Journal of Computational Physics 353 (2018) 435–459

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

An efficient mass-preserving interface-correction level set/ghost fluid method for droplet suspensions under depletion forces

Zhouyang Ge *, Jean-Christophe Loiseau [1], Outi Tammisola, Luca Brandt

Linné Flow Centre and SeRC (Swedish e-Science Research Centre), KTH Mechanics, S-100 44 Stockholm, Sweden

ARTICLE INFO

Article history:
Received 22 January 2017
Received in revised form 23 October 2017
Accepted 24 October 2017
Available online 12 November 2017

Keywords:
Multiphase flow
Level set method
Ghost fluid method
Colloidal droplet
Depletion force

ABSTRACT

Aiming for the simulation of colloidal droplets in microfluidic devices, we present here a numerical method for two-fluid systems subject to surface tension and depletion forces among the suspended droplets. The algorithm is based on an efficient solver for the incompressible two-phase Navier–Stokes equations, and uses a mass-conserving level set method to capture the fluid interface. The four novel ingredients proposed here are, firstly, an interface-correction level set (ICLS) method; global mass conservation is achieved by performing an additional advection near the interface, with a correction velocity obtained by locally solving an algebraic equation, which is easy to implement in both 2D and 3D. Secondly, we report a second-order accurate geometric estimation of the curvature at the interface and, thirdly, the combination of the ghost fluid method with the fast pressure-correction approach enabling an accurate and fast computation even for large density contrasts. Finally, we derive a hydrodynamic model for the interaction forces induced by depletion of surfactant micelles and combine it with a multiple level set approach to study short-range interactions among droplets in the presence of attracting forces.

1. Introduction

In the field of colloidal science, much progress has been made on the synthesis of elementary building blocks (Fig. 1) mimicking molecular structures to elaborate innovative materials, *e.g.* materials with complete three dimensional band gaps [1–4]. The basic elements of such colloidal molecules are particles or droplets less than one millimeter in size, and their self-assembly relies on either lengthy brownian motion or careful microfludic designs, on top of typical colloidal interactions, *e.g.* depletion attraction and electrostatic repulsion [5–7]. Regardless of the approach, however, questions remain why the colloidal particles/droplets undergo certain path to organize themselves and how such process can be controlled and optimized. Since full data are not yet accurately accessible from experiments in such miniature systems, computer simulations will be useful to provide supplemental information.

Scaling down to microscale appears first to be a convenience for the numerical simulations of multicomponent and multiphase systems as the non-linear Navier–Stokes (NS) equations can be reduced to the linear Stokes equations. This allows

* Corresponding author.
E-mail addresses: zhoge@mech.kth.se (Z. Ge), jean-christophe.loiseau@ensam.eu (J.-C. Loiseau), outi@mech.kth.se (O. Tammisola), luca@mech.kth.se (L. Brandt).
[1] Present address: Laboratoire DynFluid, Arts et Métiers ParisTech, 151 boulevard de l'hôpital, 75013 Paris, France.

https://doi.org/10.1016/j.jcp.2017.10.046
0021-9991/

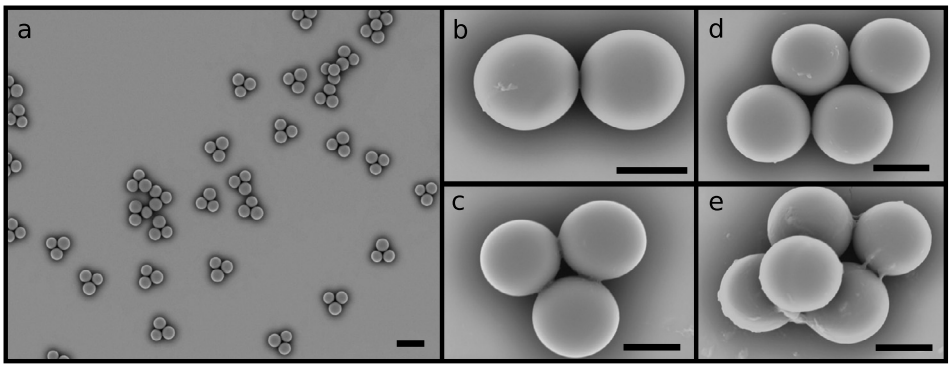


Fig. 1. Self-assembled colloidal clusters. a) Electron micrograph of a suspension of triplet clusters. Scale bar, 30 μm. b–e) Close up of doublet, triplet, quadruplet, and quintuplet clusters. Scale bars, 10 μm. Further details are available in [7], photograph courtesy of Dr. Joshua Ricouvier.

the use of boundary integral methods (BIM) [8], *e.g.* most recently the GGEM-based BIM [9,10] solving the Stokes equations in general geometries. However, it is also possible to use the conventional unsteady, fractional-step/projection-method NS solver at low Reynolds number, combined with an interface description method [11,12]. The latter approach is more versatile, probably less difficult to implement, and enjoys a rich literature of standard numerical techniques. Here, in view of a rich range of possible applications and considering also the rapid development of inertial microfluidics (where inertial effects are used to better control the flow behavior) we take the approach of simulating the incompressible, two-fluid NS as outlined in [13]. The splitting procedure proposed in [13] enables the use of fast solvers for the pressure Poisson equation also for large density and viscosity contrasts. The remaining choice then is to be made among the available interface-description methods.

Generally, there are two categories of methods to resolve an interface in a NS solver, *i.e.* front-tracking methods and front-capturing methods. An example of the front-tracking method is the immersed boundary method (IBM) [14,15]. Using Lagrangian points in a moving frame, IBM can offer a high interface resolution without the need to deform the underlying mesh in the fixed frame. However, the coupling of the two meshes relies on a regularized delta function, which introduces certain degrees of smearing. Moreover, large interface deformation requires frequent mesh rearrangement; and topology changes may have to be handled manually. These constraints make IBM typically more expensive and less appealing for droplet simulations.

Front-capturing methods, on the other hand, are Eulerian and handle topology changes automatically; they are therefore easier to parallelize to achieve higher efficiency. One of such methods is the volume-of-fluid (VOF) method [16], which defines different fluids with a discontinuous color function. The main advantage of VOF is its intrinsic mass conservation. It suffers however from inaccurate computations of the interface properties, *e.g.* normals and curvatures. This makes it less favorable for simulations of microfluidic systems where surface tension is the dominant effect and requires accurate modelling.

Another popular front-capturing method is the level set (LS) method [17,18]. Contrary to VOF, LS prescribes the interface through a (Lipschitz-)continuous function which usually takes the form of the signed distance to the interface. Under this definition, normals and curvatures of the interface can be readily and accurately computed. However, the problem when simulating incompressible flows is that mass loss/gain may occur and accumulate because the LS function embeds no volume information. In addition, errors can also arise from solving the LS advection equation and/or the reinitialization equation, a procedure commonly required to reshape the LS into a distance function. Therefore, additional measures have to be taken to ensure mass conservation.

Many different approaches have been proposed to make LS mass-conserving, which can be classified into the following four methodologies. The first approach is to improve the LS discretization and reinitialization so that numerical errors are reduced. In practice, one can increase the order of LS fluxes [19], minimize the displacement of the zero LS during reinitialization [20,19], or employ local mesh refinement [21–23]. By doing so, mass loss can be greatly reduced, although the LS function is still inherently non-conservative. The second remedy couples the LS with a conservative description (*e.g.* VOF) or Lagrangian particles. For example, the hybrid particle level set method [24], the coupled level set volume-of-fluid (CLSVOF) method [25], the mass-conserving level set (MCLS) method [26], or the recent curvature-based mass-redistribution method [27]. With varying level of coupling, these methods can usually preserve mass really well; the drawback is that the complexity and some inaccuracy (due to interpolation, reconstruction, *etc.*) of the other method will be imported. The third approach improves mass conservation by adding a volume-constraint in the LS or NS formulation. Examples of this kind include the interface-preserving LS redistancing algorithm [28] and the mass-preserving NS projection method [29]. Finally, one can also smartly modify the definition of the LS, such as the hyperbolic-tangent level set [30], to reduce the overall mass loss.

With the physical application of colloidal droplets in mind, and using ideas from some of the above-mentioned methods, we heuristically propose an interface-correction level set (ICLS) method. The essential idea of ICLS is to construct a normal

velocity supported on the droplet interface and use it in an additional LS advection to compensate for mass loss, in a way similar to inflating a balloon. Because no coupling with VOF or Lagrangian particles is required, the simplicity and high accuracy of the original LS method is preserved, yet the extra computational cost of this procedure is negligible.

Provided a mass-preserving level set method, the coupled flow solver must also accurately compute the surface tension, a singular effect of the normal stress on the interface. This is particularly important for microfluidic systems; as surface tension scales linearly with the dimension, it decays slower than volumetric forces (*e.g.* gravity) when the size of the system reduces. To handle such discontinuities, one approach is the continuum surface force (CSF) [31], originally developed for the VOF method, later extended to the LS [18]. Although easy to implement, CSF effectively introduces an artificial spreading of the interface by regularizing the pressure difference, and it can become erroneous when two interfaces are within its smoothing width. A second, non-smearing approach is the ghost fluid method (GFM). Proposed initially for solving compressible Euler equations [32], GFM provides a finite-difference discretization of the gradient operator even if the stencil includes shocks. It has been proven to converge [33] and was soon applied for treating the pressure jump in multiphase flows [34]. We note that although the GFM can be reformulated in a similar way to the CSF [35,36], its treatment for discontinuous quantities is sharp in the finite difference limit.

Several implementation options of the GFM were suggested in [34,35,37]. Here, we follow the methodology of [37], *i.e.* using the GFM for the pressure jump due to surface tension while neglecting the viscous contribution. As will be discussed later, this choice is especially suitable for microfluidic applications where the capillary effect is strong. To efficiently solve for the pressure, we further combine the GFM with a fast pressure-correction method (FastP*) [13]. Such a combination enables a direct solve of the pressure Poisson equation using the Gauss elimination in the Fourier space; it is the most efficient when the computational domain is periodic, but it also applies to a range of homogeneous Dirichlet/Neumann boundary conditions via fast sine/cosine transforms [38], see *e.g.* a recent open-source distribution [39]. Using a second-order accurate, grid-converging interface curvature estimation, we will show that the coupled ICLS/NS solver can handle large density/viscosity contrasts and converges between first and second order in both space and time.

Finally, a unique challenge to the simulation of colloidal droplets is the modeling of near-field interactions. It is known that two or more colloids can interact via dispersion, surface, depletion, and hydrodynamic forces [5]. Apart from the hydrodynamic forces which is determined directly from the NS, and the dispersion forces which arise from quantum mechanical effects, the depletion and surface forces must be modelled. These forces can be either attraction or repulsion and are typically calculated from the gradient of a potential. Based on colloidal theory, we propose a novel hydrodynamic model for the depletion force in the framework of the ICLS/NS solver. Our method relies on two extensions: *i)* extending the single level set (SLS) function to multiple level set (MLS) functions; and *ii)* extending the GFM for computation of the gradient of depletion potential. MLS has the benefits that each droplet within a colloidal cluster can be treated individually, is allowed to interact with the other droplets, and is guarded from its own mass loss. MLS also prevents numerical coalescence of droplets when they get too close. The computational complexity, proportional to the number of MLS functions (l) and the number of cells in each dimension (N), is higher than SLS. However, we note that many techniques exist to reduce the CPU cost and/or memory consumption if lN^d ($d = 2$ or 3) is large. For detailed implementations of such optimized algorithms we refer to [40–42]. In the present paper, we will demonstrate the self-assembly of colloidal droplets using one droplet per MLS function.

The paper is organized as follows. In Sec. 2, the governing equations for the incompressible, two-phase flow are briefly presented. In Sec. 3, the classical signed-distance LS methodology together with some commonly used numerical schemes is discussed. We then introduce the ICLS method in Sec. 4, starting from the derivation ending with a demonstration. We further provide a geometric estimation of the interface curvature tailored to the GFM in Sec. 5. The complete ICLS/NS solver is outlined in Sec. 6, including a detailed description of the implementation and three examples of validation. In Sec. 7, we propose a MLS/GFM-based method for the modeling of near-field depletion potential. Finally, we summarize the overall methodology in Sec. 8.

2. Governing equations for interfacial two-phase flow

The dynamics of the incompressible flow of two immiscible fluids is governed by the Navier–Stokes equations, written in the non-dimensional form

$$\nabla \cdot \boldsymbol{u} = 0, \tag{1a}$$

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} = \frac{1}{\rho_i}\left(-\nabla p + \frac{1}{Re}\nabla \cdot \left[\mu_i(\nabla \boldsymbol{u} + \nabla \boldsymbol{u}^T)\right]\right) + \frac{1}{Fr}\boldsymbol{g}, \tag{1b}$$

where $\boldsymbol{u} = \boldsymbol{u}(\boldsymbol{x}, t)$ is the velocity field, $p = p(\boldsymbol{x}, t)$ is the pressure field, and $\boldsymbol{g}$ is a unit vector aligned with gravity or buoyancy. ρ_i and μ_i are the density and dynamic viscosity ratios of fluid i ($i = 1$ or 2) and the reference fluid. These properties are constant in each phase and subject to a jump across the interface, which we denote as $[\rho]_\Gamma = \rho_2 - \rho_1$ for density and $[\mu]_\Gamma = \mu_2 - \mu_1$ for viscosity. For viscous flows, the velocity and its tangential derivatives are continuous on the interface [43]. However, the pressure is discontinuous due to the surface tension and the viscosity jump, *i.e.*

$$[p]_\Gamma = \frac{1}{We}\kappa + \frac{2}{Re}[\mu]_\Gamma \boldsymbol{n}^T \cdot \nabla \boldsymbol{u} \cdot \boldsymbol{n}, \tag{2}$$

where κ is the interface curvature, and $\boldsymbol{n}$ is the normal to the interface. If the surface tension coefficient, $\tilde{\sigma}$, varies on the interface the tangential stress is also discontinuous. In this paper, we assume constant and uniform $\tilde{\sigma}$. In Eqs. (1b) and (2), Re, We, and Fr are, respectively, the Reynolds, Weber, and Froude numbers, defined as

$$Re = \frac{\tilde{\rho_1}\tilde{U}\tilde{L}}{\tilde{\mu_1}}, \qquad We = \frac{\tilde{\rho_1}\tilde{U}^2\tilde{L}}{\tilde{\sigma}}, \qquad Fr = \frac{\tilde{U}^2}{\tilde{g}\tilde{L}}, \tag{3}$$

where $\tilde{U}$, $\tilde{L}$, $\tilde{\rho_1}$, $\tilde{\mu_1}$, and $\tilde{g}$ denote the reference dimensional velocity, length, density, dynamic viscosity, and gravitational acceleration. Note that $\rho_1 = 1$ and $\mu_1 = 1$ (i.e. we define fluid 1 as the reference fluid).

3. Classical level set methodology

In the level set framework, the interface Γ is defined implicitly as the zero value of a scalar function $\phi(\boldsymbol{x}, t)$, *i.e.* $\Gamma = \{\boldsymbol{x} \mid \phi(\boldsymbol{x}, t) = 0\}$. Mathematically, $\phi(\boldsymbol{x}, t)$ can be any smooth or non-smooth function; but it is classically shaped as the signed Euclidean distance to the interface [44,18], *viz.*

$$\phi(\boldsymbol{x}, t) = sgn(\boldsymbol{x})|\boldsymbol{x} - \boldsymbol{x}_{\boldsymbol{\Gamma}}|, \tag{4}$$

where $\boldsymbol{x}_\Gamma$ denotes the closest point on the interface from nodal point $\boldsymbol{x}$, and $sgn(\boldsymbol{x})$ is a sign function equal to 1 or -1 depending on which side of the interface it lies. For two-phase problems with single level set, $sgn(\boldsymbol{x})$ provides a natural "color function" for phase indication. Furthermore, with this definition, geometric properties such as the unit normal vector, $\boldsymbol{n}$, and the local mean curvature, κ, can be conveniently computed as

$$\boldsymbol{n} = \frac{\nabla\phi}{|\nabla\phi|}, \tag{5}$$

$$\kappa = -\nabla \cdot \boldsymbol{n}. \tag{6}$$

3.1. Advection

The motion of a fluid interface is governed by the following PDE

$$\frac{\partial\phi}{\partial t} + \boldsymbol{u} \cdot \nabla\phi = 0, \tag{7}$$

where $\boldsymbol{u}$ is the flow velocity field. Despite of its simple form, obtaining an accurate and robust solution to Eq. (7) is challenging. For two-fluid problems, state-of-the-art level set transport schemes include the high-order upstream-central (HOUC) scheme [19], the weighted essentially non-oscillatory (WENO) scheme [43], the semi-Lagrangian scheme [45], or the semi-jet scheme [46]. Quantitative comparisons of these schemes in various test cases can be found in [19,46]. We note that the choice of the scheme is case-dependent, *i.e.* depending on the smoothness of the overall level set field or the stiffness of Eq. (7). For flows involving moderate deformations, HOUC is usually sufficient and most efficient. For more complex flows, WENO or semi-Lagragian/jet schemes combined with grid refinement might be pursed. In the present study, we use either HOUC5 or WENO5 (5 denotes fifth-order accuracy) to evaluate $\nabla\phi$.

For the temporal discretization of Eq. (7), we use a three-stage total-variation-diminishing (TVD) third-order Runge–Kutta scheme [47]. Denoting $f(\phi) = -\boldsymbol{u} \cdot \nabla\phi$, it updates ϕ from time level n to $n+1$ in three sub-steps

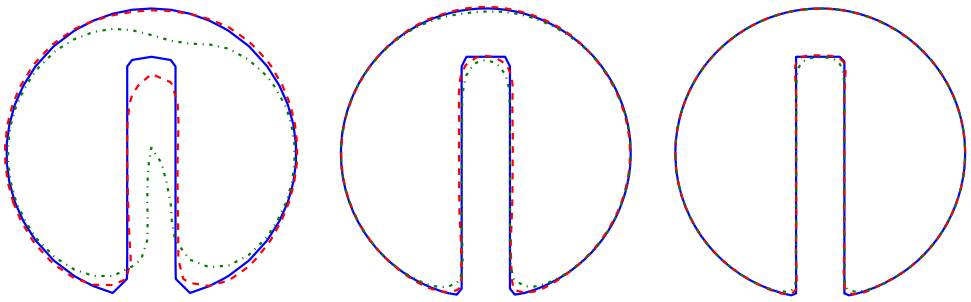
$$\begin{cases} \phi^1 = \phi^n + \Delta t \cdot f(\phi^n) \\ \phi^2 = \frac{3}{4}\phi^n + \frac{1}{4}\phi^1 + \frac{1}{4}\Delta t \cdot f(\phi^1) \\ \phi^{n+1} = \frac{1}{3}\phi^n + \frac{2}{3}\phi^2 + \frac{2}{3}\Delta t \cdot f(\phi^2). \end{cases} \tag{8}$$

Finally, we note that Eq. (7) does not need to be solved in the entire computational domain, as only the near-zero values are used to identify the interface and compute its curvature. This motivated the so-called narrow band approach [48,40], which localizes the level set to the interface using index arrays. Combined with optimal data structures [41,42], fast computation and low memory footprint may be achieved at the same time. In our implementation, we store all the level set values while only update those in a narrow band, *i.e.* solving $\phi_t + c(\phi)\boldsymbol{u} \cdot \nabla\phi = 0$ with the cut-off function given as

$$c(\phi) = \begin{cases} 1 & \text{if} \quad |\phi| < \gamma \\ 0 & \text{otherwise}, \end{cases} \tag{9}$$

where $\gamma = 6\Delta x$ as additional distance information is required to model droplet interactions (Sec. 7). This is equivalent to [40] with a simplified $c(\phi)$.

Zalesak's disk. The Zalesak's disk [49], *i.e.* a slotted disc undergoing solid body rotation, is a standard benchmark to validate level set solvers. The difficulty of this test lies in the transport of the sharp corners and the thin slot, especially in under-resolved cases. The initial shape should not deform under solid body rotation. Hence, by comparing the initial level set field



(a) 50×50 (b) 100×100 (c) 200×200

Fig. 2. Comparison of the initial interface and its shape after one full rotation for different mesh resolutions. Solid lines depict the initial interface. Two different schemes have been used to evaluate the gradients, namely HOUC5 (dashed lines) and WENO5 (dash-dotted line). (For interpretation of the colors in this figure, the reader is referred to the web version of this article.)

and that after one full rotation one can characterise the degree of accuracy of a numerical solver. Here, the parameters are chosen so that a disk of radius 0.15, slot width of 0.05 is centered at $(x, y) = (0, 0.25)$ of a $[-0.5, 0.5] \times [-0.5, 0.5]$ box. The constant velocity field is given as

$$u = -2\pi y, \quad v = 2\pi x. \tag{10}$$

Three different mesh resolutions have been considered, namely 50×50, 100×100 and 200×200. Fig. 2 depicts the shape of the interface after one full rotation of the disk, solving Eq. (7) only. Along with the results of the HOUC5 scheme (red dashed line), the shape of the interface obtained using the WENO5 scheme (green dash-dotted line) is also reported in this figure. Both schemes yield good results on fine grids, but HOUC5 clearly outperforms WENO5 on the coarsest mesh considered here.

3.2. Reinitialization

Although the level set function is initialized to be a signed-distance, it may lose this property as time evolves, causing numerical issues particularly in the evaluation of the normal and the curvature [18]. In order to circumvent these problems, an additional treatment is required to constantly reshape ϕ into a distance function, *i.e.* $|\nabla\phi| = 1$. This can be done either with a direct, fast marching method (FMM) [17], or by converting it into a time-dependent Hamilton–Jacobi equation [18]

$$\frac{\partial \phi}{\partial \tau} + S(\phi_0)(|\nabla\phi| - 1) = 0, \tag{11}$$

where τ is a pseudo-time, and $S(\phi_0)$ is a mollified sign function of the original level set, usually defined as

$$S(\phi_0) = \begin{cases} -1 & \text{if} \quad \phi_0 < -\Delta x \\ 1 & \text{if} \quad \phi_0 > \Delta x \\ \frac{\phi_0}{\sqrt{\phi_0^2 + \Delta x^2}} & \text{otherwise.} \end{cases} \tag{12}$$

Comparing with FMM, the second approach allows the use of higher order schemes (*e.g.* WENO5) and is easy to parallelize; hence, it has been a much more popular choice. However, as pointed out by Russo and Smereka [20], using regular upwinding schemes for $\nabla\phi$ near the interface does not preserve the original location of the zero level set. This can lead to mass loss, especially if the level set is far from a distance function and Eq. (11) needs to be evolved for long time. A simple solution is to introduce a "subcell fix" [20], which pins the interface in the reinitialization by modifying the stencil. Beautifully as it works in redistancing the level set, this method is however only second order accurate and thus not well-suited for evaluating curvature. Its fourth order extension [50] suffers from stability issues and may require a very small pseudo-time step [22]. Based on these observations, in this paper we solve Eq. (11) using the classical WENO5 [43] and the same SSP-RK3 [47]. The reinitialization is not performed at every physical time step, but depends on the advection velocity. In our applications, it typically requires one to two iterations of Eq. (11) per ten to a hundred time steps.

Distorted elliptic field. In order to illustrate the redistancing procedure, a test case similar to the one in [20] is considered. Define the initial level set as

$$\phi(x, y, 0) = f(x, y)\left(\sqrt{\left(\frac{x^2}{4} + \frac{y^2}{16}\right)} - 1\right),$$

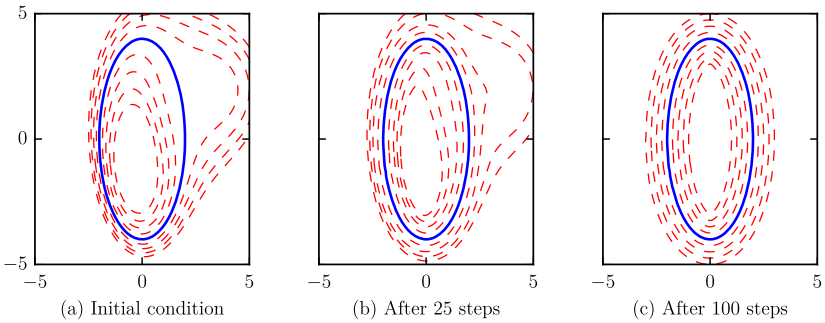


Fig. 3. Illustration of the reinitialization procedure. The shape of the ellipsoid is depicted as the thick solid line. The dashed lines then depict iso-contours of $\phi(x, y)$ ranging from -1 to 1 by increments of 0.25. (For interpretation of the colors in this figure, the reader is referred to the web version of this article.)

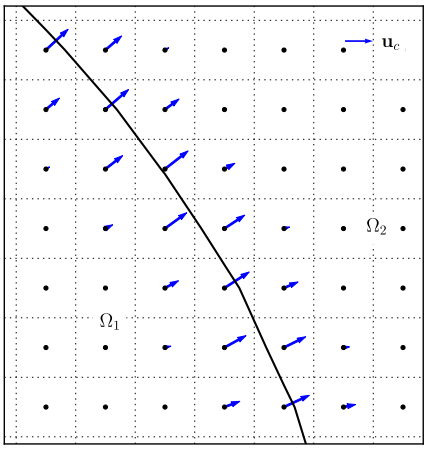


Fig. 4. 2D illustration of the mass correction. The solid line represents the interface. The arrows indicate the normal correction-velocity located at cell centers of the grid.

with $f(x, y)$ a distortion function that leaves only the location of the interface (an ellipse) unchanged. The initial condition is displayed in Fig. 3(a), where the shape of the ellipse is depicted as the thick blue line; the red dashed lines depict iso-contours of ϕ ranging from -1 to 1. Clearly, this initial condition is far from being equidistant. However, as $\phi(x, y, \tau)$ is evolved under Eq. (11), it eventually converges towards a signed-distance function as seen in Fig. 3(b) and (c).

4. Interface-correction level set (ICLS) method

It is known that classical level set methods lead to mass loss when applied to multiphase flows, partially because there is no underlying mass conservation in the level set formalism, partially because of the reinitialization procedure. Such mass loss can sometimes be reduced or even removed by using the various approaches listed in Sec. 1, *e.g.* the CLSVOF method [25] or the hybrid particle level set method [24]. However, doing so often makes the level set schemes complicated to implement and less efficient. To maintain the simplicity of the original level set method, we propose an alternative approach to conserve mass by performing small corrections near the interface. Because such corrections are done by directly solving a PDE (same as Eq. (7)), the proposed method is straightforward to implement in both 2D and 3D. Meanwhile, because the correction does not need to be performed at every time step, the additional cost is also negligible. Below, we first present the derivation of the correction-velocity, then we demonstrate the mass conservation with an example.

Let Γ divide a domain into two disjoint subsets Ω_1 (*e.g.* a droplet) and Ω_2 (*e.g.* the ambient fluid), and V denote the volume of Ω_1 (Fig. 4). Without loss of generality, we let $\phi < 0$ in Ω_1, and $\phi > 0$ in Ω_2. The rate of change of V can be written as the integral of a normal velocity $\boldsymbol{u}_c$ defined on Γ [29], *i.e.*

$$\int_\Gamma \boldsymbol{n} \cdot \boldsymbol{u}_c \, d\Gamma = \frac{\delta V}{\delta t}, \tag{13}$$

where $\boldsymbol{n}$ is the outward-pointing normal from the interface Γ. If $-\delta V/\delta t$ corresponds to the mass loss over an arbitrary period of time (it does not have to be the time step of the level set advection), then $\boldsymbol{u}_c$ can be thought as a surface velocity

that corrects the volume by an amount $\delta V/\delta t$, hence compensating the mass loss. In other words, if $\boldsymbol{u}_c$ is known, then the following PDE can be solved,

$$\frac{\partial \phi}{\partial t} + \boldsymbol{u}_c \cdot \nabla \phi = 0, \tag{14}$$

after which the mass loss accumulated over δt is removed.

To obtain such a surface correction-velocity $\boldsymbol{u}_c$, we introduce a speed function f_s, an auxiliary pressure p_c, and express the rate of change of $\boldsymbol{u}_c$ as

$$\frac{d\boldsymbol{u}_c}{dt} = -f_s \nabla p_c. \tag{15}$$

Here, p_c can be imagined as a non-dimensional correction-pressure in Ω_1. If $f_s = 1$, the physical interpretation of Eq. (15) is analogous to the inflation of a balloon by δV under pressure p_c over time Δt. It is more evident rewriting $\boldsymbol{u}_c$ in the form of the impulse-momentum theorem (per unit "mass" of the interface)

$$\boldsymbol{u}_c = -\int_0^{\Delta t} \nabla p_c \, dt, \tag{16}$$

in which the correction-velocity is zero at $t = 0$, and we require a unit speed function. In general, substituting Eq. (16) into Eq. (13) results in

$$\int_0^{\Delta t} dt \int_\Gamma \boldsymbol{n} \cdot (-f_s \nabla p_c) \, d\Gamma = \frac{\delta V}{\delta t}. \tag{17}$$

In order for ∇p_c to be compatible with $\boldsymbol{u}_c$, p_c has to be differentiated at the interface. Using a 1D regularized Heaviside function of ϕ, such as

$$H_\epsilon(\phi) = \begin{cases} 1 & \text{if} \quad \phi > \epsilon \\ \frac{1}{2}\left[1 + \frac{\phi}{\epsilon} + \frac{1}{\pi}\sin\left(\frac{\pi\phi}{\epsilon}\right)\right] & \text{if} \quad |\phi| \leqslant \epsilon \\ 0 & \text{otherwise,} \end{cases} \tag{18}$$

with $\epsilon = 1.5\Delta x$ the half smoothing width, the correction-pressure and its gradient in Eq. (17) can be conveniently written as

$$p_c = \big(1 - H_\epsilon(\phi)\big) p_0, \tag{19}$$

and

$$\int_\Gamma \nabla p_c = -\int_\Gamma \delta_\epsilon(\phi) \nabla \phi \, p_0, \tag{20}$$

where $\delta_\epsilon(\phi)$ is the derivative of $H_\epsilon(\phi)$, and p_0 is a constant. Note that $\boldsymbol{n} \cdot \nabla \phi = |\nabla \phi|$, we can denote $\int_\Gamma f_s \delta_\epsilon(\phi) |\nabla \phi| \, d\Gamma = A_f$ and express the constant pressure algebraically

$$p_0 = \frac{\delta V}{\delta t} \frac{1}{A_f \Delta t}, \tag{21}$$

by substituting Eq. (20) into (17), and approximating the time integration to first order, *i.e.* $\int_0^{\Delta t} A_f \, dt = A_f \Delta t$. Finally, Eqs. (15) (20) and (21) can be combined to give

$$\boldsymbol{u}_c(\phi) = \frac{\delta V}{\delta t} \frac{f_s \delta_\epsilon(\phi)}{A_f} \nabla \phi, \tag{22}$$

or

$$\boldsymbol{u}_c(\phi) = \frac{\delta V}{\delta t} \frac{f_s}{A_f} \nabla H_\epsilon(\phi). \tag{23}$$

Once $\boldsymbol{u}_c$ is found, Eq. (14) can be solved for one time step to correct the mass loss. Here, we have required a bounded support for $\boldsymbol{u}_c$, *i.e.* $\boldsymbol{u}_c = \boldsymbol{0}$ for $|\phi| \geqslant \epsilon$ (see Fig. 4). There are two benefits of spreading the surface velocity. First, it allows an easy handling of the interface location, as $\boldsymbol{u}_c$ only depends on a 1D Dirac delta function of the level set. The choice of $\delta_\epsilon(\phi)$ can also be different from the trigonometric form implied from Eq. (18); however, we prove in Appendix A that

the discretization error of $\int_\Gamma \boldsymbol{n}\cdot\boldsymbol{u}_c d\Gamma$ is always zero, independent of $\delta_\epsilon(\phi)$. The important point here is we spread the *correction-velocity* rather than the *interface*. The interface remains sharp, as it is implicitly represented by the level set function. The second benefit of spreading $\boldsymbol{u}_c$ is that it greatly reduces the risk of numerical instability. As $\boldsymbol{u}_c$ is supported on a 2ϵ band around the interface, the maximal nodal value of $\boldsymbol{u}_c$ scales with $1/\epsilon$. In our tests, we have never found its non-dimensional value to exceed 1. Therefore, the CFL conditions imposed by Eq. (14) is satisfied as long as we use the same temporal scheme (*e.g.* RK3) for solving Eq. (7) and Eq. (14). Lastly, we remind the reader that our correction-velocity differs conceptually from the extension-velocity proposed for solving Stefan problems [51,52]. The extension-velocity by design will keep the level set a distance function; while the design principle here is to preserve the global mass. This distinction is clear comparing the construction procedures of the two velocities.

A final question is the choice of the speed function f_s, acting as a pre-factor for $\boldsymbol{u}_c$ in Eq. (22) or (23). To the best of the authors knowledge, there is no simple, universally-valid criteria for such corrections. Two possible ways are

$$f_s \equiv \begin{cases} 1 & \text{uniform speed} \\ \kappa(\phi) & \text{curvature-dependent speed.} \end{cases} \tag{24}$$

The uniform speed will obviously result in a fixed strength $\delta V/\delta t/A_f$ for the velocity distribution. In the case of a static spherical droplet, this is the ideal choice for f_s, since the droplet should remain a sphere. In more general cases, when a fluid interface is subject to deformations or topological changes, a curvature-dependent speed may be more appropriate. This is based on the assumption that local structures of higher curvature or regions where the flow characteristics merge tend to be under-resolved [24]; hence, they are more prone to mass losses. Indeed, a linear curvature weight has been adopted by many and demonstrated to produce accurate results in different contexts [27,53]. Furthermore, κ/A_f reduces to $1/A_f$ when the curvature is uniform. Therefore, we can rewrite Eq. (23) using a curvature-dependent speed

$$\boldsymbol{u}_c(\phi) = \frac{\delta V}{\delta t}\frac{\kappa(\phi)}{A_f}\nabla H_\epsilon(\phi). \tag{25}$$

Clearly, this correction-velocity is larger in highly curved parts, and smaller in flatter parts. It thus includes "local" information while maintaining "global" mass conservation. Standard central-difference discretization applies, where the components of $\boldsymbol{u}_c$ can be obtained at either the cell faces or cell centers. The computation of $\kappa(\phi)$ is crucial and will be presented in the next section. We stress that such a curvature-dependence is not unique. In principle, one can choose different weight-functions, and validate the choice based on the specific applications. Practically, the difference is expected to be negligible since the mass loss remains small (typically around 10^{-5}) at each correction step.

After correcting the level set on a 2ϵ band around the interface, a reinitialization step is required to redistance the values within the entire narrow band (2γ). The two procedures can be readily combined, since it is not necessary to perform mass correction at every time step. Also, because the formalism is cast in a level set frame, generalization from 2D to 3D is trivial. Comparing with other mass-preserving methods, the additional computational cost of ICLS is small. This is due to the simple algebraic expression of $\boldsymbol{u}_c$ (Eq. (25)), and only one solve of Eq. (14) is required; whereas a typical VOF-coupling method involves solving another set of transport equations [25], or reconstructing the interface by an iterative procedure [27].

In summary, the ICLS method proceeds by performing the following steps:

1. Advect ϕ^n from time t^n to t^{n+1} with Eq. (7), using the flow velocity $\boldsymbol{u}^n$.
2. If reinitialization will be executed (otherwise, go to step 3):
   (a) Perform mass correction with Eq. (14), using $\boldsymbol{u}_c$ from Eq. (25).
   (b) Reinitialize ϕ^{n+1} with Eq. (11).
3. Exit the level set solver.

Deforming circle. To assess the performance of ICLS on mass conservation, we test the standard benchmark of a circle deformed by a single vortex. Here, the circle of radius 0.15 is initially centered at $(x, y) = (0.5, 0.75)$ of a $[0, 1]\times[0, 1]$ box. The velocity is imposed directly and can be obtained from the stream function

$$\psi(x, y, t) = \frac{1}{\pi}\sin^2(\pi x)\sin^2(\pi y)\cos\left(\frac{\pi t}{T}\right),$$

where T is traditionally set to 8. Under this flow, the circle will be stretched to maximum at $t = T/2$ and rewound to its initial condition at $t = T$. Although formulated simply, accurately transporting the interface without mass loss is a difficult task.

We perform this test on three different meshes using the complete level set solver: HOUC5 is used for the level set advection, WENO5 is used for reinitialization every 5 to 20 time steps, the mass correction is performed every 5 to 10 time steps; and the time step is chosen such that $\Delta t/\Delta x = 0.32$. Fig. 5 shows the shapes of the filament/circle at $t = 4$ and $t = 8$ at various resolutions. From the upper panel, it is clearly seen that the filament has a longer tail and head due to mass correction; as we increase the resolution, the difference becomes smaller. The lower panel of Fig. 5 depicts the final

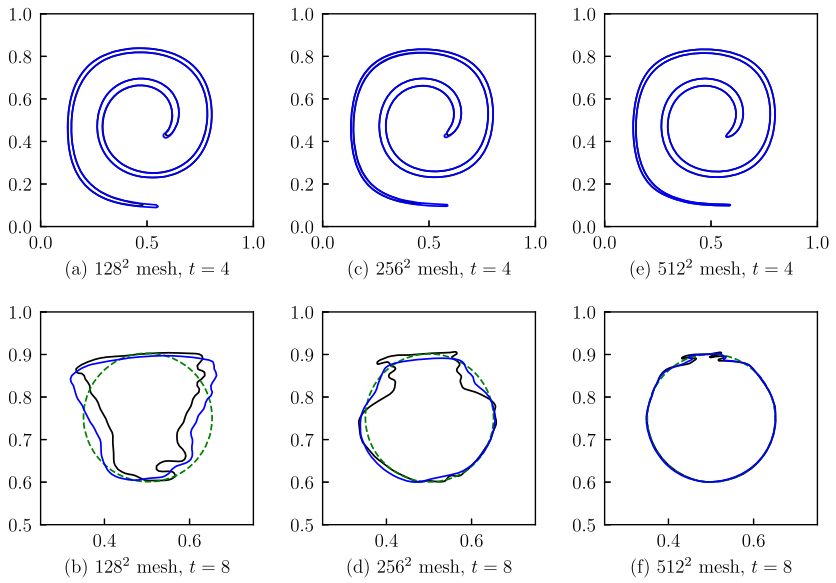


Fig. 5. Interface at $t = 4$ and $t = 8$ for different meshes. The solid black lines indicate simulations without mass correction, the solid blue lines indicate simulations with the current mass correction method, the green dashed lines in (b)(d)(f) indicate the original circle. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

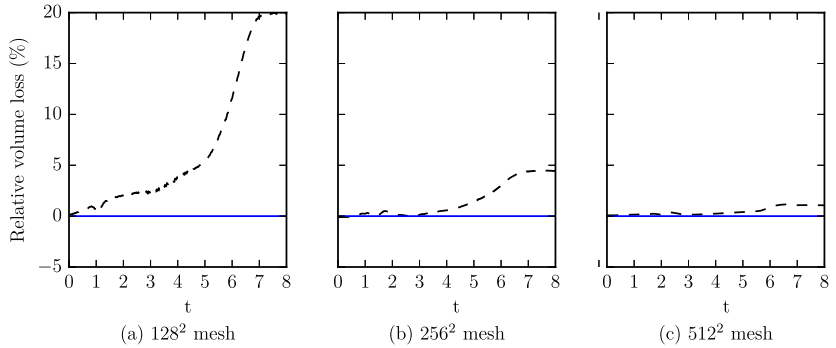


Fig. 6. Relative volume loss for three different meshes. Dashed lines indicate simulations without mass correction; solid lines indicate simulations with mass correction.

shapes, ideally the initial circle if the motion is totally passive. Some artifacts are visible due to the fact that the filament is always under-resolved at the maximum stretching and the level set will automatically merge the characteristics to yield an entropy solution [17]. We note that the final outcome can be tuned by modifying the frequency of the reinitialization/mass correction, a trade-off between the appearance and the mass loss. However, the objective here is to demonstrate the mass conservation enforced by ICLS, which is clearly illustrated in Fig. 6. For passive transport involving large deformations, we recommend particle-based methods [24]. Examples of droplets/bubbles in physical conditions using ICLS will be shown in the validations (Sec. 6.5) and applications (Sec. 7) below.

5. Curvature computation

Curvature computation is crucial to interfacial flows in the presence of surface tension, as inaccurate curvature can result in unphysical spurious currents [23,37], and even more so in our case when we apply curvature-dependent interface corrections. In this section, we first briefly describe the calculation of cell-center curvatures; *i.e.*, the curvature evaluated at the same nodal position as the level set function. Then, we introduce a geometric approach for the estimation of interface curvatures corresponding to the zero level set. The second step is specially tailored to the ghost fluid method that will be presented in Sec. 6.2.

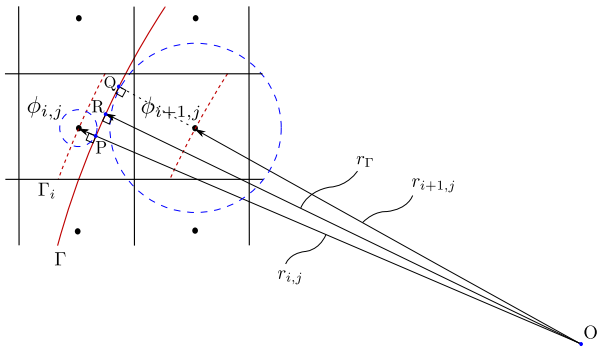


Fig. 7. Estimation of the interface's curvature from neighboring cells.

5.1. Cell-center curvature

From Eq. (6), the curvature κ can be evaluated as

$$\kappa = -\frac{\phi_{yy}\phi_x^2 + \phi_{xx}\phi_y^2 - 2\phi_x\phi_y\phi_{xy}}{\left(\phi_x^2 + \phi_y^2\right)^{3/2}} \tag{26}$$

and as

$$\kappa_M = -\frac{\left\{\begin{array}{r}(\phi_{yy}+\phi_{zz})\phi_x^2 + (\phi_{xx}+\phi_{zz})\phi_y^2 + (\phi_{xx}+\phi_{yy})\phi_z^2 \\ -2\phi_x\phi_y\phi_{xy} - 2\phi_x\phi_z\phi_{xz} - 2\phi_y\phi_z\phi_{yz}\end{array}\right\}}{\left(\phi_x^2 + \phi_y^2 + \phi_z^2\right)^{3/2}} \tag{27}$$

in 2D and 3D Cartesian coordinates, respectively, where the subscript M denotes the mean curvature [17]. The curvature can be determined from these expressions using simple central finite-differences. It has to be noted, however, that such evaluation of κ involves second derivatives of the level set field $\phi(\boldsymbol{x})$. As a consequence, if the calculation of ϕ is only second-order accurate, the resulting κ will be of order zero. To nonetheless retain a grid converging κ, one can use the compact least-squares scheme proposed by Marchandise et al. [54]. Their approach provides a second-order, grid converging evaluation of the cell-center curvature. It moreover smears out undesired high frequency oscillations possibly introduced by the velocity field. A similar procedure has also been adopted in other works [37,27].

The principle of the least squares approach is to solve an over-determined linear system, $\boldsymbol{Ax} = \boldsymbol{b}$, where $\boldsymbol{A}$ is a matrix built from the local coordinates, $\boldsymbol{x}$ is a unknown array containing the reconstructed level set values and its spatial derivatives, and $\boldsymbol{b}$ is the original level set field. The detailed descriptions can be found in [54]. Here, we only note that the level set function remains unmodified after this step. From a practical point of view, provided the mesh considered is uniform in all directions, the pseudo-inverse of the matrix $\boldsymbol{A}$ only needs to be evaluated once and applied close to the interface. Therefore, the computational cost of this least-squares calculation is negligible.

5.2. Interface curvature

The least-squares approach described in the previous section only allows one to compute the nodal curvature κ of the level set field ϕ. For computations using the GFM (Sec. 6.2), one might however require an accurate evaluation of the curvature at the exact location of the interface. Provided a grid-converging cell-center curvature, the actual curvature at the interface can be interpolated from its neighboring cells weighted by the level set [55,56]. Here we present a slightly different but robust algorithm to estimate the interface curvature, with a straight-forward geometrical interpretation.

2D estimation. Suppose the interface Γ cuts through two adjacent cells, (i, j) and $(i+1, j)$, where the cell-center curvatures $\kappa_{i,j}$ and $\kappa_{i+1,j}$ are known. In 2D, we can determine the radius of curvature at each cell directly from

$$\kappa_{i,j} = -\frac{1}{r_{i,j}}, \qquad \kappa_{i+1,j} = -\frac{1}{r_{i+1,j}}, \tag{28}$$

as illustrated in Fig. 7. Since the level set is defined as the signed distance to the interface, Γ must be tangent to a circle of radius $|\phi_{i,j}|$ centered at (i, j), and parallel to the contour line of $\Gamma_i = \{\boldsymbol{x}|\phi = \phi_{i,j}\}$ (otherwise they will not remain equidistant). We also know Γ lies between (i, j) and $(i+1, j)$, then it must pass through P (see Fig. 7). Since Γ and Γ_i are parallel and there is only one line normal to both curves passing through P, $r_{i,j}$ and OP must originate from the same point, O. Then we get

Table 1
Grid convergence of the current interface curvature calculation in both 2D and 3D.

Grid points per diameter	16	32	48	64
L_∞ 2D	1.144×10^{-2}	2.904×10^{-3}	1.285×10^{-3}	7.227×10^{-4}
L_∞ 3D	1.527×10^{-2}	3.888×10^{-3}	1.732×10^{-3}	9.753×10^{-4}

$$|OP| = r_{i,j} - s_\Gamma \phi_{i,j}. \tag{29}$$

where s_Γ is a sign function equal to 1 if the interface wrapping the negative level set is convex, and equal to -1 if concave.

The same argument holds for cell $(i+1, j)$, which yields $|OQ| = r_{i+1,j} - s_\Gamma \phi_{i+1,j}$. We can therefore write the radius of the interface curvature between (i, j) and $(i+1, j)$ as

$$r_\Gamma = \frac{|OP| + |OQ|}{2}, \tag{30}$$

so that the interface curvature becomes

$$\kappa_\Gamma = \frac{2}{\kappa_{i,j}^{-1} + \kappa_{i+1,j}^{-1} + s_\Gamma(\phi_{i,j} + \phi_{i+1,j})}. \tag{31}$$

The above derivation provides a relation between the interface curvature and that at the adjacent cell-centers in the x direction. Similar results can be obtained in the y direction (*e.g.* between $\phi_{i,j}$ and $\phi_{i,j-1}$). The assumptions we have made here are 1) the cell-center curvatures are accurate and 2) the interface curvatures at P and Q are the same, so that OP and OQ are co-centered (or, $|OP| \approx |OQ| \approx |OR|$). The second assumption is essentially a sub-cell approximation, and we expect it to be valid as long as the interface is well-resolved. One exception we have found is when two interfaces are closer than about $2\Delta x$, the local level set field will develop "corners". In that case, the cell-center curvatures are erroneous and the underlying assumptions we require here are not fulfilled. We do not discuss that case in the present paper. However, we demonstrate in the next section that a second-order convergence is achieved when the interface is resolved.

3D estimation. In three dimensions, the mean curvature of a surface can be written as

$$\kappa_\Gamma = -\left(\frac{1}{r_{\Gamma 1}} + \frac{1}{r_{\Gamma 2}}\right), \tag{32}$$

where $r_{\Gamma 1}$ and $r_{\Gamma 2}$ are the two principal radii corresponding to the maximal and minimal planar radius of curvature. Note that we do not need to approximate the interface as a sphere since there is always a plane where the previous picture (Fig. 7) holds. Under the same assumption as for the 2D case, that the interface at P and Q have the same principal radii (hence the same curvature), one can again relate the nodal curvatures to their nearby interface as

$$\begin{aligned} \kappa_{i,j,k} &= -\left(\frac{1}{r_{\Gamma 1} + s_\Gamma \phi_{i,j,k}} + \frac{1}{r_{\Gamma 2} + s_\Gamma \phi_{i,j,k}}\right), \\ \kappa_{i+1,j,k} &= -\left(\frac{1}{r_{\Gamma 1} + s_\Gamma \phi_{i+1,j,k}} + \frac{1}{r_{\Gamma 2} + s_\Gamma \phi_{i+1,j,k}}\right), \end{aligned} \tag{33}$$

where s_Γ is the same sign function defined for the 2D case. Comparing equations (32) and (33), it is natural to expand Eq. (33) into a Taylor series and to approximate the interface curvature directly as

$$\kappa_\Gamma = \frac{\epsilon_{i+1}\kappa_i - \epsilon_i \kappa_{i+1}}{\epsilon_{i+1} - \epsilon_i} + O(\epsilon_i^2, \epsilon_{i+1}^2), \tag{34}$$

where

$$\epsilon_i = s_\Gamma \phi_{i,j,k}. \tag{35}$$

Since the level set must change sign across the interface, Eq. (34) is always defined and it reduces to the exact value if the cell center happens to be on the interface. Similarly, the whole procedure is repeated in the y and z directions.

Finally, in order to ensure a robust estimation, we perform an additional quadratic least squares approximation on the curvature field near the interface, similar to [54]. This procedure takes place before the 3D estimation (Eq. (34)), and essentially improves the accuracy of cell-center curvatures by removing possible high-frequency noise. We note that the second averaging is optional, and different methods can be found in literature to evaluate the cell-center curvatures [50]. In the present paper, the least squares approach mentioned in Sec. 5.1 is used for all the cases.

To assess the accuracy of our interface curvature estimation, we calculate the L_∞ norm of a circle/sphere of radius 0.25 centered in a unit square/cube. Table 1 summarizes the error after one step of the calculations on different resolutions, which are also plotted in Fig. 8. Clearly, second-order convergence is achieved in both 2D and 3D cases.

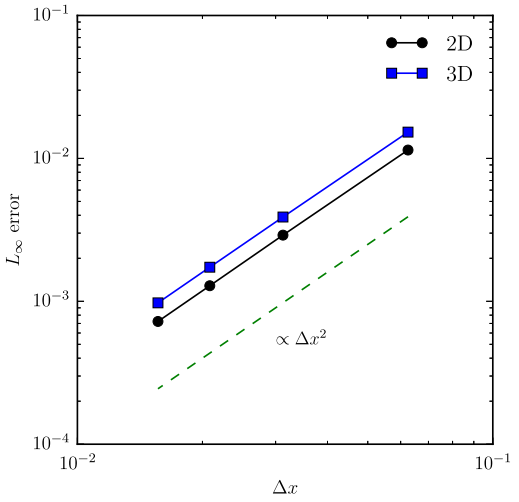


Fig. 8. Second order convergence of the interface curvature computation in both 2D and 3D.

6. Solution of the Navier–Stokes equations

In this section, we outline the flow solver developed from that of Breugem [57] for particle-laden flows. After advancing the level set from ϕ^n to ϕ^{n+1}, the density and viscosity fields are updated by

$$\rho^{n+1} = \rho_1 H_s(\phi^{n+1}) + \rho_2(1 - H_s(\phi^{n+1})), \tag{36a}$$

$$\mu^{n+1} = \mu_1 H_s(\phi^{n+1}) + \mu_2(1 - H_s(\phi^{n+1})), \tag{36b}$$

where

$$H_s(\phi) = \begin{cases} 1 & \text{if} \quad \phi > 0 \\ 0 & \text{otherwise}, \end{cases} \tag{37}$$

is a simple step function.

Next, a prediction velocity $\boldsymbol{u}^*$ is computed by defining $\mathbf{RU}^n$ as

$$\mathbf{RU}^n = -\nabla \cdot (\boldsymbol{u}^n \boldsymbol{u}^n) + \frac{1}{Re}\left(\frac{1}{\rho^{n+1}} \nabla \cdot \left[\mu^{n+1}(\nabla \boldsymbol{u}^n + (\nabla \boldsymbol{u}^n)^T)\right]\right) + \frac{1}{Fr}\boldsymbol{g}, \tag{38}$$

which is the right-hand side of the momentum equation (1b) excluding the pressure gradient term. Integrating in time with the second-order Adams–Bashforth scheme (AB2) yields

$$\boldsymbol{u}^* = \boldsymbol{u}^n + \Delta t\left(\frac{3}{2}\mathbf{RU}^n - \frac{1}{2}\mathbf{RU}^{n-1}\right). \tag{39}$$

To enforce a divergence-free velocity field (Eq. (1a)), we proceed by solving the Poisson equation for the pressure as in the standard projection method [58], *i.e.*

$$\nabla \cdot \left(\frac{1}{\rho^{n+1}} \nabla p^{n+1}\right) = \frac{1}{\Delta t} \nabla \cdot \boldsymbol{u}^*. \tag{40}$$

The surface tension between two fluids is also computed during this step, using the ghost fluid method [32] (Sec. 6.2). This allows for an accurate and sharp evaluation of the pressure jump even at large density contrasts [37]. Finally, the velocity at the next time level is updated as

$$\boldsymbol{u}^{n+1} = \boldsymbol{u}^* - \frac{\Delta t}{\rho^{n+1}} \nabla p^{n+1}. \tag{41}$$

6.1. Fast pressure-correction method

In the above outline, a Poisson equation for the pressure (Eq. (40)) must be solved at each time step. This operation takes most of the computational time in the projection method, as it is usually solved iteratively. In addition, the operation count of iterative methods depends on the problem parameters (*e.g.* density ratio) and the convergence tolerance [13]. On

the other hand, Dong and Shen [59] recently developed a velocity-correction method that transforms the variable-coefficient Poisson equation into a constant-coefficient one. The essential idea is to split the pressure gradient term in Eq. (40) in two parts, one with constant coefficients, the other with variable coefficients, *i.e.*

$$\frac{1}{\rho^{n+1}}\nabla p^{n+1} \rightarrow \frac{1}{\rho_0}\nabla p^{n+1} + (\frac{1}{\rho^{n+1}} - \frac{1}{\rho_0})\nabla \hat{p}, \tag{42}$$

where $\rho_0 = min(\rho_1, \rho_2)$ and $\hat{p}$ is the approximate pressure at time level $n+1$. This splitting reduces to the exact form of Eq. (40) within the lower-density phase, while its validity in the higher-density phase and at the interface depends on the choice of $\hat{p}$. Later, Dodd and Ferrante [13] showed that by explicitly estimating $\hat{p}$ from two previous time levels as

$$\hat{p} = 2p^n - p^{n-1}, \tag{43}$$

the resulting velocity field in Eq. (41) will be second-order accurate in both space and time, independent of the interface advection method. Furthermore, if the computational domain includes periodic boundaries or can be represented by certain combination of homogeneous Dirichlet/Neumann conditions [38], the constant-coefficient part of Eq. (42) can be solved directly using Gauss elimination in the Fourier space. Such a FFT-based solver can lead to a speed-up of $10-40$ times, thus the name fast pressure-correction method (FastP*). Following this approach, Eqs. (40) and (41) are modified as

$$\nabla^2 p^{n+1} = \nabla \cdot \left[(1 - \frac{\rho_0}{\rho^{n+1}})\nabla \hat{p} \right] + \frac{\rho_0}{\Delta t}\nabla \cdot \boldsymbol{u}^* \tag{44}$$

and

$$\boldsymbol{u}^{n+1} = \boldsymbol{u}^* - \Delta t \left[\frac{1}{\rho_0}\nabla p^{n+1} + (\frac{1}{\rho^{n+1}} - \frac{1}{\rho_0})\nabla \hat{p} \right]. \tag{45}$$

6.2. Ghost fluid method

As discussed before, surface tension is commonly computed using the continuum surface force (CSF) model [31], in which the pressure jump across an interface is represented as a forcing term on the right-hand side of Eq. (1b). Despite its simplicity, CSF introduces an unfavorable smearing in the density and pressure profiles, resulting in an artificial spreading of the interface (typically over a thickness of $3\Delta x$). An alternative approach is the so-called ghost fluid method (GFM), originally developed by Fedkiw et al. [32] to capture the boundary conditions in the inviscid compressible Euler equations. Unlike CSF, GFM enables a numerical discretization of the gradient operator while preserving the discontinuity of the differentiated quantity. It was extended to viscous flows by Kang et al. [34] and has been successfully utilized in multiphase flow simulations, see *e.g.* [37,60,61].

Recall from Eq. (2) that the pressure jump has two components, one arising from the surface tension, the other from the viscosity difference of the two fluids. In [34], a complete algorithm is provided to compute the two contributions, making the density, viscosity, and pressure all sharp. However, having a sharp viscosity profile requires an extra step to evaluate the divergence of the deformation tensor (see Eq. (38)). That is, for cells adjacent to the interface, the second derivatives of the velocity must be evaluated using the techniques developed in [62,34]. However, rewriting Eq. (2) as

$$[p]_\Gamma = \frac{1}{Re}\left(\frac{\kappa}{Ca} + 2[\mu]_\Gamma \boldsymbol{n}^T \cdot \nabla \boldsymbol{u} \cdot \boldsymbol{n} \right), \tag{46}$$

reveals that surface tension is the dominant term when the Capillary number, $Ca = We/Re$, is small. For the applications we are interested in, *e.g.* colloidal droplets in microfluidic channels, Ca is of the order of 10^{-5}. Therefore, in the present implementation, we regularize the viscosity profile (*i.e.* replacing $H_s(\phi)$ in Eq. (36b) with $H_\epsilon(\phi)$ in Eq. (18)) and use GFM only for the pressure jump.

6.2.1. Spatial discretization

Eqs. (38), (44), and (45) are discretized on a standard staggered grid using a second-order conservative finite volume method. It is equivalent to central differences in all three directions if the mesh is uniform. A detailed description of the discretization of the individual terms can be found in [13], Sec. 2.2.1. For brevity, we show here only the 2D evaluations of ∇p and $\nabla^2 p$ due to GFM.

As sketched in Fig. 9, computing $\nabla^2 p$ at node (i, j) requires three entries of p in each direction. If CSF is used, all gradient terms can be evaluated with the straightforward central-difference, *i.e.*

$$(\nabla^2 p)_{i,j} = \frac{p^s_{i-1,j} - 2p^s_{i,j} + p^s_{i+1,j}}{\Delta x^2} + \frac{p^s_{i,j-1} - 2p^s_{i,j} + p^s_{i,j+1}}{\Delta y^2}. \tag{47}$$

However, the pressure at the cells adjacent to the interface will have to be smeared out; hence we denote them with p^s. In order for the pressure to be sharp, GFM creates an artificial fluid (the "ghost" fluid) and assumes that the discontinuity can

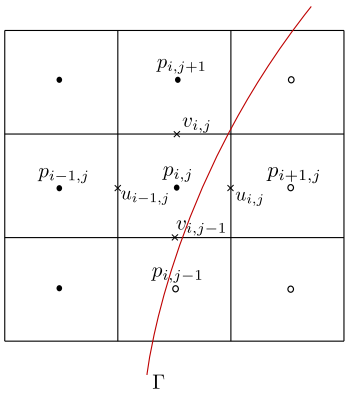


Fig. 9. Schematic of the 2D staggered grid where pressure locates at cell centers and velocity components locate at cell faces. The curved line specifies the interface Γ; filled and empty circles indicate discontinuous pressure (or density) values in phase 1 and 2, respectively.

be extended beyond the physical interface. That is, if we know the corresponding jumps of pressure, then its derivatives can be evaluated without smearing by removing such jumps. For the particular case depicted in Fig. 9, Eq. (47) can be re-written as (see [62] for the intermediate steps)

$$\begin{aligned}(\nabla^2 p)_{i,j} = &\frac{p_{i-1,j} - 2p_{i,j} + p_{i+1,j}}{\Delta x^2} - \frac{[p]_{i,j}}{\Delta x^2} - \frac{1}{\Delta x}\left[\frac{\partial p}{\partial x}\right]_{i+1/2,j} \\ &+ \frac{p_{i,j-1} - 2p_{i,j} + p_{i,j+1}}{\Delta y^2} - \frac{[p]_{i,j-1}}{\Delta y^2},\end{aligned} \tag{48}$$

where we recall $[\cdot]_{i,j}$ denotes the discontinuity from fluid 1 to fluid 2 at cell (i, j) (same for $[\cdot]_{i,j-1}$, *etc.*).

To determine the jump terms in Eq. (48), we first note that the velocity and its material derivatives across the interface of viscous flows are continuous [34,37], resulting in

$$\left[\frac{1}{\rho^{n+1}}\nabla p^{n+1}\right]_\Gamma = \mathbf{0}. \tag{49}$$

Furthermore, owing to the splitting that allows us to solve only for a constant-coefficient Poisson equation (Eq. (44)), Eqs. (42) and (49) lead to

$$\left[\frac{1}{\rho_0}\nabla p^{n+1}\right]_\Gamma + \left[(\frac{1}{\rho^{n+1}} - \frac{1}{\rho_0})\nabla \hat{p}\right]_\Gamma = \mathbf{0}, \tag{50}$$

which also implies that the pressure gradient terms are continuous everywhere (*e.g.* the subscript can be $(i+1/2, j)$), along any direction.

Denoting the right-hand side of Eq. (44) as RP, it is discretized as

$$\begin{aligned}RP_{i,j} = &\left((1 - \frac{\rho_0}{\rho^{n+1}_{i+1/2,j}})\frac{\partial \hat{p}}{\partial x}_{i+1/2,j} - (1 - \frac{\rho_0}{\rho^{n+1}_{i-1/2,j}})\frac{\partial \hat{p}}{\partial x}_{i-1/2,j}\right)/\Delta x \\ &+ \left((1 - \frac{\rho_0}{\rho^{n+1}_{i,j+1/2}})\frac{\partial \hat{p}}{\partial y}_{i,j+1/2} - (1 - \frac{\rho_0}{\rho^{n+1}_{i,j-1/2}})\frac{\partial \hat{p}}{\partial y}_{i,j-1/2}\right)/\Delta y \\ &- \frac{1}{\Delta x}\left[(1 - \frac{\rho_0}{\rho^{n+1}})\frac{\partial \hat{p}}{\partial x}\right]_{i+1/2,j} + \frac{\rho_0}{\Delta t}\left(\frac{u^*_{i,j} - u^*_{i-1,j}}{\Delta x} + \frac{v^*_{i,j} - v^*_{i,j-1}}{\Delta y}\right),\end{aligned} \tag{51}$$

again using GFM [62]. Comparing Eqs. (48) and (51), we note that the jump of the first derivatives cancels out recognizing Eq. (50). With a modified right-hand side, RP^*, defined as

$$RP^*_{i,j} = RP_{i,j} + \frac{1}{\Delta x}\left[(1 - \frac{\rho_0}{\rho^{n+1}})\frac{\partial \hat{p}}{\partial x}\right]_{i+1/2,j}, \tag{52}$$

the discrete form of Eq. (44) reduces to

$$\frac{p^{n+1}_{i-1,j} - 2p^{n+1}_{i,j} + p^{n+1}_{i+1,j}}{\Delta x^2} + \frac{p^{n+1}_{i,j-1} - 2p^{n+1}_{i,j} + p^{n+1}_{i,j+1}}{\Delta y^2} = \frac{[p]^{n+1}_{i,j}}{\Delta x^2} + \frac{[p]^{n+1}_{i,j-1}}{\Delta y^2} + RP^*_{i,j}. \tag{53}$$

Eq. (53) is still not ready to solve, since the pressure jumps for the first point away from the interface (*e.g.* $[p]_{i,j}^{n+1}$) are not known. Following [37], we perform a Taylor series expansion around Γ,

$$[p]_{i,j}^{n+1} = [p]_{\Gamma}^{n+1} + (x_i - x_\Gamma)\left[\frac{\partial p}{\partial x}\right]_{\Gamma}^{n+1} + O((x_i - x_\Gamma)^2), \tag{54}$$

where $[p]_{\Gamma}^{n+1} = \kappa_{\Gamma,x}/We$, and $\kappa_{\Gamma,x}$ is estimated from Eq. (31) in 2D and from Eq. (34) in 3D, along the x direction using $\phi_{i,j}^{n+1}$ and $\phi_{i+1,j}^{n+1}$. The jump of the pressure gradient at the interface can be similarly expanded at (i, j)

$$\left[\frac{\partial p}{\partial x}\right]_{\Gamma}^{n+1} = \left[\frac{\partial p}{\partial x}\right]_{i,j}^{n+1} + O(x_\Gamma - x_i), \tag{55}$$

resulting in

$$[p]_{i,j}^{n+1} = \frac{\kappa_{\Gamma,x}}{We} + (x_i - x_\Gamma)\left[\frac{\partial p}{\partial x}\right]_{i,j}^{n+1} + O((x_i - x_\Gamma)^2). \tag{56}$$

Using Eq. (50), we can re-write Eq. (56) as

$$[p]_{i,j}^{n+1} = \frac{\kappa_{\Gamma,x}}{We} + (x_i - x_\Gamma)\left[(1 - \frac{\rho_0}{\rho^{n+1}})\frac{\partial \hat{p}}{\partial x}\right]_{i,j} + O((x_i - x_\Gamma)^2), \tag{57}$$

where the jump term on the right-hand side can be explicitly calculated using the family of identities of the form [34]

$$[AB] = [A]\tilde{B} + \tilde{A}[B], \quad \tilde{A} = aA_1 + bA_2, \quad a + b = 1. \tag{58}$$

Although Eqs. (57) and (58) lead to a second-order pressure jump, it is much simpler to keep only the leading-order term, *i.e.*

$$[p]_{i,j}^{n+1} = \frac{\kappa_{\Gamma,x}}{We} + O(x_i - x_\Gamma). \tag{59}$$

This way, the pressure jump varies only with the local curvature, remains invariant across the interface, and is second-order accurate when the density is uniform. For the test cases shown below, Eq. (59) is used. Thus, the complete discretization of Eq. (44) reads

$$\frac{p_{i-1,j}^{n+1} - 2p_{i,j}^{n+1} + p_{i+1,j}^{n+1}}{\Delta x^2} + \frac{p_{i,j-1}^{n+1} - 2p_{i,j}^{n+1} + p_{i,j+1}^{n+1}}{\Delta y^2} = \frac{1}{We}\left(\frac{\kappa_{\Gamma,x}}{\Delta x^2} + \frac{\kappa_{\Gamma,y}}{\Delta y^2}\right) + RP_{i,j}^{*}, \tag{60}$$

with $RP_{i,j}^{*}$ defined in Eq. (52) corresponding to Fig. 9.

Clearly, the resulting linear system (Eq. (60)) has a standard positive definite, symmetric coefficient matrix, and it can be solved directly using the FFT-based fast Poisson solver (Sec. 6.1). Care should be exercised when a nodal point crosses the interface in more than one direction. In those cases, the interface curvature of each crossing direction may be different and it shall not be averaged. Otherwise, the projection (Eq. (44)) and correction (Eq. (45)) steps can become inconsistent, making the velocity not divergence-free. Additionally, when taking the gradient of the pressure-correction term; *e.g.* its derivative along the x direction, the correct discretization should be

$$\frac{\partial \hat{p}}{\partial x}_{i,j} = \frac{\left(\hat{p}_{i+1,j} - (2[p]_{i+1,j}^{n} - [p]_{i+1,j}^{n-1})\right) - \hat{p}_{i,j}}{\Delta x}. \tag{61}$$

After removing the jump, the divergence of the bracket term in Eq. (44) is evaluated in the same way as in [13].

Finally, we can re-write Eqs. (44) and (45) compactly as

$$\nabla^2 p^{n+1} = \nabla_g^2[p]_\Gamma + \nabla \cdot \left[(1 - \frac{\rho_0}{\rho^{n+1}})\nabla_g \hat{p}\right] + \frac{\rho_0}{\Delta t}\nabla \cdot \boldsymbol{u}^*, \tag{62}$$

$$\boldsymbol{u}^{n+1} = \boldsymbol{u}^* - \Delta t\left[\frac{1}{\rho_0}\nabla_g p^{n+1} + (\frac{1}{\rho^{n+1}} - \frac{1}{\rho_0})\nabla_g \hat{p}\right], \tag{63}$$

where ∇_g and $\nabla_g^2[p]_\Gamma$ denote, respectively, the gradient operator considering the jump and the extra jump terms from the laplacian operator due to GFM.

6.3. Time integration

In the current work, a second-order accurate Adams–Bashforth scheme is used for the time integration. The time step is restricted by convection, diffusion, surface tension, and gravity, due to our explicit treatment of these terms. As suggested in [34], the overall time step restriction is

$$\Delta t \leqslant 1 \Big/ \left(C_{CFL} + V_{CFL} + \sqrt{(C_{CFL} + V_{CFL})^2 + 4G_{CFL}^2 + 4S_{CFL}^2} \right), \tag{64}$$

where C_{CFL}, V_{CFL}, G_{CFL}, and S_{CFL} are the "speeds" due to convection, viscosity, gravity, and surface tension, respectively. Specifically, they are given as

$$C_{CFL} = \frac{|u|_{max}}{\Delta x} + \frac{|v|_{max}}{\Delta y} + \frac{|w|_{max}}{\Delta z}, \tag{65}$$

$$V_{CFL} = \frac{1}{Re} max\left(\frac{\mu_1}{\rho_1}, \frac{\mu_2}{\rho_2} \right) \left(\frac{2}{\Delta x^2} + \frac{2}{\Delta y^2} + \frac{2}{\Delta z^2} \right), \tag{66}$$

$$G_{CFL} = \sqrt{\frac{1}{Fr} \frac{|(1 - \frac{\rho_1 + \rho_2}{2\rho}) g|_{max}}{min(\Delta x, \Delta y, \Delta z)}}, \tag{67}$$

$$S_{CFL} = \sqrt{\frac{1}{We} \frac{|\kappa|_{max}}{min(\rho_1, \rho_2) \left[min(\Delta x, \Delta y \Delta z) \right]^2}}. \tag{68}$$

where $|\kappa|_{max}$ in (68) can be approximated by $1/\Delta x$ in 2D and $2/\Delta x$ in 3D, assuming Δx is the smallest grid spacing.

The reasons we choose an explicit temporal scheme rather than an implicit one are twofold. First, for applications involving a large density and viscosity contrast, the stability restriction imposed by surface tension is usually greater than that imposed by diffusion. Second, an implicit formulation of GFM has been admitted to be challenging to develop [37], and it was shown in a recent study [63] that a capillary time-step constraint exists, irrespective of the type of implementation, due to the temporal sampling of surface capillary waves. Fortunately, the fast pressure-correction method enables the use of FFT for the constant-coefficient Poisson equation and hence an accurate and fast solution of the two-fluid Navier–Stokes equation can be obtained.

6.4. Full solution procedure

We summarize the full solution procedure as follows:

1. Advance the interface explicitly from ϕ^n to ϕ^{n+1} using the ICLS, and update the density ρ^{n+1} and the viscosity μ^{n+1}.
2. Advance the velocity field explicitly from $\boldsymbol{u}^n$ to $\boldsymbol{u}^*$ with Eqs. (38) and (39).
3. Project the velocity field by solving the constant-coefficient Poisson Eq. (62) making use of the FastP* and the GFM.
4. Update the velocity from $\boldsymbol{u}^*$ to $\boldsymbol{u}^{n+1}$ explicitly with Eq. (63), again using the FastP* and the GFM.

6.5. Validations

In this section, we validate the coupled ICLS/NS solver using three benchmark examples with increasing complexities. Specifically, the first example verifies the discrete momentum balance for fluids of the same density and viscosity. This concerns the surface tension computed by the GFM using interface curvatures. Then, the density and viscosity ratios are significantly increased (up to 10^4) to test the combined FastP* and GFM. Using the same test, we also provide a convergence check of the complete flow solver. Finally, the overall accuracy is assessed by simulating a 3D bubble in comparison with experiments.

6.5.1. Spurious currents

A common problem in multiphase-flow simulations is the artificial velocity generated at the fluid interface due to errors in the curvature computation. To access the significance of such spurious currents, we test a stationary droplet of diameter $D = 0.4$ placed at the center of a unit box. The surface tension between the inner and outer fluid is $\sigma = 1$, the viscosity is uniformly $\mu = 0.1$, and the density ratio is 1. By changing the density ρ of both fluids, the Laplace number $La = \sigma \rho D / \mu^2$ can be varied. The spurious currents are thus determined from the resulting capillary number $Ca = |U_{max}| \mu / \sigma$ at a non-dimensional time $t\sigma/(\mu D) = 250$. Here, we compare the results on a 32×32 mesh with the GFM implementation by Desjardins et al. [37]. As listed in Table 2, the capillary numbers from both tests remain very small for all the Laplace numbers, with the present results being one-order smaller.

We also note that the spurious currents reported in Table 2 are obtained by performing the level set reinitialization at about every 100 time steps. However, if we turn off the reinitialization, such spurious velocity will eventually go to

Table 2
Dependence of spurious current capillary number Ca on the Laplace number for a static droplet with surface tension on a 32×32 mesh in comparison with Desjardins et al. [37].

La	12	120	1,200	12,000	120,000	1,200,000
Ca	2.85×10^{-6}	3.14×10^{-6}	3.63×10^{-6}	3.87×10^{-6}	3.41×10^{-6}	5.79×10^{-7}
Ca from [37]	4.54×10^{-5}	3.67×10^{-5}	3.62×10^{-5}	4.15×10^{-5}	3.75×10^{-5}	8.19×10^{-6}

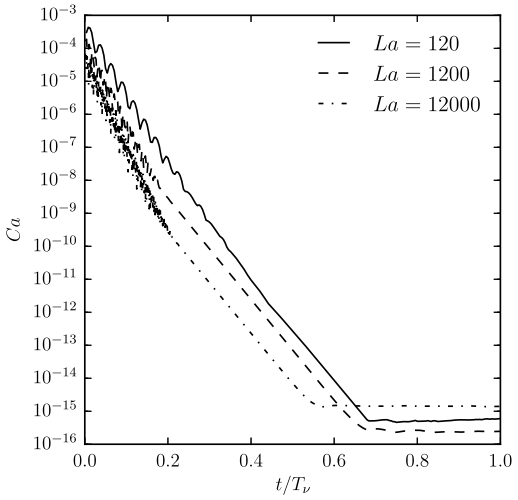


Fig. 10. Temporal evolution of the spurious currents without performing level set reinitialization at three Laplace numbers as in [64].

machine zero, as shown in Fig. 10, where time is non-dimensionalized with the viscous time scale, $T_\nu = \rho D^2/\mu$. The nearly exponential decay of Ca and the collapsing of the three curves are the result of the viscous damping of the spurious velocity, as the shape of droplet relaxes to its numerical equilibrium. Similar results are obtained and explained in greater detail in [64] using a balanced-force continuum-surface-force surface-tension formulation and the VOF. The result in Fig. 10 therefore validates the computation of the surface tension with the GFM.

6.5.2. Capillary wave

To verify the solver at large density and (dynamic) viscosity contrasts, we simulate a small-amplitude capillary wave for which there exists an analytical solution derived by Prosperetti [65]. Specifically, an initially sinusoidal interface is imposed between two immiscible, viscous fluids of infinite depth and lateral extent. When the lower fluid is heavier, the balance between inertia, viscosity, and surface tension results in a decaying free-surface wave. By requiring matching kinematic viscosity $\nu_u = \nu_l$ (u for upper, l for lower), the solution of the wave amplitude in terms of Laplace transforms can be inverted analytically and compared with the simulation results.

We set up our simulation in the same way as suggested in [13]. Here, two fluids of equal depth are placed in a 1×3 (64×192 grid points) domain, where the streamwise direction ($L = 1$) is periodic and the vertical direction ($H = 3$) wall-bounded. The interface has an initial wavelength of $\lambda = 1$ and an amplitude of $a_0 = 0.01$. With varying density ratios ρ_l/ρ_u, the non-dimensional parameters for the test are

$$Re = 100, \quad We = 1, \quad Fr = \infty, \quad \rho_l/\rho_u = 10 - 10,000, \quad \nu_l = \nu_u. \tag{69}$$

The CFL number $\Delta t/\Delta x$ is 2.5×10^{-2} for $\rho_l/\rho_u = 10$ and 10^2, and it is reduced to 2.5×10^{-3} for $\rho_l/\rho_u = 10^3$ and 2.5×10^{-4} for $\rho_l/\rho_u = 10^4$.

Fig. 11 shows the temporal evolution of the wave amplitude up to $t = 10$. The excellent agreement with Prosperetti's analytical solution [65] confirms the normal stress balance computed using the GFM. And accurate results at very large density contrasts are realized by combining the FastP* with GFM. Note that the dynamic viscosity ratio μ_l/μ_u also varies from 10 to 10^4. However, neglecting its contribution to the pressure jump by regularizing the viscosity profile yields accurate results since the Capillary number is small ($Ca = We/Re = 0.01$), as discussed in conjunction with Eq. (46).

6.5.3. Convergence

We continue to check the temporal and spatial convergence rates of the coupled ICLS/NS solver. Here, the same test problem as in Sec. 6.5.2 is used, with the non-dimensional parameters given as

$$Re = 500, \quad We = 1, \quad Fr = \infty, \quad H_0 = 0.05, \quad \rho_l/\rho_u = 20, \quad \mu_l/\mu_u = 20, \tag{70}$$

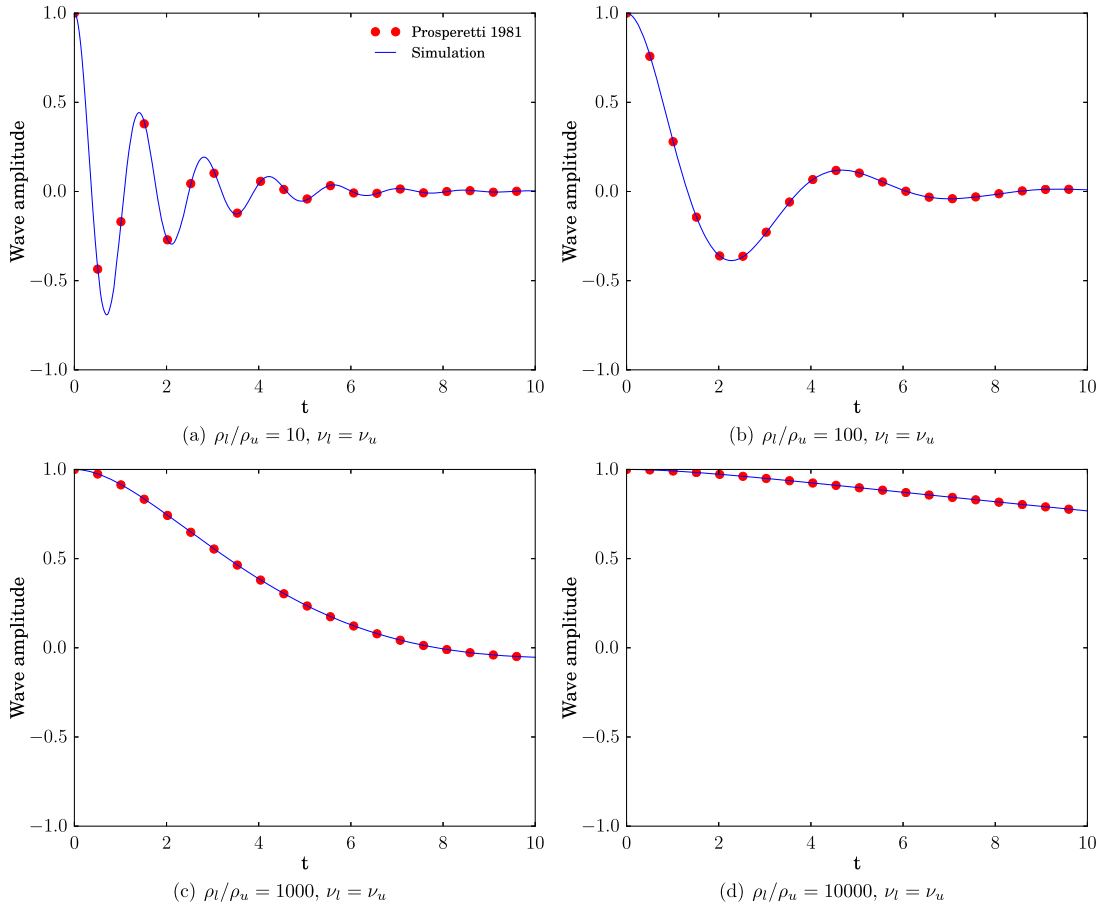


Fig. 11. Time development of the capillary wave amplitude (normalized to a_0) for increasing density ratios and matching kinematic viscosity in comparison with Prosperetti's analytical solution [65].

again following [13]. Placing the fluids in a 1×1 box, the flow is simulated under different time steps or on different meshes so that the errors can be computed between successive solutions.

Table 3 shows the convergence rates for the velocity component u and the pressure p in the L_2 norm. Here, the temporal convergence is evaluated at $t = 6.25 \times 10^{-2}$ on a 256^2 grid, by increasing the time step from $\Delta t = 4.88 \times 10^{-5}$ to $2\Delta t$ and $4\Delta t$. Two iterations of reinitialization are performed every $25 - 100$ time steps. The observed convergence rates for both velocity and pressure is between first and second order. Considering that we use RK3 for LS and AB2 for NS, the reduced convergence is probably due to the reinitialization that perturbs the interface. Changing the frequency of the reinitialization, we indeed observe different convergence rates (they can also exceed second order if the density ratio is 1, not shown). Next, the spatial convergence is obtained by successively refining the grid from 32^2 to 64^2 to 128^2. Using the same time step $\Delta t = 4.88 \times 10^{-5}$ and interpolating the solution to the coarse grid after one solve, the results display nearly second order convergence for the velocity and a super-convergence for the pressure. We note that the GFM has been proven convergent (but without a rate) for variable-coefficient Poisson equations [33]. Our results thus show improved accuracy in two fluid problems, when a constant-coefficient Poisson equation is obtained by combining the GFM with the FastP*.

6.5.4. Rising bubble

Finally, we compute four cases of a rising bubble to access the overall accuracy of the current ICLS/NS solver in 3D in the presence of moderate deformations. Originally documented by Grace [66], it was observed that a single gas bubble rising in quiescent liquid has four characteristic shapes: spherical, ellipsoidal, skirted, or dimpled. The governing non-dimensional numbers are the Morton number M, Eotvos number Eo (sometimes referred to as the Bond number), and the terminal Reynolds number Re_t, defined as

$$M = \frac{g\mu_l^4}{\rho_l \sigma^3}, \quad Eo = \frac{\Delta\rho g d^2}{\sigma}, \quad Re_t = \frac{\rho_l U_\infty d}{\mu_l}, \tag{71}$$

Table 3
Temporal and spatial convergence rates for the velocity component u and the pressure p.

	$L_2^{4\Delta t,2\Delta t}$	$L_2^{2\Delta t,\Delta t}$	Rate	$L_2^{4\Delta x,2\Delta x}$	$L_2^{2\Delta x,\Delta x}$	Rate
u	2.46×10^{-8}	1.03×10^{-8}	1.19	2.16×10^{-7}	5.95×10^{-8}	1.82
p	1.13×10^{-6}	3.85×10^{-7}	1.46	3.25×10^{-3}	6.11×10^{-4}	2.67

Table 4
Comparison of computed terminal Reynolds number (Re_C) and experimental terminal Reynolds number (Re_G) obtained from the Grace diagram [66] under four different Morton (M) and Eotvos (Eo) numbers.

Case	Bubble regime	M	Eo	Re_G	Re_C	Mass loss (%)
(a)	Spherical	1×10^{-3}	1	1.7	1.73	9.86×10^{-5}
(b)	Ellipsoidal	0.1	10	4.6	4.57	3.32×10^{-4}
(c)	Skirted	1	100	20.0	19.21	1.64×10^{-2}
(d)	Dimpled	1000	100	1.5	1.71	3.28×10^{-3}

(a) Spherical (b) Ellipsoidal (c) Skirted (d) Dimpled

Fig. 12. Bubble shapes resulting from different Morton (M) and Eotvos (Eo) numbers, as indicated in Table 4.

where d is the bubble diameter, $\Delta\rho$ is the density difference, U_∞ is the terminal velocity of the bubble, and the subscripts l and g denote, in order, the liquid and gas phase. The Morton and Eotvos number are defined purely by the material properties of the chosen fluids, while the terminal Reynolds number provides a measure of the steady-state bubble velocity.

Table 4 lists the four representative cases we select for the simulations. A spherical bubble of diameter $d = 1$ is centered in a domain of size $(L_x \times L_y \times L_z) = (3d \times 6d \times 3d)$. A grid of $96 \times 192 \times 96$ points is used, giving the bubble an initial resolution of 32 points per diameter. Periodic boundary conditions are imposed in the x (spanwise) and y (rising) directions whereas no friction, no penetration is enforced in the z direction. As suggested by Annaland et al. [67], a ratio of 100 between the density and viscosity of liquid and gas is sufficiently high to approximate such gas-liquid systems, leading to $\Delta\rho \approx \rho_l$. Re and We in Eq. (3) can thus be obtained from M and Eo as

$$Re = \left(\frac{Eo^3}{M}\right)^{1/4}, \quad We = Eo. \tag{72}$$

The CFL number, $\Delta t/\Delta x$, is 1.6×10^{-4} for cases (a), (b), and (d), and 1.6×10^{-3} for case (c). The simulation is integrated in time up to $t = 10$ to ensure the bubble reaches nearly steady state.

The results of the bubble terminal velocities are presented in Table 4. The difference between the computed Reynolds, Re_C, and the terminal Reynolds, Re_G, measured by Grace [66] remains small for all four cases. The bubble mass is conserved, with a maximal mass loss of about 0.02% found in the skirted case, where the bubble undergoes a large and rapid deformation. The corresponding bubble shapes are illustrated in Fig. 12, which clearly displays spherical, ellipsoidal, skirted, and dimpled shapes. We can therefore conclude that the dynamics of a single rising bubble is well-captured.

7. Droplet interactions

A unique feature of colloidal suspensions is the interaction between neighboring droplets, displaying fascinating behaviors such as self-assembly, self-replication, *etc.* The reason for such interactions is rather complex; it often arises from a combination of fluid mechanical effects and physicochemical properties of the substance. To study the droplet interactions in the present ICLS/NS framework, we provide in this section a hydrodynamic model for the depletion forces. The method is a natural extension of the LS and GFM, and we demonstrate the clustering of droplets in various structures from a dumbbell to a face-centered cubic crystal.

7.1. Extension to multiple level set

The level set method discussed so far involves one marker function; we call it single level set (SLS) method. Thanks to its Eulerian nature, SLS can describe many droplets at the same time, provided that they do not need to be distinguished from each other. On the other hand, SLS can also be extended to multiple level set (MLS), so that each droplet has its own color function. This has several benefits including distinction and tracking of each droplet, independent curvature computation, and ability to prevent numerical coalescence, *etc.*. Furthermore, with the narrow band approach [48,40] and the various

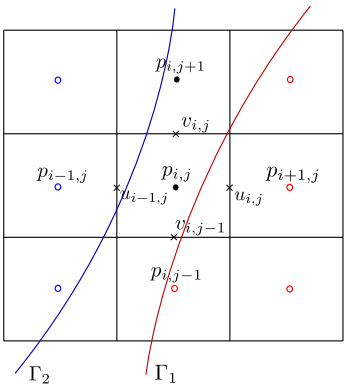


Fig. 13. Pressure jump in the presence of multiple interfaces within two grid cells. Red and blue circles indicate nodal pressure in droplet 1 and 2, respectively. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

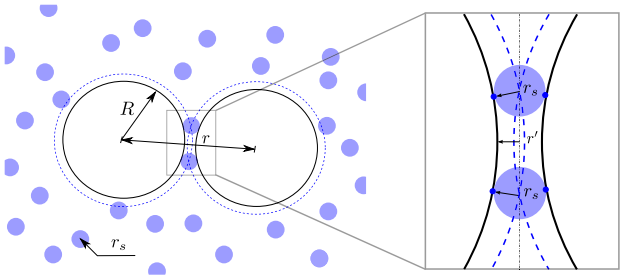


Fig. 14. Depletion of surfactant micelles of radius r_s between larger colloidal droplets of radius R, separated by distance r. The dashed lines around larger spheres represent the region from which the centers of small spheres are excluded. They overlap when $r \leqslant 2R + 2r_s$. Inset: a zoom-in sketch of two droplets near contact.

other techniques introduced in Sec. 1 [41,42], the additional computational and memory cost as the number of the level set functions increases is limited.

The extension from SLS to MLS is straightforward. Assuming no droplets will overlap, each level set function is simply advected successively. When two droplets get close (typically within two grid cells, see Fig. 13), the pressure jump across each interface needs to be considered and superimposed. That is, Eq. (48) (corresponding to Fig. 9) should be modified as

$$\begin{aligned}(\nabla^2 p)_{i,j} = &\frac{p_{i-1,j} - 2p_{i,j} + p_{i+1,j}}{\Delta x^2} - 2\frac{[p]_{i,j}}{\Delta x^2} - \frac{1}{\Delta x}\left[\frac{\partial p}{\partial x}\right]_{i+1/2,j} + \frac{1}{\Delta x}\left[\frac{\partial p}{\partial x}\right]_{i-1/2,j} \\ &+ \frac{p_{i,j-1} - 2p_{i,j} + p_{i,j+1}}{\Delta y^2} - \frac{[p]_{i,j-1}}{\Delta y^2},\end{aligned} \tag{73}$$

Similarly, all the jumps should be removed consistently when computing the pressure gradient in the subsequent step. The above modification applies to both SLS and MLS, as the compact formulas (Eqs. (62) and (63)) remain the same; although MLS is clearly more accurate in resolving the near field structure.

7.2. Near-field interactions

As introduced earlier, colloidal droplets transported in microfluidic devices are subject to various forces, a typical of which is the depletion force. The depletion force arises from the exclusion of the surfactant micelles in the colloidal suspension. It is often characterized as a near-field attracting potential [68,5], and plays a key role in the droplet dynamics [7, 69]. Below, we first provide a brief background on the colloidal theory of the depletion potential, then present a numerical model to enforce the depletion force using MLS and GFM.

7.2.1. The colloidal theory of the depletion potential

The original depletion potential model proposed by Asakura and Oosawa [68] assumes the surfactant micelles as non-interacting hard-spheres. As sketched in Fig. 14, a suspension of such small spheres around the large colloidal droplets creates an osmotic pressure on the droplet surface. When the distance between two droplets is less than the diameter of the surfactant micelles, there will be a pressure defect due to the exclusion of the micelles, thus creating an attracting force. Integrating this force with respect to the inter-droplet distance r leads to a potential energy

$$U(r) = \begin{cases} \infty & \text{if} \quad r \leqslant 2R \\ -p_{os} V_{ex} & \text{if} \quad 2R < r \leqslant 2R + 2r_s \\ 0 & \text{otherwise,} \end{cases} \tag{74}$$

where V_{ex} is the excluded volume and p_{os} is the osmotic pressure. For spherical droplets, V_{ex} can be calculated analytically

$$V_{ex}(r) = \frac{4\pi (R + r_s)^3}{3} \left[1 - \frac{3r}{4(R + r_s)} + \frac{r^3}{16(R + r_s)^3} \right], \tag{75}$$

where R and r_s are, respectively, the radii of the big and small spheres. The osmotic pressure is given as

$$p_{os} = nkT, \tag{76}$$

where n is the number density of the small spheres, k is the Boltzmann constant, and T is the temperature. The negative sign in Eq. (74) corresponds to the tendency of the system to reduce its potential energy as the overlap increases. This is equivalent to increasing the total entropy of the small spheres [70], and it provides a physical description of the depletion force even when the droplets are deformable, or when p_{os} cannot be expressed by the van't Hoff's formula (Eq. (76)) [68].

7.2.2. A hydrodynamic model for the depletion force

Based on the above theory, the depletion force acting on a droplet is simply the derivative of the depletion potential, *i.e.* $F(r) = dU/dr = -p_{os} dV_{ex}/dr$. However, dV_{ex}/dr is not always straightforward to evaluate for non-spherical droplets; and unlike rigid-body dynamics, $F(r)$ cannot be applied directly to the motion of a liquid drop. In order to induce locally an aggregation, we take a closer look at the overlap region. As illustrated in Fig. 14, when the surface distance between two colloidal droplets is less than $2r_s$, there is a small area in which the osmotic pressure is subject to a jump. Assuming the concentration of the surfactant micelles changes abruptly, it resembles the jump of the Laplace pressure; however, it will not generate any flow if the pressure is uniform in the depleted region. On the contrary, if the osmotic pressure varies continuously within the overlap, *i.e.* $p' = p'(r')$, then we can write it as a Taylor-series expansion from $r' = r_s$

$$p'(r'/r_s) = p'(1) + \left(\frac{r'}{r_s} - 1 \right) \frac{\partial p'}{\partial r'/r_s}, \tag{77}$$

where the distance to the droplet surface r' is normalized by the surfactant micelle radius. An expansion of the osmotic pressure with the distance corresponds to a gradient of the micelle concentration near the gap. And if the micelle is much smaller than the droplet, as it is in many microfluidic devices [7], the gradient will be very sharp. Conversely, when the distance to the surface varies slowly, such as in the gap of a droplet and a flat wall, a uniform pressure will be recovered. Furthermore, a favorable pressure gradient from the overlap center will generate an outflow, pulling the droplets towards each other. Hence, Eq. (77) provides a hydrodynamic model for the depletion force.

In Eq. (77), the gradient of the osmotic pressure $\partial p'/\partial(r'/r_s)$ is not known *a priori*. It can be obtained by equating the depletion force acting on one droplet, *i.e.*

$$-p_{os} A_{ex} = \int_{\Omega} \left(p'(1) - p'(r'/r_s) \right) dS, \tag{78}$$

where A_{ex} is the effective area of the overlap Ω. Assuming a constant $\partial p'/\partial(r'/r_s)$, the above yields a linear dependence of the osmotic pressure on r'. Note that this is not the same as p' varying linearly with the distance to the overlap center (see Fig. 14). A description of the implementation and verification will be shown in the next section.

7.2.3. A MLS/GFM-based method for computing the depletion force

Provided a hydrodynamic model for the depletion force between two droplets, we can easily generalize it to multiple droplets using the MLS. Thanks to the distance information embedded in the level set functions, it is straightforward to identify the overlap region of arbitrary geometries. Furthermore, as the jump of the osmotic pressure occurs only across the overlap shell, we can define

$$[p']_{\Omega} = p'(r'/r_s) - p'(1), \tag{79}$$

similar to the Laplace pressure jump $[p]_{\Gamma}$ implemented by the GFM. Based on these observations, we propose a numerical method to compute the depletion force as laid out in Algorithm 1.

The overall idea of Algorithm 1 is to enforce the depletion attraction in the projection step through the use of MLS and GFM. Specifically, we first locate the overlap region of a pair of droplets with its own level set function, and define r' as the average of the two distances. Then, Eq. (78) can be integrated numerically to obtain $\partial p'/\partial(r'/r_s)$, which together with Eqs. (77) and (79) gives $[p']_{\Omega}$. This variable pressure jump manifests itself as a modification term on the right-hand side of

Algorithm 1: A pseudo code for computing the depletion force.

```
Enter the pressure solver. Compute the right-hand side of Eq. (62).
for m = 1 : (N − 1) do
    Get the level set for droplet m, φ_m.
    for n = (m + 1) : N do
        Get the level set for droplet n, φ_n.
        where φ_m < r_s and φ_n < r_s do r' = (φ_m + φ_n)/2, tag as overlap.
        Compute [p']_Ω from Eqs. (78) and (79) within overlap.
        forall the i, j, k do
            if entering overlap then
                Add the osmotic pressure jump [p']_{i,j,k}.
            else
                Remove the osmotic pressure jump [p']_{i,j,k}.
            end
        end
    end
end
Solve for p^{n+1} regularly using the FastP* and GFM. Exit the pressure solver.
```

Eq. (62), allowing us to use GFM to impose it across a sharp overlap shell. The resulting flow is divergence-free provided that all the jump terms are removed consistently in the correction step. Therefore, Eqs. (62) and (63) are re-formulated as[2]

$$\nabla^2 p^{n+1} = \nabla_g^2([p]_\Gamma + [p']_\Omega) + \nabla\cdot\left[\left(1-\frac{\rho_0}{\rho^{n+1}}\right)\nabla_g\hat{p}\right] + \frac{\rho_0}{\Delta t}\nabla\cdot\boldsymbol{u}^*, \tag{80}$$

and

$$\boldsymbol{u}^{n+1} = \boldsymbol{u}^* - \Delta t\left[\frac{1}{\rho_0}\nabla_g p^{n+1} + \left(\frac{1}{\rho^{n+1}} - \frac{1}{\rho_0}\right)\nabla_g\hat{p}\right]. \tag{81}$$

Approaching drops. We verify the depletion force model and its numerical implementation by simulating 2 to 14 approaching droplets in a quiescent fluid environment. Specifically, we set the droplet radius $R = 0.5$, the computational domain $3\times3\times3$, and the resolution $\Delta x = 1/32$. The radius of the surfactant micelle is set to be $r_s = 1/16$, corresponding to $2\Delta x$. The viscosity and density ratios of the droplet to the ambient fluid are both 1. The non-dimensional parameters are $La = 2000$ and $Fr = \infty$, leading to a reference Laplace pressure jump $p_\sigma = 80$ and neglected gravity. The uniform osmotic pressure is either 10 or 40.

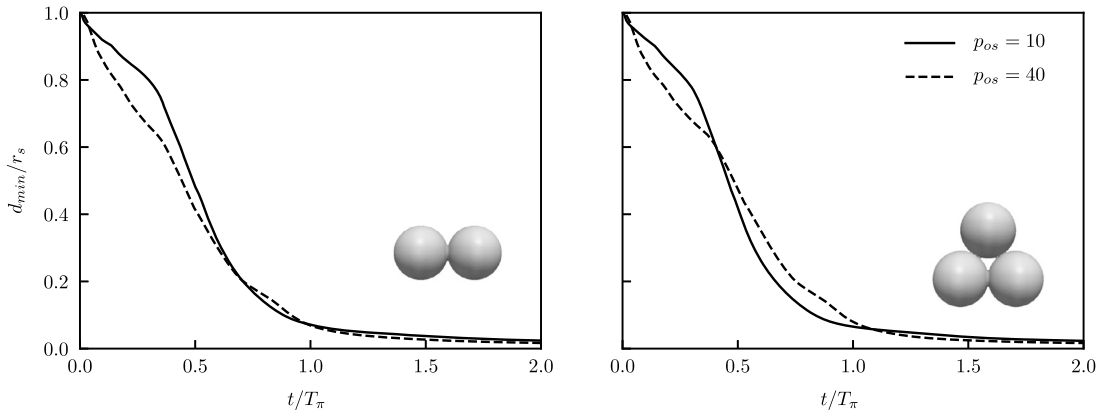
The temporal evolutions of the minimal surface distances in the case of two and three droplets are shown in Fig. 15. Here, time is scaled by a factor $T_\pi = (r_s/R)(p_\sigma/p_{os})$. The droplets, originally separated by a distance of r_s, get closer to the limit of the grid spacing at $t \approx T_\pi$. For the present study, we let the droplets aggregate without applying any repulsion models, except that the magnitude of the osmotic pressure is reduced when $d_{min}/r_s < 0.1$. The smooth approaching in all cases and the collapse of the distance curve clearly evidence an attracting depletion force. To assess the robustness of the method, we further tested clustering of droplets into shapes from a 2D diamond to a face-centered cubic (FCC) composed of 14 drops, illustrated here in Fig. 16. FCC represents the unit structure of one of the most compact sphere packings. Therefore, we can conclude that the hydrodynamic model implemented by the MLS/GFM-based method is accurate and robust in computing the depletion forces.

8. Conclusion

A numerical method mainly intended for the hydrodynamic simulations of colloidal droplets in microfluidic devices has been developed and validated. The code is based on an efficient and sharp solver of the incompressible, two-fluid Navier–Stokes equations, and uses a mass-conserving level set method to capture the fluid interface. This combination provides a general framework for any multiphase flow problems (see *e.g.* our recent study on jet instabilities [71]), and allows us to develop specific methods for the simulations of droplets in saturated surfactant suspensions with depletion forces as in the recent experiment in [7]. Particularly, we have developed or extended four numerical techniques to improve the general accuracy:

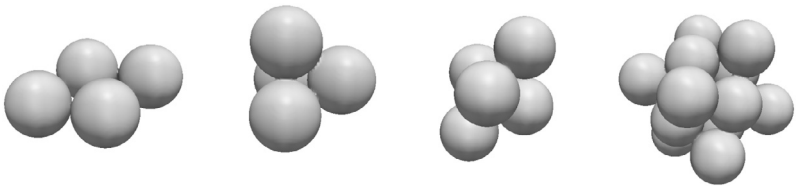
1. A mass-conserving, interface-correction level set method (ICLS) is proposed. As a standalone level set module, it is efficient, accurate, guarantees global mass conservation, and is simple to implement. It also enables corrections that can depend on the local curvature or any other parameter of interest.

[2] Eqs. (63) and (81) are identical in form; however, $[p']_\Omega$ has to be removed when evaluating $\nabla_g p^{n+1}$ and $\nabla_g\hat{p}$ in Eq. (81), as it is done in Eq. (61).



(a) Two droplets (b) Three droplets

Fig. 15. Minimal distance between the droplet surfaces as function of time in the presence of depletion forces proportional to $p_{os} = 10$ (solid line) and $p_{os} = 40$ (dashed line). Simulation of (a) two droplets and (b) three droplets suspended in an initially quiescent fluid. Due to symmetry, only the minimal distance is plotted.



(a) Diamond (b) Tetrahedron (c) Quintuplet (d) FCC

Fig. 16. Examples of droplet clusters of different structures.

2. A geometric estimation of the interface curvature based on nodal curvatures is introduced. As an important ingredient both for the mass correction (ICLS) and the surface tension computation, we show that the calculation converges in second-order both in 2D and 3D, and can lead to machine-zero spurious currents for a stationary 2D droplet.
3. The ghost fluid method (GFM) for the computation of surface tension is combined with the FastP* method [13]. This enables the use of FFT-based solvers for a direct pressure solve, and can accurately account for surface tension at large density ratios.
4. A ghost fluid/multiple level set (GFM/MLS-based) method is also proposed to compute the interaction force caused by depletion potentials between multiple droplets or between droplets and a nearby wall. The approach can possibly be extended to account for surfactant diffusion at the interface and in the liquid.

The last technique applies specifically to the simulation of colloidal droplets in microfluidic devices. This will enable us to further explore the effects of the near-field interactions as those observed experimentally in [7], and potentially improve the design of microfluidic devices. In addition, the combination of the GFM for sharp interfaces and the FastP* method [13] can be exploited for the simulations of droplet in turbulent flows as in [72], adding an accurate representation of evaporation thanks to the ICLS approach proposed here.

Acknowledgements

The work is supported by the Microflusa project. This effort receives funding from the European Union Horizon 2020 research and innovation programme under Grant Agreement No. 664823. O.T. acknowledges support from Swedish Research Council (VR) through the Grant Nr. 2013-5789. L.B. and J.-C.L. also acknowledge financial support by the European Research Council grant, no. ERC-2013-CoG-616186, TRITOS. The computer time was provided by SNIC (Swedish National Infrastructure for Computing). Last but not least, Z.G. thanks Mehdi Niazi, Michael Dodd, Walter Fornari, Dr. Olivier Desjardins, Dr. Sébastien Tanguy, Dr. Marcus Herrmann, and Dr. David Salac for interesting and helpful discussions.

Appendix A. Discretization error of $\int_\Gamma \boldsymbol{n} \cdot \boldsymbol{u}_c d\Gamma$

Similar to [73], we define the discretization error

$$E = \left| \left(\prod_{k=1}^{d} \Delta x_k \right) \sum_{j \in Z^d} \hat{\delta}_\epsilon(\Gamma, g, \boldsymbol{x}_j) - \int_\Gamma \boldsymbol{n} \cdot \boldsymbol{u}_c d\Gamma \right|, \tag{A.1}$$

where $\hat{\delta}_\epsilon$ is a Dirac delta function of variable strength g supported on the surface Γ, and $\boldsymbol{x} \in \mathbb{R}^d$. Following the derivations in Sec. 4, the extension of g to $\mathbb{R}^d$ is provided by Eq. (22), allowing one to write

$$E = \left| \left(\prod_{k=1}^{d} \Delta x_k \right) \sum_{j \in Z^d} \frac{\delta V}{\delta t} \frac{f_s \delta_\epsilon(\phi(\boldsymbol{x}_j)) |\nabla \phi(\boldsymbol{x}_j)|}{A_f} - \int_\Gamma \boldsymbol{n} \cdot \boldsymbol{u}_c d\Gamma \right|. \tag{A.2}$$

Here, $\delta_\epsilon(\phi)$ is a one dimensional regularized delta function depending on the level set ϕ, and the expression is simplified noting that $\boldsymbol{n} \cdot \nabla \phi = |\nabla \phi|$ (it does not have to be a distance function). By definition, $A_f = \int_\Gamma f_s \delta_\epsilon(\phi) |\nabla \phi| d\Gamma$, discretely reducing Eq. (A.2) to

$$E = \left| \frac{\delta V}{\delta t} - \int_\Gamma \boldsymbol{n} \cdot \boldsymbol{u}_c d\Gamma \right|. \tag{A.3}$$

Comparing with Eq. (13), it is obvious that $E = 0$. That is, the discretization error of $\int_\Gamma \boldsymbol{n} \cdot \boldsymbol{u}_c d\Gamma$ used in the mass correction is identically zero, independent of the choice of the regularized delta function.

References

[1] Y. Xia, B. Gates, Y. Yin, Y. Lu, Monodispersed colloidal spheres: old materials with new applications, Adv. Mater. 12 (2000) 693–713.
[2] O.D. Velev, S. Gupta, Materials fabricated by micro- and nanoparticle assembly – the challenging path from science to engineering, Adv. Mater. 21 (2009) 1897–1905.
[3] F. Li, D.P. Josephson, A. Stein, Colloidal assembly: the road from particles to colloidal molecules and crystals, Angew. Chem., Int. Ed. 50 (2011) 360–388.
[4] S. Sacanna, D.J. Pine, Shape-anisotropic colloids: building blocks for complex assemblies, Curr. Opin. Colloid Interface Sci. 16 (2011) 96–105.
[5] J. Mewis, N.J. Wagner, Colloidal Suspension Rheology, Cambridge University Press, New York, USA, 2012.
[6] Gi-Ra Yi, D.J. Pine, S. Sacanna, Recent progress on patchy colloids and their self-assembly, J. Phys. Condens. Matter 25 (2013) 193101.
[7] B. Shen, J. Ricourvier, F. Malloggi, P. Tabeling, Designing colloidal molecules with microfluidics, Adv. Sci. (2016) 1600012.
[8] C. Pozrikidis, Boundary Integral and Singularity Methods for Linearized Viscous Flow, Cambridge University Press, New York, USA, 1992.
[9] A. Kumar, M.D. Graham, Accelerated boundary integral method for multiphase flow in non-periodic geometries, J. Comput. Phys. 231 (2012) 6682–6713.
[10] L. Zhu, C. Rorai, M. Dhrubaditya, L. Brandt, A microfluidic device to sort capsules by deformability: a numerical study, Soft Matter 10 (2014) 7705–7711.
[11] M. Wörner, Numerical modeling of multiphase flows in microfluidics and micro process engineering: a review of methods and applications, Microfluid. Nanofluid. 12 (2012) 841–886.
[12] C. Galusinski, P. Vigneaus, On stability condition for bifluid flows with surface tension: application to microfluidics, J. Comput. Phys. 227 (2008) 6140–6164.
[13] M.S. Dodd, A. Ferrante, A fast pressure-correction method for incompressible two-fluid flows, J. Comput. Phys. 273 (2014) 416–434.
[14] C.S. Peskin, Flow patterns around heart valves: a numerical method, J. Comput. Phys. 10 (1972) 252–271.
[15] M. Uhlmann, A immersed boundary method with direct forcing for simulation of particulate flows, J. Comput. Phys. 209 (2005) 448–476.
[16] R. Scardovelli, S. Zaleski, Direct numerical simulation of free-surface and interfacial flow, Annu. Rev. Fluid Mech. 31 (1999) 567–603.
[17] J.A. Sethian, Level Set Method and Fast Marching Method, Cambridge University Press, New York, USA, 1999.
[18] M. Sussman, P. Smereka, S. Osher, A level set approach for computing solutions to incompressible two-phase flow, J. Comput. Phys. 114 (1994) 146–159.
[19] R.R. Nourgaliev, T.G. Theofanous, High-fidelity interface tracking in compressible flows: unlimited anchored adaptive level set, J. Comput. Phys. 227 (2007) 836–866.
[20] G. Russo, P. Smereka, A remark on computing distance functions, J. Comput. Phys. 163 (2000) 51–67.
[21] J. Strain, Tree methods for moving interfaces, J. Comput. Phys. 151 (1999) 616–648.
[22] C. Min, F. Gibou, A second order accurate level set method on non-graded adaptive cartesian grids, J. Comput. Phys. 225 (2007) 300–321.
[23] M. Herrmann, A balanced force refined level set grid method for two-phase flows on unstructured flow solver grids, J. Comput. Phys. 227 (2008) 2674–2706.
[24] D. Enright, R. Fedkiw, J. Ferziger, I. Mitchell, A hybrid particle level set method for improved interface capturing, J. Comput. Phys. 183 (2002) 83–116.
[25] M. Sussman, E.G. Puckett, A coupled level set and volume-of-fluid method for computing 3d and axisymmetric incompressible two-phase flows, J. Comput. Phys. 162 (2000) 301–337.
[26] S.P. van der Pijl, A. Segal, C. Vuik, P. Wesseling, Computing three-dimensional two-phase flows with a mass-conserving level set method, Comput. Vis. Sci. 11 (2008) 221–235.
[27] K. Luo, C. Shao, Y. Yang, J. Fan, An mass conserving level set method for detailed numerical simulation of liquid atomization, J. Comput. Phys. 298 (2015) 495–519.
[28] M. Sussman, E. Fatemi, An efficient, interface-preserving level set redistancing algorithm and its application to interfacial incompressible fluid flow, J. Sci. Comput. 20 (1997) 1165–1191.
[29] D. Salac, A general, mass-preserving Navier–Stokes projection method, Comput. Phys. Commun. 204 (2016) 97–106.
[30] E. Olsson, G. Kreiss, A conservative level set method for two phase flow, J. Comput. Phys. 210 (2005) 225–246.
[31] J. Brackbill, D. Kothe, C. Zemach, A continuum method for modeling surface tension, J. Comput. Phys. 100 (1992) 335–354.
[32] R.P. Fedkiw, T. Aslam, B. Merriman, S. Osher, A non-oscillatory eulerian approach to interfaces in multimaterial flows (the ghost fluid method), J. Comput. Phys. 152 (1999) 457–492.
[33] X.-D. Liu, T.C. Sideris, Convergence of the ghost fluid method for elliptic equations with interfaces, Math. Comput. 72 (2003) 1731–1746.
[34] M. Kang, R.P. Fedkiw, X.-D. Liu, A boundary condition capturing method for multiphase incompressible flow, J. Sci. Comput. 15 (2000) 323–360.
[35] B. Lalanne, L.R. Villegas, S. Tanguy, F. Risso, On the computation of viscous terms for incompressible two-phase flows with level set/ghost fluid method, J. Comput. Phys. 301 (2015) 289–307.
[36] S. Popinet, Numerical models of surface tension, Annu. Rev. Fluid Mech. 50 (2018) 49–75.

[37] O. Desjardins, V. Moureau, H. Pitsch, An accurate conservative level set/ghost fluid method for simulating turbulent atomization, J. Comput. Phys. 227 (2008) 8395–8416.
[38] U. Schumann, R.A. Sweet, Fast fourier transforms for direct solution of Poisson's equation with staggered boundary conditions, J. Comput. Phys. 75 (1988) 123–137.
[39] P.S. Costa, Cans (canonical Navier–Stokes), https://github.com/p-costa/CaNS. (Accessed 11 October 2017).
[40] D. Peng, B. Merriman, S. Osher, H. Zhao, M. Kang, A pde-based fast local level set method, J. Comput. Phys. 155 (1999) 410–438.
[41] M.B. Nielsen, K. Museth, Dynamic tubular grid: an efficient data structure and algorithms for high resolution level sets, J. Sci. Comput. 26 (2006) 261–299.
[42] E. Brun, A. Guittet, F. Gibou, A local level-set method using a hash table data structure, J. Comput. Phys. 231 (2012) 2528–2536.
[43] X.-D. Liu, S. Osher, T. Chan, Weighted essentially non-oscillatory schemes, J. Comput. Phys. 115 (1994) 200–213.
[44] W. Mulder, S. Osher, J.A. Sethian, Computing interface motion in compressible gas dynamics, J. Comput. Phys. 100 (1992) 209–228.
[45] J. Strain, Semi-lagrangian methods for level set equations, J. Comput. Phys. 151 (2) (1999) 498–533.
[46] G. Velmurugana, E.M. Kolahdouzb, D. Salac, Level set jet schemes for stiff advection equations: the semijet method, arXiv:1509.0283, 2016.
[47] C.-W. Shu, S. Osher, Efficient implementation of essentially non-oscillatory shock-capturing schemes, J. Comput. Phys. 77 (1988) 439–471.
[48] D. Adalsteinsson, J.A. Sethian, A fast level set method for propagating interfaces, J. Comput. Phys. 111 (2) (1995) 269–277.
[49] S.T. Zalesak, Fully multidimensional flux-corrected transport algorithms for fluids, J. Comput. Phys. 31 (1979) 335–362.
[50] A. du Chéné, C. Min, F. Gibou, Second-order accurate computation of curvatures in a level set framework using novel high-order reinitialization schemes, J. Sci. Comput. 35 (2008) 114–131.
[51] S. Chen, B. Merriman, S. Osher, P. Smereka, A simple level set method for solving stefan problems, J. Comput. Phys. 135 (1997) 8–29.
[52] D. Adalsteinsson, J.A. Sethian, The fastconstruction of extension velocities in level set methods, J. Comput. Phys. 148 (1999) 2–22.
[53] M. Aanjaneya, S. Patkar, R. Fedkiw, A monolithic mass tracking formulation for bubbles in incompressible flow, J. Comput. Phys. 247 (2013) 17–61.
[54] E. Marchandise, P. Geuzaine, N. Chevaugeon, J.-F. Remacle, A stabilized finite element method using a discontinuous level set approach for the computation of bubble dynamics, J. Comput. Phys. 225 (2007) 949–974.
[55] M.M. Francois, S.J. Cummins, E.D. Dendy, D.B. Kothe, J.M. Sicilian, M.W. Williams, A balanced force algorithm for continuous and sharp interfacial surface tension models within a volume tracking framework, J. Comput. Phys. 213 (2006) 141–173.
[56] J. Luo, X.Y. Hu, N.A. Adams, A conservative sharp interface method for incompressible multiphase flows, J. Comput. Phys. 284 (2015) 547–565.
[57] W.-P. Breugem, Numerical simulation of drop impact on a liquid-liquid interface with a multiple marker front-capturing method, J. Comput. Phys. 228 (2012) 4444–4467.
[58] A.J. Chorin, Numerical solution of the Navier–Stokes equations, Math. Comput. 22 (1968) 745–762.
[59] S. Dong, J. Shen, A time-stepping scheme involving constant coefficient matrices for phase-field incompressible flows with large density ratios, J. Comput. Phys. 231 (2012) 5788–5804.
[60] E. Coyajee, B.J. Boersma, A second-order accurate immersed boundary method for fully resolved simulations of particle-laden flows, J. Comput. Phys. 231 (2009) 4469–4498.
[61] S. Tanguy, A. Berlemont, Application of a level set method for simulations of droplet collisions, Int. J. Multiph. Flow 31 (2005) 1015–1035.
[62] X.-D. Liu, R.P. Fedkiw, M. Kang, A boundary condition capturing method for Poisson's equation on irregular domains, J. Comput. Phys. 160 (2000) 151–178.
[63] F. Denner, B.G.M. van Wachem, Numerical time-step restrictions as a result of capillary waves, J. Comput. Phys. 285 (2015) 24–40.
[64] S. Popinet, An accurate adaptive solver for surface-tension-driven interfacial flows, J. Comput. Phys. 228 (2009) 5838–5866.
[65] A. Prosperetti, Motion of two superposed viscous fluids, Phys. Fluids 24 (1981) 1217.
[66] J.R. Grace, Shapes and velocities of bubbles rising in infinite liquids, Trans. Instit. Chem. Eng. 51 (1973) 116–120.
[67] M. van Sint Annaland, N.G. Deen, J.A.M. Kuipers, Numerical simulation of gas bubbles behaviour using a three-dimensional volume of fluid method, Chem. Eng. Sci. 60 (2005) 2999–3011.
[68] S. Asakura, F. Oosawa, Interaction between particles suspended in solutions of macromolecules, J. Polym. Sci. XXXIII (1958) 183–192.
[69] B. Shen, Transport and Self-Assembly of Droplets in Microfluidic Devices, PhD thesis, Physics, Université Pierre et Marie Curie – Paris VI, 2014, English. MNT: 2014PA066499. Tel-01127629.
[70] P. Melby, A. Prevost, D.A. Egolf, J.S. Urbach, Depletion force in a bidisperse granular layer, Phys. Rev. E 76 (2007) 051307.
[71] O. Tammisola, J.-C. Loiseau, L. Brandt, Effect of viscosity ratio on the self-sustained instabilities in planar immiscible jets, Phys. Rev. Fluids 2 (3) (2017) 033903.
[72] M.S. Dodd, A. Ferrante, On the interaction of Taylor length scale size droplets and isotropic turbulence, J. Fluid Mech. 806 (2016) 356–412.
[73] B. Engquist, A.-K. Tornberg, R. Tsai, Discretization of dirac delta functions in level set methods, J. Comput. Phys. 207 (2005) 28–51.