphi) \delta \varphi.
$$

Let us introduce the mixed variable $\Psi \equiv H$, and we proceed with the computation of $\Psi$ in the weak sense to derive the derivation order. We assume that $\Gamma$ never touches the external boundary $\partial \Omega$.

Remark that the evaluation of $\Psi$ is only needed in a surrounding of the interface $\Gamma$. Consequently, an accurate computation of $\Psi$ near $\Gamma$ is only requested. By using the Green transformation, the boundary term vanishes and we introduce the residual $\mathcal{R}_\Psi$ as follows:

$$
\mathcal{R}_\Psi(\Psi, \varphi) = \int_{\Omega} \Psi \xi + \int_{\Omega} \frac{\nabla \varphi}{|\nabla \varphi|} \cdot \nabla \xi = 0, \quad \forall \xi \in H^1_0(\Omega),
$$

(3.4)

This article is protected by copyright. All rights reserved.
where the space $H^{-1}(\Lambda)$ represents the dual space of $H^1_0(\Lambda)$. After the linearization with respect to the variables $\varphi$ and $\Psi$, we obtain the following equation coupling the increments $\delta \Psi$ and $\delta \varphi$:

$$
\int_\Lambda \delta \Psi \xi + \int_\Lambda \frac{\nabla \delta \varphi}{|\nabla \varphi|} \cdot \nabla \xi = -\left\{ \mathcal{R}_\Psi(\Psi, \varphi), \xi \right\}_{H^{-1},H^1_0}, \quad \forall \xi \in H^1_0(\Lambda).
$$

Let us introduce the following weighted multiple linear forms:

$$
d(a, u; v) = \int_\Lambda 2\psi \mathbf{D}(u) : \mathbf{D}(v), \quad e(a, v; q) = -\int_\Lambda q \text{div} \, u, \quad d(\varphi, v; w) = \int_\Lambda w \nabla \varphi \cdot v,
$$

$$
e(\varphi, v; q) = \int_\Lambda q \text{div} \, v, \quad g(\varphi, v; w) = \int_\Lambda \varphi v \cdot w, \quad m(u, v; w) = \int_\Lambda u \cdot v,
$$

$$
i(\varphi, v; w) = \int_\Lambda \psi \nabla \varphi \quad \text{and} \quad h(\varphi, v; w) = \int_\Lambda 2\varphi w \mathbf{D}(v) : \mathbf{D}(w),
$$

defined for all $u, v, w \in (H^1(\Lambda))^d$; $q \in L^2(\Lambda); w \in L^\infty(\Lambda); \varphi, \psi \in H^1(\Lambda)$ and $\mathbf{T} \in (L^\infty(\Lambda))^{d \times d}$. The mixed problem is given by the system (3.1-3.2-3.3-3.4). Let $\Xi_n \equiv (u_n, p_n, \Psi_n, \varphi_n)^T$ and $\mathcal{R}(\Xi_n)$ be the corresponding vector of unknowns and the global residual, respectively. The Newton-Raphson method reduces the nonlinear problem $\mathcal{P}_n$ into a sequence of linear sub-problems [49]. Thereafter, we drop the subscript $n$ whenever it is contextually clear. After linearization with respect to $\Xi_n$, the tangent system reads:

$$
\text{Given } \Xi^k, \text{ find } \delta \Xi^k = (\delta u^k, \delta \varphi^k, \delta \Psi^k, \delta \varphi^k) \in \mathcal{V}(u_n) \times \mathcal{Q} \times H^1(\Lambda) \times \mathcal{X}(\varphi_n) \text{ such that }
$$

$$
\text{Re}(1 + 2\partial_\varphi) \mathcal{M}(\delta u^k, v; \rho \varepsilon(\varphi^k)) + \text{Re}(\partial_\varphi) \mathcal{M}(\delta u^k, v; \rho \varepsilon(\varphi^k)) + \partial(a(\delta u^k, v; \mu \varepsilon(\varphi^k))
$$

$$
+ \text{Re}(1 - \rho^*) g(\delta \varphi^k, v; \varepsilon(\varphi^k) \left(\frac{(1 + 2\partial_\varphi)u^k - (1 + \partial_\varphi)u_{n-1} + \partial_\varphi u_{n-2} + u^k \cdot \nabla u^k)}{(1 + \partial_\varphi)\Delta t_n}\right))
$$

$$
- \frac{1}{C_a} g(\delta \Psi^k, v; \varepsilon(\varphi^k) \nabla \varphi^k) - \frac{1}{C_a} g(\delta \varphi^k, v; \Psi^k \varepsilon(\varphi^k) \nabla \varphi^k) - \frac{1}{C_a} g(\delta \varphi^k, v; \Psi^k \varepsilon(\varphi^k) \nabla \varphi^k)
$$

$$
+ (1 - \mu) h(\delta \varphi^k, v; \varepsilon(\varphi^k), u^k) - \text{Re}(1 - \rho^*) g(\delta \varphi^k, v; \varepsilon(\varphi^k) g)
$$

$$+ b(v, \delta \varphi^k) = -\left\{ \mathcal{R}_\Xi(\Xi^{T,k}), v \right\}_{\mathcal{V}(0), \mathcal{V}(0), \mathcal{Q}(0), \mathcal{Q}(0)), \mathcal{X}(0), \mathcal{X}(0)}, \quad (3.5)
$$

$$
b(\delta u^k, q) = \left\{ \mathcal{R}_p(u_k), q \right\}_{\mathcal{Q}(0), \mathcal{Q}(0)). \quad (3.6)
$$

$$
e(\delta \varphi^k, \xi; 1) + f(\delta \varphi^k, \xi; \nabla \varphi^k)^{-1}(\mathbf{I} - n^k \otimes n^k) = -\left\{ \mathcal{R}_\Psi(\Psi^k, \varphi^k), \xi \right\}_{H^{-1},H^1_0}, \quad (3.7)
$$

$$
e(\varphi^k, \psi; 1 + 2\partial_\varphi) + i(\delta \varphi^k, \psi; u^k) = g(\psi, \delta u^k; \nabla \varphi^k) = -\left\{ \mathcal{R}_\varphi(\varphi^k, u^k), \psi \right\}_{\mathcal{X}(0), \mathcal{X}(0)}, \quad (3.8)
$$

Accepted Article

This article is protected by copyright. All rights reserved.
for all \((v, q, \xi, \psi) \in V(0) \times Q \times H^1_0(\Lambda) \times \mathcal{X}(0)\), where the corresponding residuals express as:

\[
\begin{align*}
\langle A_{\xi}(\xi^T, k), v \rangle_{V(0) \times V(0)} &= a(u^k, v; \mu \xi(\phi^k)) + b(v, q^k) - \frac{1}{Ca} d(\phi^k, v; \Psi \xi \delta \xi(\phi^k)) \\
&+ \frac{\text{Re}}{1 + \partial_n^2} m((1 + \partial_n)\Delta t_n) (u^k - \partial_n u_{n-1} + \partial_n^2 u_{n-2}, v),
\end{align*}
\]

\[
\begin{align*}
&+ \frac{\text{Re}}{2} \left( u^k, v; \rho \xi(\phi^k), u^k \right) - \text{Re} \left( g, v; \rho \xi(\phi^k) \right),
\end{align*}
\]

\[
\begin{align*}
\langle A_p(u^k), q \rangle_{P(0) \times P(0)} &= b(u^k, q),
\end{align*}
\]

\[
\begin{align*}
\langle A_{|v|^2}(\phi^k, u^k), \psi \rangle_{X(0) \times X(0)} &= c \left( (1 + \partial_n)\varphi^k - \partial_n \varphi_{n-1} + \partial_n^2 \varphi_{n-2}, \psi; 1 \right) + d(\phi^k, u^k; \psi),
\end{align*}
\]

\[
\begin{align*}
\langle A_{\Psi}(\Psi, \phi^k), \xi \rangle_{H^1 \times H^1} &= e \left( \Psi^k, \xi; 1 \right) + f(\phi^k, \xi; |\nabla \phi^k|^{-1} \text{Id}).
\end{align*}
\]

Given the solutions at \(t < t_n\), we iteratively compute \(\Xi_n^k\) at \(t_n\) by using two strategies that enable us to compute the increments \(\delta \Xi_n^k = (\delta u_n^k, \delta p_n^k, \delta \Psi_n^k, \delta \phi_n^k)\). For any sub-iteration \(k \geq 1\), we consider:

(i) Strategy I is the classical quadratically convergent Newton method and it consists in solving

\[
\left\{ D\mathcal{R}(\Xi_n^k) \left[ \delta \Xi_n^k \right], \xi \right\} = - \left\{ \mathcal{R}(\Xi_n^k), \xi \right\}, \forall \xi.
\]

We explicitly update afterwards the solution as follows: \(\Xi_n^{k+1} = \Xi_n^k + \delta \Xi_n^k\).

(ii) Strategy II is a formulation in the multidimensional case of the method introduced by Kou et al. [50]. It features a third-order convergence behavior in some neighborhood of the solution without requiring the evaluation of the second derivatives of the residual. The increment is computed at each Newton iteration in two steps as follows:

\[
\begin{align*}
(1) \quad \Xi_n^{k+0.5} &= \Xi_n^k + \delta \Xi_n^k \quad \text{with} \quad \left\{ D\mathcal{R}(\Xi_n^k) \left[ \delta \Xi_n^{k} \right], \xi \right\} = \left\{ \mathcal{R}(\Xi_n^k), \xi \right\}, \forall \xi, \\
(2) \quad \Xi_n^{k+1} &= \Xi_n^{k+0.5} + \delta \Xi_n^{k+0.5} \quad \text{with} \quad \left\{ D\mathcal{R}(\Xi_n^k) \left[ \delta \Xi_n^{k+0.5} \right], \xi \right\} = - \left\{ \mathcal{R}(\Xi_n^{k+0.5}), \xi \right\}, \forall \xi.
\end{align*}
\]

Observe that, for Strategy II, the minus sign appears in front of the residual only in the second step.

The efficiency index is an indicator commonly used to compare the Newton variants; it measures the corresponding order of convergence per residual or Jacobian evaluation. However, we notice that this index does not represent always the best way of comparison in the multidimensional case, as it does not account for the differences in the computational cost of the Jacobian assembly, Jacobian factorization and residual evaluation. Strategy II requires two evaluations of the residual and one assembly of the Jacobian, yielding to an efficiency index of \(\sqrt{3}\). This efficiency index is better than the one of Strategy I given by \(\sqrt{2}\), see [50]. The Kou’s method bases on a modification of the Werrakoon-Fernando method [51] and its efficiency compared to the main existing third-order methods is demonstrated in [50] for several basic mono-dimensional test cases. A similar cubically convergent method was subsequently developed in [52]. The later strategy doesn’t require the evaluation of the second derivative, and needs one evaluation of the residual and two evaluations of the Jacobian per iteration.

To compare the computational cost of both residual’s and Jacobian’s assembly, we proceed with the assembly of the linear system using a structured mesh having 75′600 triangles and 29′600 vertices. The computational cost of the residual’s assembly is 20.7138 s, whereas the computational cost of the Jacobian’s assembly is equal to 39.1561 s, and the cost of the Jacobian factorization is 35.7054 s. Hence, the evaluation of the residual is significantly cheaper than the assembly and factorization evaluation.

This article is protected by copyright. All rights reserved.
of the Jacobian matrix. It is more beneficial to consider a method that only needs to assemble and factorize once the Jacobian at each Newton iteration. Consequently, we opt to use the Kou’s method developed in [50] for the capillary problem among other cubically convergent method, and we consider the aforementioned generalization of this method to the multidimensional case.

Furthermore, we use a second order extrapolation of the solution at previous times to assign the starting values of the velocity and level set at each Newton loop. For practical computations, the stopping criterion is based on the evaluation of the global residual, and we set the Newton tolerance to $10^{-9}$ in our computations.

3.3. Space discretization and banded level set approach

We consider a partition $\mathcal{T}_h$ of $\Lambda$ consisting of geometrically conforming open simplicial elements $K$, i.e. triangles for $d = 2$ and tetrahedra for $d = 3$, such that $\Lambda = \bigcup_{K \in \mathcal{T}_h} K$. Regarding the finite dimensional spaces, we consider a Taylor-Hood finite element approximation for the discretization of velocity and pressure, see e.g. [53], while we use the $P_2$ Lagrange finite elements to approximate the level set function $\varphi$ and the mixed variable $\Psi$.

We now focus on the resolution of the Jacobian system (3.5-3.6-3.7-3.8) arising from the linearization of (3.1-3.2-3.3-3.4) after time and space discretization. The pressure is only computed up an arbitrary constant, and the global matrix of the linear system is then singular. To solve the discretized problem using a direct method, we can use, for instance, the procedure that consists in adding a real Lagrange multiplier which imposes a constant average pressure value [46]; that corresponds to adding one line in the global linear system. Otherwise, we can fix the pressure value on one degree of freedom before solving the linear system.

The linear system corresponding to the discretized problem (3.5-3.6-3.7-3.8) has a sparse and block-structure. We use the package Mumps\textsuperscript{1} for the factorization and as direct solver on distributed memory architectures. This package provides a direct method using the multifrontal method that represents a version of the Gaussian elimination for large sparse systems of equations with either symmetric or unsymmetric matrices [54]. We refer to [54, 55, 56, 57] for detailed descriptions of the

---

\textsuperscript{1}MUMPS – http://graal.ens-lyon.fr/MUMPS

This article is protected by copyright. All rights reserved.
methods and algorithms used. For the tridimensional problem, iterative methods using for instance the preconditioned conjugate gradient algorithm can also be used [45].

We thereafter give further insight into the structure of the matrix of the linear system corresponding to (3.5-3.6-3.7-3.8), and we show how the size of the linear system can be reduced. Indeed, the capillary term in the momentum equation is only localized in the vicinity of the interface $\Gamma$, and we only need to evaluate the mixed variable $\Psi$ in a small surrounding of $\Gamma$. Consequently, we only assemble the equation (3.7) in a banded strip of width $\epsilon'$ around the interface, with $\epsilon'$ is parameter proportional to the mesh size. We commonly choose $\epsilon' = 4h$ in our computations. That helps to rule out the unneeded computational effort. The banded strip is given by:

$$B_{\epsilon'}(t) = \{ K \in \mathcal{T}_h : \delta_{\epsilon'}(\varphi) \neq 0 \}.$$  

Only the coefficients that correspond to these elements are then considered in the global matrix. To graphically show the size reduction of the linear system, we use the Matrix-Market format‡ (ASCII-based) to display the sparse and block-structured Jacobian matrix. This format is elaborated in such a way that only the non-zero entries are encoded, and the corresponding coordinates are explicitly stored. We thereafter convert into the Matlab sparse format§ to graphically visualize the sparsity pattern. Let us consider a two-dimensional case and a structured mesh of 33'836 triangular elements.

We first consider the equation (3.7) in the entire $\Lambda$ and we assemble the matrix of the corresponding linear system. The corresponding matrix displayed in Fig. 1(left) shows a sparse block-structure and it has the size $288'489^2$. By restricting (3.7) in the banded domain $B_{\epsilon'}$, the matrix size is reduced to 223'338², see Fig. 1(right). A significant reduction of the size of the linear system is accordingly obtained and corresponds to almost 22.6% in the number of rows and columns.

3.4. Anisotropic mesh adaptation

We use the bidimensional anisotropic mesh generator BAMG based a Delaunay-type triangulation [58, 59]. We use a metric-based approach and we follow the mesh adaptation procedure described in [43, 23]. However, we consider a different meshing criterion more appropriate to the present capillary problem. Compared to our previous work [23], the mesh adaptation procedure also aims to refine the mesh in areas of less regular velocity, whereas it must coarsen the mesh in zones of regular velocity, even coarser than the initial mesh.

Given the solution $\Xi_n$ at time step $t^n$, we shall consider a meshing criterion $\chi(\Xi_n)$. We introduce the metric tensor given by the Hessian matrix of the previous scalar field: $\nabla^2 \chi(\Xi_n)$. For each triangle $K$ in the mesh $\mathcal{T}_h$, a singular value decomposition of the Jacobian of the affine transformation that maps a reference equilateral triangle into $K$ is needed. An adapted mesh is then generated by shrinking all triangles in both eigenvectors’ directions of $\nabla^2 \chi(\Xi_n)$, while we adjust the triangle’s sizes in these directions such that the interpolation error becomes equidistributed.

To increase the computational accuracy near the interface and in zones where more complex flow

§Matlab – http://ch.mathworks.com/
patterns (such as recirculation zones / vortices) occur, we introduce the following criterion:

$$\chi(\Xi) = \left( \left( \rho^* + (1 - \rho^*) H \phi \right) \|u\|^2 + 4 \left( \mu^* + (1 - \mu^*) H \phi \right) \|D(u)\|^2 \right)^{1/2}.$$

We tune the parameters of the mesh adaptation, such as the maximum anisotropy ratio, the smallest edge’s length and the maximal aspect ratio of all mesh elements $K \in \mathcal{T}_h$, to generate either almost isotropic or anisotropic meshes, see snapshots in Fig. 5 and Fig. 9 and Movies (1-2-3-4-5) in the supplementary material. In the numerical examples, we use moderately anisotropic meshes unless otherwise stated.

![Figure 2. Example 1: (left) Temporal evolution of the area. (middle) Comparison of the final shapes with the reference solutions in [34]. (right) A closed-view in zones of maximal discrepancies between the different shapes. The Same color code as for the global view is used.](image)

4. NUMERICAL EXAMPLES

In what follows, we provide a set of numerical examples in both two-dimensional and three-dimensional cases to address the main features of the proposed method.

**Software implementation** The presented method has been implemented using the Rheolef [45] environment. It is a general purpose C++ library for scientific computing, with special emphasis on finite elements and parallel computation. Distributed-memory parallelism is performed via MPI\(^\text{¶}\). The Scotch library is used for distributed mesh partitioning\(^\text{¶¶}\). The library relies upon the Boost\(^\text{**}\), Blas\(^\text{††}\) and UMFPACK\(^\text{‡‡}\) libraries for much of its functionalities. The different meshes have been generated using GMSH\(^\text{§§}\) and BAMG\(^\text{¶¶}\). The computational results are displayed graphically using the software Paraview\(^\text{***}\), while the plots are generated using Gnuplot\(^\text{†††}\).

\(^\text{¶}\)Message Passing Interface – http://www.mpi-forum.org

\(^\text{¶¶}\)Scotch – http://www.labri.fr/perso/pelegrin/scotch

\(^\text{**}\)Boost libraries – http://www.boost.org

\(^\text{††}\)Basic Linear Algebra Subprograms library – http://www.netlib.org/blas

\(^\text{‡‡}\)Umfpack routines – http://www.cise.ufl.edu/research/sparse/umfpack/

\(^\text{§§}\)GMSH – http://www.geuz.org/gmsh

\(^\text{¶¶}\)BAMG – http://people.sc.fsu.edu/ jburkardt/data/bamg/bamg.html

\(^\text{***}\)Paraview – http://www.paraview.org

\(^\text{†††}\)Gnuplot – http://www.gnuplot.info

This article is protected by copyright. All rights reserved.
4.1. Example 1: Numerical validation using the benchmark of the rising bubble dynamics

To proceed with the validation of our numerical method, we consider the quantitative benchmark of the rising bubble dynamics in a heavier fluid introduced by Hysing et al. [34]. For a comparative study, we only consider the set-up of the ellipsoidal bubble where full agreement is obtained by the different codes in [34], while we only show a sample result for the set-up of the skirted bubble and the corresponding adapted mesh in thin filamentary regions. Unless otherwise stated, the numerical results are obtained using Strategy II which behaves generally computationally cheaper than Strategy I. More details about the computational costs will be provided afterwards.

The bubble is initially circular having a radius of \( r_0 = 0.25 \), yielding an area \( |\Omega|_{exact} = \pi^2/16 \) and placed at \( x = (0.5, 0.5) \) in the rectangular computational domain \( \Lambda = [0, 1] \times [0, 2] \). Simulations are performed over the time period \( (0, T) = (0, 3) \). An external gravity force \( g \approx (0, -0.98)^T \) is considered and points toward the bottom of the domain. The physical parameters are given by \( \rho_o = 10^3, \mu_1 = 10^2, \mu_o = 10, \mu_1 = 1 \) and \( \gamma = 24.5, \) which leads to the dimensionless numbers \( \rho^* = 0.1, \mu^* = 0.1, \) \( \text{Re} = 35, \) \( \text{Eo} = 10 \) and \( \text{Ca} = 0.286. \) The no-slip wall conditions are imposed on the horizontal boundaries (i.e. representing \( \Sigma_D \)), while the free slip conditions \( u \cdot \nu = 0 \) and \( t \cdot D(u) \cdot \nu = 0 \) are prescribed on the remaining boundaries.

Let \( u_x \) and \( u_y \) be the velocity components such that \( u = (u_x, u_y) \). Following the benchmark, some quantities of interest are considered and represent the center of mass \( Y_c(t) \), the degree of circularity \( \bar{f}(t) \), the minimum circularity \( \bar{f}_{\min} \) and the corresponding incidence time \( t_{\bar{f}=\bar{f}_{\min}} \), the rise velocity.
\( V_c(t) \), the maximal velocity \( V_{c,\max} \) and the corresponding incidence time \( t|_{V_c=V_{c,\max}} \). We have:

\[
Y_c(t) = \frac{1}{|\Omega|} \int_{\Omega} y \, dx, \quad f(t) = \frac{2\sqrt{\pi|\Omega|}}{|\Gamma|} \quad \text{and} \quad V_c(t) = \frac{1}{|\Omega|} \int_{\Omega} u_y \, bx.
\]

Let \( \text{NTS} \) and \( \text{NV} \) be the numbers of time steps and mesh vertices, respectively. We evaluate the mean mesh size \( \bar{h} \) during the simulation period as the mesh size of a structured triangular mesh having the same \( \text{NV} \) at every time step:

\[
\bar{h} = \frac{1}{\text{NTS}} \sum_{t=0}^{\text{NTS}} \frac{4}{\sqrt{1 + 8\text{NV} - 3}}.
\]

Since that characterizes a structured mesh of similar size, it leads to a problem with the same number of degrees of freedom as the problem solved on the adapted mesh, and consequently having the same size of the linear system.

In Fig. 2 the bubble shapes at the final time \( T = 3 \) with different mesh resolutions are compared with the shapes obtained at \( T = 3 \) in [34] using the finest meshes. A close-up view in the zone of maximal discrepancy is also provided in Fig. 2(right), showing overall a good agreement. The bubble’s area is clearly preserved equal to \( |\Omega|_{\text{exact}} \) during the simulation period, as depicted in Fig. 2(left). Fig. 3 shows that no significant differences can be seen in the temporal evolution of the center of mass \( Y_c \). The curves of the circularity \( f \) agree well for the finest mesh, while only an enlarged section around \( t|_{V_c=V_{c,\max}} \) depicts some differences that are minimized for finer meshes. Similarly, the evolution of the rise velocity in our simulations is agreeing well with the reference results, see Fig. 4(left). The close-up view in Fig. 4(right) clearly shows the agreement, while the complete convergence seems to be achieved. In Fig. 5, we provide some snapshots showing the temporal evolution of the interface during the simulation period and the corresponding adapted meshes. Observe that the initial bubble deforms while staying compact for all \( t \in (0,3) \). During the simulation period, the meshing procedure clearly allows to generate high density meshes around the free interface and in particular areas of high norm of the strain tensor. However, big mesh elements are generated in front of the rising bubble where the velocity field has a more regular pattern. See the movies in the supplementary material.

We now proceed with a quantitative comparison of the above-mentioned benchmark quantities against some computational results available in the published literature using different computational frameworks, as detailed in Hysing et al. [34], Štrubelj et al. [7], Klostermann et al. [9] and Doyeux et al. [8]. Several mesh refinement levels are used, and computational results are

<table>
<thead>
<tr>
<th>( 1/h )</th>
<th>17.5</th>
<th>35.0</th>
<th>72.1</th>
<th>143.4</th>
<th>250.1</th>
<th>Hysing et al.[34]</th>
<th>Štrubelj et al.[7]</th>
<th>Klostermann et al.[9]</th>
<th>Doyeux et al.[8]</th>
</tr>
</thead>
<tbody>
<tr>
<td>( 1/h_{\text{min}} )</td>
<td>25.1</td>
<td>57.9</td>
<td>90.5</td>
<td>195.3</td>
<td>334.4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( f</td>
<td><em>{V_c=V</em>{c,\max}} )</td>
<td>0.9213</td>
<td>0.9079</td>
<td>0.9046</td>
<td>0.9013</td>
<td>0.9013</td>
<td>0.9012 ± 0.0001</td>
<td>0.8876</td>
<td>0.9044</td>
</tr>
<tr>
<td>( t</td>
<td><em>{V_c=V</em>{c,\max}} )</td>
<td>2.0000</td>
<td>1.3928</td>
<td>1.8964</td>
<td>1.8851</td>
<td>1.8835</td>
<td>1.8895 ± 0.0145</td>
<td>1.8915</td>
<td>1.9625</td>
</tr>
<tr>
<td>( Y_c(t=3) )</td>
<td>0.2391</td>
<td>0.2410</td>
<td>0.2418</td>
<td>0.2419</td>
<td>0.2419 ± 0.0002</td>
<td>0.2457</td>
<td>0.2348</td>
<td>0.2412</td>
<td></td>
</tr>
<tr>
<td>( Y_c(t=3) )</td>
<td>0.10357</td>
<td>0.09743</td>
<td>0.09428</td>
<td>0.09290</td>
<td>0.09281</td>
<td>0.09263 ± 0.0050</td>
<td>0.09235</td>
<td>0.09516</td>
<td>0.09248</td>
</tr>
<tr>
<td>( Y_c(t=3) )</td>
<td>1.0918</td>
<td>1.0812</td>
<td>1.0810</td>
<td>1.0809</td>
<td>1.0809</td>
<td>1.0808 ± 0.0009</td>
<td>1.0679</td>
<td>1.0696</td>
<td>1.0815</td>
</tr>
</tbody>
</table>

Table I. Example 1: Comparisons with benchmark quantities and available published results. The results are obtained with the mesh adaptation technique.
Figure 5. Example 1: Snapshots showing the moderately anisotropic adapted meshes, the pressure isocontour lines and the corresponding shapes $\Gamma$ of the rising bubble at times $t = 0.214, 1.178, 1.75, 2.321, 2.571, 3$, respectively. During the simulation period, the meshes are characterized by $1/h \approx 37.1$, $1/h_{\text{min}} \approx 58.8$ and $1/h_{\text{max}} \approx 10$.

provided in Table I showing overall very consistent results. The mean mesh size and the smallest edge of triangles $K \in \mathcal{T}_h$ are also provided.

Similar to [34], we measure the temporal evolution of the different benchmark quantities $q_t$ against a reference solution, referred to as $q_t^r$, obtained with the finest meshes and the smallest time step size. We compute the following relative errors:

$$
|e|_1 = \sqrt{\sum_{t=1}^{\text{NTS}} \frac{|q_t^r - q_t|}{|q_t^r|}}, \quad |e|_2 = \sqrt{\sum_{t=1}^{\text{NTS}} \frac{|q_t^r - q_t|^2}{\sum_{t=1}^{\text{NTS}} |q_t^r|^2}} \quad \text{and} \quad |e|_\infty = \frac{\max_t |q_t^r - q_t|}{\max_t |q_t^r|}.
$$

This article is protected by copyright. All rights reserved.
In addition, we compute the convergence rate ROC for the desired quantities, that evaluates the convergence of the method toward an approximate discrete solution:

\[
\text{ROC}_i = \frac{\log_{10}(||e^{l-1}||_i/||e^l||_i)}{\log_{10}(h^{l-1}/h^l)}, \quad i \in \{1, 2, \infty\},
\]

where \(e^l\) and \(e^{l-1}\) stand for the errors generated on consecutive meshes of sizes \(h^l\) and \(h^{l-1}\), respectively. We consider the standard linear interpolation to account for the differences in the time step sizes.

Let us first consider structured triangular meshes. The relative errors are provided in Table II, showing a convergence rate of about 2 for the circularity in the \(l_1\) and \(l_2\) norms, and a convergence rate of about 1.5 in the \(l_\infty\) norm. The convergence rate of the center of mass approaches 1.5 in all norms. When using the adaptive mesh technique aforementioned, results in Table III suggest that the circularity approaches a convergence order of 2.5 in the \(l_1\) and \(l_2\) norms, while a rate of 2 is observed in the \(l_\infty\) norm. The convergence order of the center of mass is of about 2.5 in all norms.

![Figure 6. Example 1: Convergence curves of the residuals using Strategy I and Strategy II in the semi-logarithmic scale.](image)

<table>
<thead>
<tr>
<th>(1/h)</th>
<th>(|e|_1)</th>
<th>ROC_1</th>
<th>(|e|_2)</th>
<th>ROC_2</th>
<th>(|e|_\infty)</th>
<th>ROC_\infty</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\xi)</td>
<td>39</td>
<td>3.9292E-3</td>
<td>1.603</td>
<td>4.4786E-3</td>
<td>1.812</td>
<td>7.4143E-3</td>
</tr>
<tr>
<td>72</td>
<td>1.1574E-3</td>
<td>1.993</td>
<td>1.3324E-3</td>
<td>1.977</td>
<td>2.6123E-3</td>
<td>1.701</td>
</tr>
<tr>
<td>143</td>
<td>3.2724E-4</td>
<td>1.841</td>
<td>3.7146E-4</td>
<td>1.861</td>
<td>8.4124E-4</td>
<td>1.651</td>
</tr>
<tr>
<td>17</td>
<td>2.0930E-2</td>
<td>2.1738E-2</td>
<td>3.0102E-2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(Y_c)</td>
<td>39</td>
<td>6.8396E-3</td>
<td>1.347</td>
<td>7.1977E-3</td>
<td>1.331</td>
<td>6.7916E-3</td>
</tr>
<tr>
<td>72</td>
<td>2.5769E-3</td>
<td>1.592</td>
<td>2.9968E-3</td>
<td>1.429</td>
<td>2.9706E-3</td>
<td>1.387</td>
</tr>
<tr>
<td>143</td>
<td>9.2256E-4</td>
<td>1.497</td>
<td>1.1577E-3</td>
<td>1.386</td>
<td>1.2075E-3</td>
<td>1.312</td>
</tr>
</tbody>
</table>

Table II. Example 1: Convergence results for the circularity \(\xi\) and the center of mass \(Y_c\). The results are obtained with structured meshes.

<table>
<thead>
<tr>
<th>(1/h)</th>
<th>(|e|_1)</th>
<th>ROC_1</th>
<th>(|e|_2)</th>
<th>ROC_2</th>
<th>(|e|_\infty)</th>
<th>ROC_\infty</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\xi)</td>
<td>17.5</td>
<td>2.5921E-3</td>
<td>2.6717E-3</td>
<td>3.0156E-3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>38.9</td>
<td>7.5355E-4</td>
<td>1.547</td>
<td>7.7084E-4</td>
<td>1.556</td>
<td>8.5744E-4</td>
<td>1.574</td>
</tr>
<tr>
<td>72.1</td>
<td>1.6915E-4</td>
<td>2.421</td>
<td>1.9821E-4</td>
<td>2.201</td>
<td>2.9010E-4</td>
<td>1.756</td>
</tr>
<tr>
<td>143.4</td>
<td>3.1350E-5</td>
<td>2.451</td>
<td>4.5012E-5</td>
<td>2.402</td>
<td>6.5744E-5</td>
<td>2.159</td>
</tr>
<tr>
<td>(Y_c)</td>
<td>17.5</td>
<td>9.9752E-3</td>
<td>1.0110E-2</td>
<td>3.9854E-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>38.9</td>
<td>1.9744E-3</td>
<td>2.028</td>
<td>2.2811E-3</td>
<td>1.864</td>
<td>4.8755E-3</td>
<td>2.630</td>
</tr>
<tr>
<td>72.1</td>
<td>3.9801E-4</td>
<td>2.559</td>
<td>5.2915E-4</td>
<td>2.368</td>
<td>8.9808E-4</td>
<td>2.471</td>
</tr>
<tr>
<td>143.4</td>
<td>7.0714E-5</td>
<td>2.512</td>
<td>8.7084E-5</td>
<td>2.624</td>
<td>1.7741E-4</td>
<td>2.359</td>
</tr>
</tbody>
</table>

Table III. Example 1: Convergence results for the circularity \(\xi\) and the center of mass \(Y_c\). The results are obtained with adapted meshes.

This article is protected by copyright. All rights reserved.
In the following, we proceed with a comparative study between Strategy I and Strategy II. The first test concerns the convergence properties of both strategies, and we particularly highlight the convergence rates of each Newton variant. Let us consider an initial mesh having $25^3$ 600 elements and we perform simulations using fixed time steps. The residuals $R(\Xi_k^n)$ at iteration $k$ are computed in the $L^\infty$ norm. Given the same initial conditions, we perform computations for various time step sizes $\Delta t$ and we report the residuals convergence curves in Fig. 6. The corresponding convergence rates are expressed as:

$$\frac{\ln \left( \frac{|R(\Xi_k^n)|}{|R(\Xi_{k-1}^n)|} \right)}{\ln \left( \frac{|R(\Xi_{k-1}^n)|}{|R(\Xi_{k-2}^n)|} \right)}$$

for $k \geq 2$. (4.1)

Observe that overall the convergence is very fast for both strategies. The convergence rates in Table IV show that the quadratic convergence is obtained with Strategy I, while the third-order convergence rate is obtained with Strategy II. These are the expected behavior of the two strategies and confirm that the tangent problem is correctly derived and implemented. Notice that any small error would deteriorate the quadratic or cubic behavior, and it rather leads to a linear convergence slope. Gradually increasing the time step impacts on the performance of the Newton method. Indeed, for large values of $\Delta t$, the starting value for the Newton loop becomes far from the expected solution, yielding a plateau where the residual first decreases slowly; it converges subsequently with the expected quadratic or cubic behavior. Above a threshold value of $\Delta t$, a complete breakdown of the algorithm holds. Remark that it exists numerical strategies that improve the Newton convergence when the initial guess is far from the solution such as the damped Newton algorithm, but that is definitely beyond the goal of this work.

<table>
<thead>
<tr>
<th>$\Delta t$</th>
<th>CPU (Strategy I)</th>
<th>CPU (Strategy II)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.001</td>
<td>8.9481</td>
<td>12.9423</td>
</tr>
<tr>
<td>0.01</td>
<td>15.9305</td>
<td>15.8132</td>
</tr>
<tr>
<td>0.1</td>
<td>27.4434</td>
<td>20.6437</td>
</tr>
<tr>
<td>0.15</td>
<td>33.7927</td>
<td>22.9934</td>
</tr>
<tr>
<td>0.2</td>
<td>42.9935</td>
<td>26.9260</td>
</tr>
<tr>
<td>0.275</td>
<td>59.6415</td>
<td>34.8457</td>
</tr>
</tbody>
</table>

Table V. Example 1: Comparison between the computing times when using Strategy I and Strategy II for several values of the time step size.

We now focus on the computational cost for both strategies. To have a quantitative comparison, we evaluate the wall clock time for serial simulations that indicates, together with the errors evaluation, how much effort is needed to establish a certain accuracy. Further insights into the parallel performances and scalability will be only provided subsequently in the three-dimensional
We perform computations with the same structured mesh having 25,600 triangles. In Table V, we plot the timings of the computations for both strategies with respect to the time step size. We can observe that Strategy I is slightly cheaper than Strategy II when the time step is very small (almost for $\Delta t \leq 0.01$). In such a situation, both algorithms converge in less than two iterations. Strategy II is then more expensive than Strategy I because Strategy II requires one more residual’s assembly and resolution of the linear system in each Newton subiteration. However, Strategy II becomes more beneficial for larger time steps thanks to the third-order convergence of this method.

<table>
<thead>
<tr>
<th>mesh size</th>
<th>$\Delta t_{\text{max}}$(Strategy I)</th>
<th>$\Delta t_{\text{max}}$(Strategy II)</th>
<th>$\Delta t_{\text{CFL}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/15</td>
<td>0.89</td>
<td>0.51</td>
<td>0.032</td>
</tr>
<tr>
<td>1/30</td>
<td>0.91</td>
<td>0.50</td>
<td>0.011</td>
</tr>
<tr>
<td>1/60</td>
<td>0.86</td>
<td>0.44</td>
<td>0.0041</td>
</tr>
<tr>
<td>1/120</td>
<td>0.435</td>
<td>0.325</td>
<td>0.0014</td>
</tr>
<tr>
<td>1/240</td>
<td>0.172</td>
<td>0.114</td>
<td>0.0005</td>
</tr>
</tbody>
</table>

Table VI, Example 1: Comparison between the maximal time step $\Delta t_{\text{max}}$ for Strategy I and Strategy II and the time step limit from the stability criterion for several values of the mesh size $h$.

Thereafter, we compare the time step size used in the present method and the time step limit from the stability criterion. Only structured meshes are used and the time step adaptation is disabled. A set of simulations for successively refined meshes are performed, and we display the maximum time steps enabled for both strategies. Using an explicit decoupling scheme where the capillary term is considered as a right-hand side in the momentum equation leads to a stability condition given by the convective time step limit. It imposes restrictions on the time step size: $\Delta t_{\text{CFL}} < \sqrt{\frac{\langle \rho \rangle h^3}{2\pi\gamma}}$, where $\langle \rho \rangle$ is the average fluid density at the interface [29, 31]. We perform numerical simulations using both strategies and we report in Table VI the maximum time step sizes. Results illustrate the gain we have by using the fully implicit schemes, for which we can increase $\Delta t$ up to almost 300 times $\Delta t_{\text{CFL}}$. Moreover, we observe that, although Strategy II converges faster than Strategy I, Strategy I allows to use larger values of $\Delta t$ than Strategy II.

Let us focus now on the adaptive time scheme and we perform simulations using the same structured mesh. In Fig. 7, we compare the adapted values of $\Delta t$ for both strategies against the capillary time step restriction for explicit schemes and we additionally report the corresponding numbers of Newton subiterations. We mainly observe that $\Delta t$ have almost similar values when using both strategies. However, while Strategy II usually converges after two or three iterations, Strategy I usually converges after five iterations.

Consequently, although the quadratically convergent Strategy I allows higher time steps, Strategy II with the third-order convergence features globally better. Indeed, Strategy II is computationally cheaper thanks to its cubic convergence, and we don’t usually use too large time steps as those allowed by Strategy I. In what follows, Strategy II will be used in our computations.

In this last paragraph, we consider the second set-up of the skirted bubble described in Hysing et al. [31]. We only present a sample result obtained using the adaptive time steps scheme and the mesh adaptation procedure, without making any grid convergence study. We consider the
same computational domain and initial conditions adopted in the previous test case. The physical parameters are $\rho_o = 10^3$, $\rho_i = 1$, $\mu_o = 10$, $\mu_i = 0.1$ and $\gamma = 1.96$, while the dimensionless parameters are $\rho^* = 10^{-3}$, $\mu^* = 10^{-2}$, $Re = 35$, $Eo = 125$ and $Ca \approx 3.571$.

Fig. 9 provides successive snapshots showing the adapted meshes and the corresponding interfaces’

This article is protected by copyright. All rights reserved.
shapes, see also Movie 6 in the supplementary material. The adapted mesh provides high mesh density and then more accurate computations in the zones of small filaments. In Fig. 8 the bubble’s shape at \( t = 3 \) is compared to the shapes obtained by the different groups using the finest meshes in [31]. Moreover, Fig. 8 plots the temporal development of the bubble’s area \( |\Omega(t)| \), the circularity \( \delta(t) \), the center of mass \( Y_c(t) \) and the rise velocity \( V_c \). An acceptable agreement is generally observed.

4.2. Example 2: Oscillating bubble dynamics

The second example concerns the relaxing viscous bubble subject to surface tension [60, 61, 7, 62, 63]. We consider the set-up described in [64, 65] of the ellipsoidal interface having initially
the semi-major axis \( a = 0.75 \) and the semi-minor axis \( b = 0.5 \), respectively in the \( x \)-direction and \( y \)-direction, and immersed in a fluid having the density \( \rho = 1 \) and the viscosity \( \mu = 1 \). We set the tension coefficient to \( \gamma = 10 \). The interface is initially centered in a square computational domain \( \Lambda = [-1.5, 1.5] \times [-1.5, 1.5] \). Homogeneous Dirichlet boundary conditions are considered for the velocity. Under the effect of the surface tension, the bubble start oscillating, and the capillary force shall bring the ellipse back to an equilibrium circular steady state having the same area and the radius \( R_\infty = \sqrt{ab} \approx 0.61237 \), in which the capillary force vanishes. We observe the relaxation of the bubble in the time interval \( (0, 6.5) \).

The purpose of this test case is to perform a comparison of the stability properties between the present fully implicit and a fully explicit scheme. For such an explicit scheme, the time step is first bounded by the convective time step limitation \( \Delta t_{\text{CFL}} \), whereas the most limiting time step at the end is the viscous time step limit corresponding to small velocities. We consider the explicit scheme detailed in the appendix A, in which the surface tension appears as a right hand side in the momentum equation. For all simulations in both explicit and implicit cases, we consider the same time step size \( \Delta t = 0.012 \) without applying the adaptive time scheme. The computational domain is discretized on a family of regular meshes without adapting the mesh. The successively refined meshes have the mesh size \( h < 1/20 \). We notice that the stability criterion is violated for all meshes and the simulations will obviously fail depending on the mesh size.

Fig. 10 shows the relaxation of the bubble and the temporal evolution of the bubble’s axes (ie. the vertical and horizontal radii) for several mesh sizes. The numerical results clearly depict that the explicit treatment of the capillary force leads to a severe time step restrictions. However, the stability is significantly improved with the fully implicit scheme when using large time steps and relatively fine meshes.

In Fig. 11 and Fig. 12, we provide some snapshots showing the shapes of the bubble during the relaxation. To better infer the limit of stability, we also plot the pressure profile and some isocontour lines, and we compute some errors when the equilibrium state is reached. Let \( \Delta p|_\Gamma \) represent the pressure jump across the interface and \( H_\infty = 1/R_\infty \) be the exact mean curvature on \( \Gamma \) in the steady state. Let \( \Delta \bar{p}|_\Gamma \) design the jump of pressure across \( \Gamma \) obtained at the equilibrium state with \( h = 1/350 \).
and $\Delta t = 2 \times 10^{-3}$. We introduce the following errors:

$$e_H = \sqrt{\frac{\int_{\Lambda} |\text{div} \mathbf{n} - H_\infty|^2 \delta_\varepsilon(\varphi) |\nabla \varphi|}{2\pi R_\infty}}$$

and

$$e_p(t) = \frac{\left| \Delta p_\Gamma(t) - \Delta \hat{p}_\Gamma \right|}{|\Delta \hat{p}_\Gamma|} \quad \text{with} \quad \Delta \hat{p}_\Gamma = \left| \frac{\int_\Lambda \mathcal{H}_\varepsilon(\varphi) p - \int_\Lambda (1 - \mathcal{H}_\varepsilon(\varphi)) p}{\int_\Lambda \mathcal{H}_\varepsilon(\varphi) - \int_\Lambda (1 - \mathcal{H}_\varepsilon(\varphi))} \right|$$

When using the fully explicit scheme, the solution remains stable when using a mesh size $h = 1/20$. Indeed, that corresponds to a time step size which is almost eight times larger than $\Delta t_{CFL}$. Further mesh refinement results in the non-physical numerical oscillations that pollute the solution and lead finally to a complete breakdown of the algorithm.

However, Fig. 12 shows the stability of the solution is maintained for significantly finer meshes, and we observe that it is still possible to obtain a stable solution when using time step sizes that exceed the capillary time step size by almost a factor of 160. For a mesh resolution $h = 1/320$, we observe that oscillations start to appear for $t \geq 0.3$. The pressure isocontour lines and the error $e_p$ clearly

This article is protected by copyright. All rights reserved.
show that the solution is no longer stable. Notice that although the implicit method allows us to maintain stability for significantly larger time steps, it is not always useful to use too large values in practice. In fact, using too large time steps does not help to capture the physics of the problem.

Thereafter, we verify the temporal convergence of the proposed numerical strategy. We consider the time horizon $t \in (0, T = 1)$. Let $q_h^N$ represent the discrete approximation of a generic field $q$ evaluated at the final time $t^N = T$. We compute the errors corresponding to the temporal discretization of the field $q$ with respect to a reference solution $\hat{q}_h^N$ obtained with the finest time...
step $\Delta t = 3.125 \times 10^{-4}$ as follows:

$$e_{\Delta t,i}(q) = \|q_h^N - \hat{q}_h^N\|_{i,\Omega} \quad \text{and} \quad \text{ROC}_{\Delta t,i}(q) = \frac{\log_{10}(e_{\Delta t,i}(q)/e_{\Delta t,i}(q))}{\log_{10}(\Delta t/\Delta t)},$$

where $i \in \{0,1\}$, and $\Delta t$ and $\Delta t$ represent two consecutive time steps. In Table VII, we provide the errors corresponding to the temporal discretization for successively refined time steps. Accordingly, a second order convergence behavior is observed.

<table>
<thead>
<tr>
<th>$\Delta t$</th>
<th>$e_{\Delta t,0}(u)$</th>
<th>ROC$_{\Delta t,0}(u)$</th>
<th>$e_{\Delta t,0}(p)$</th>
<th>ROC$_{\Delta t,0}(p)$</th>
<th>$e_{\Delta t,0}(\mathcal{H}_s(\varphi))$</th>
<th>ROC$_{\Delta t,0}(\mathcal{H}_s(\varphi))$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$5 \times 10^{-4}$</td>
<td>0.50086</td>
<td>0.20515047</td>
<td>0.014140</td>
<td>1.809</td>
<td>0.000336</td>
<td>1.775</td>
</tr>
<tr>
<td>$2.5 \times 10^{-3}$</td>
<td>0.13235</td>
<td>1.920</td>
<td>0.05470264</td>
<td>1.907</td>
<td>0.004035</td>
<td>1.809</td>
</tr>
<tr>
<td>$1.25 \times 10^{-3}$</td>
<td>0.03509</td>
<td>1.915</td>
<td>0.01418740</td>
<td>1.947</td>
<td>0.001153</td>
<td>1.807</td>
</tr>
<tr>
<td>$6.25 \times 10^{-4}$</td>
<td>0.00959</td>
<td>1.871</td>
<td>0.00374819</td>
<td>1.920</td>
<td>0.000336</td>
<td>1.775</td>
</tr>
</tbody>
</table>

Table VII. Example 2: Temporal error history associated to the time approximation using BDF2. Errors are computed on a fine mesh of size $h = 1/100$.

4.3. Example 3: Three-dimensional test case

In this example, we present numerical simulations showing the applicability of the present methodology in the three-dimensional case ($d = 3$). We emphasize that the aim of this test case is not to perform grid-convergence studies. To that end, we consider a set-up of the rising bubble analogous to that described in Example 4.1 with larger initial bubble. Moreover, we will present a study of the parallel performances.

The computational domain is $\Lambda = (0,1) \times (0,1) \times (0,2)$, and the time interval for the computations is $(0,1.2)$. The bubble is initially circular, centered in $(0.5,0.5,0.5)$ and has a radius equal to 0.3. The fluid is assumed to be initially at rest. The dimensionless parameters are given by $\rho^* = 0.1$, $\mu^* = 0.1$, $Re = 35$, $Eo = 52.5$ and $Ca = 1.5$. In Fig. 13 we provide snapshots of the numerical results, showing the interface $\Gamma$ (colored in red) and the velocity field. The bubble at final time $T = 1.2$ has a nonconvex shape.

We thereafter study the scalability and parallel performance of the numerical solver. However, we do not aim to improve the parallel properties of the solver with respect to the state of the art solvers. The scalability represents an important concept, especially in the three-dimensional case, and measures the ability of the parallel implementation to maintain a constant efficiency. We first introduce the speed up, which compares the computational time required on a parallel architecture to the time required by a sequential implementation. We investigate the parallel performances through the strong scalability and the weak scalability.

At each Newton iteration, the discretized problem is solved in three steps: we first assemble the linear system, then the Jacobian matrix is factorized, and we finally solve the linear system. Thereafter, we consider the test case aforementioned and we study the scaling efficiency for each step. We generate several unstructured meshes using the software GMSH. The nomenclature, number of tetrahedra and the corresponding degrees of freedom are provided in Table VIII. Fig. 14 plots in the logarithmic scale the computational timings, i.e. CPU, with respect to the degrees of
freedom per processor for each step and for the different meshes.

Regarding the strong scalability, it measures, for a fixed problem size, how much the computing time changes if the number of processors increases. A perfect strong scalability corresponds to a slope equal to one in the logarithmic scale. Indeed, a two-time faster execution holds if the number of processors is doubled. Fig. 14 shows that the assembly and the factorization steps are strongly scalable, whereas the strong scalability is deteriorated when solving the linear system. That does not affect significantly the strong scalability of the entire computation, as the time required to solve the linear system is always very small compared to the previous steps.

Regarding the weak scalability, it studies the difference in the computational time with respect to the number of processors for a fixed problem size per CPU. A perfect weak scalability corresponds to an exact overlap of the curves corresponding to the use of different meshes. Fig. 14 shows very close curves during the assembly step when using the different meshes. That results in almost a perfect weak scalability. However, the weak scalability of the Jacobian factorization and resolution of the linear system is not as good as the assembly step because the different timing curves are not

<table>
<thead>
<tr>
<th>Mesh code</th>
<th>Number of tetrahedra</th>
<th>#DOFs</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>20'893</td>
<td>171'729</td>
</tr>
<tr>
<td>M</td>
<td>49'524</td>
<td>389'278</td>
</tr>
<tr>
<td>F</td>
<td>160'894</td>
<td>4'218'199</td>
</tr>
</tbody>
</table>

Table VIII. Example 3: Nomenclature and characteristics of the different meshes used for the scalability study.

Figure 13. Example 3: Snapshots showing the shapes $\Gamma$ of the rising bubble and the velocity field in the vertical mid-planes of the bubble at times $t \in \{0, 0.3, 0.6, 0.85, 1.05, 1.2\}$, respectively.

This article is protected by copyright. All rights reserved.
superimposed.

Without claiming to be exhaustive, we provide qualitative comparison of the parallel performances of the present solver with respect to other similar solvers using the level set method. Compared to the level set solver implemented within the C++ library LifeV in a finite element framework and tested on similar mesh resolutions [66], we observe similar good scaling properties for the assembly of the linear system. Both the strong and weak scalabilities are similarly deteriorated when solving the linear system. In [67], a different method is presented and the fluid interface solver uses a hybrid front tracking/level set method: the Level Contour Reconstruction Method (LCRM), while the code is written in Fortran 2003. The parallel performances are tested on highly refined grids, showing better performances in weak scalability.

5. CONCLUSION

In this paper, we have presented new fully implicit and monolithic approaches to solve free surface problems with surface tension. For the first time, two numerical strategies based on the use of two variants of the Newton method in a finite element framework have been introduced, and we have reported computational results in both two-dimensional and three-dimensional cases. The two strategies require one assembly of the Jacobian system, while Strategy II requires one more evaluation of the global residual. Efficient derivation and implementation of the tangent problem in both cases has been proved through the study of the convergence properties. Numerical investigations have been performed, showing second-order and third-order convergences,
respectively for both strategies, as expected from a theoretical viewpoint. Fully implicit methods feature to be unconditionally stable, and we have shown numerically that stability is ensured for considerably larger time steps with respect to the explicit decoupling strategy most commonly used in the published literature [24, 29, 31]. In addition, we have observed that, although Strategy I allows the use of higher time steps, Strategy II is in general computationally cheaper and represents our preferable approach.

The improvements and extensions of the present method should be further explored. In particular, we are focusing on the development of some inexact variants of the Newton method that can be efficient and computationally cheaper compared to the present strategies. Since the Newton method presents only local convergence properties, i.e. the starting values must be close enough to the solution, we are currently investigating some globalized Newton variants based on some damping techniques to allow the use of larger time steps. Moreover, future work should investigate the robustness of the present method in the case of high Reynolds numbers. Last but not least, we also foresee the applicability of the proposed framework to the simulation of interfacial flows with topology changes.

The present approach is general and can be useful for broad set of engineering problems with multiphase flows. In particular, the current developments are part of an ongoing work to study the hydrodynamics of lipid bilayer vesicles mimicking red blood cells under bending forces in small capillaries where the Reynolds number is small enough for the Stokes limit to be valid [68, 69, 70]. The vesicle problem is highly nonlinear and the bending force includes fourth order derivatives with respect to the cell shape [71]. Fully explicit decoupling strategies are always used in the published literature, yielding a more severe stability condition than the one characterizing the capillary problem. The derivation and study of fully implicit strategies remain a challenging topic, and the present strategies can provide an efficient way to implicitly solve the vesicle problem.

ACKNOWLEDGEMENT

The authors gratefully acknowledge the support by the Swiss National Science Foundation through the grant 320030-149567. The authors would like to thank the three anonymous reviewers for their insightful comments that help improve the quality of the paper.

A. EXPLICIT SCHEME FOR A COMPARATIVE STUDY

To compare the present fully implicit method with respect to the explicit methods, we introduce the following scheme in which an explicit treatment of surface tension holds, while the backward differentiation scheme of second order approximates the time derivative terms and an explicit time discretization of the inertia term is considered.

The explicit algorithm consists in decoupling and solving the fluid problem and the level equation in a segregated fashion. The velocity and pressure are first computed using a monolithic approach for the Navier–Stokes system, in which the surface tension force is added as a source term in the right hand side of the momentum equation. The level set function is subsequently advected using the computed velocity. Finally, the mean curvature is computed and projected in the appropriate finite element space. The semi-discrete problem reads:

This article is protected by copyright. All rights reserved.
1. Given \( \varphi_{n-1} \), find the velocity \( \mathbf{u}_n \in \mathbb{V}(\mathbf{u}_b) \) and pressure \( p_n \in \mathbb{Q} \) such that:

\[
\begin{align*}
\text{Re}_c(\varphi_{n-1}) \left( \frac{3\mathbf{u}_n - 4\mathbf{u}_{n-1} + \mathbf{u}_{n-2}}{2\Delta t} + (\mathbf{u}_{n-1} \cdot \nabla) \mathbf{u}_n \right) - \text{div}(2\mu_c(\varphi_{n-1})\mathbf{D}(\mathbf{u}_n)) \\
+ \nabla p_n = \frac{1}{\text{Ca}} \mathbf{H}_{n-1} \frac{\nabla \varphi_{n-1}}{\abs{\nabla \varphi_{n-1}}} \varepsilon(\varphi_{n-1}) + \text{Re}_c(\varphi_{n-1}) \mathbf{g},
\end{align*}
\]

(A.1)

\[
\text{div} \mathbf{u}_n = 0.
\]

(A.2)

2. Given \( \mathbf{u}_n \), find the level set function \( \varphi_n \in \mathbb{V}(\varphi_{n-1}) \) such that:

\[
\frac{3\varphi_n - 4\varphi_{n-1} + \varphi_{n-2}}{2\Delta t} + \mathbf{u}_n \cdot \nabla \varphi_n = 0.
\]

(A.3)

REFERENCES


This article is protected by copyright. All rights reserved.


This article is protected by copyright. All rights reserved.


