Journal of Computational Physics 436 (2021) 110291

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

Piecewise linear interface-capturing volume-of-fluid method in axisymmetric cylindrical coordinates

Lubomír Bureš [a,b,*], Yohei Sato [b], Andreas Pautz [a,b]

[a] Swiss Federal Institute of Technology (EPFL), 1015 Lausanne, Switzerland
[b] Nuclear Energy and Safety, Paul Scherrer Institute, 5232 Villigen PSI, Switzerland

ARTICLE INFO

Article history:
Available online 22 March 2021

Keywords:
Volume-of-fluid (VOF) method
PLIC
Interface tracking
Interface reconstruction
Height-function method
Axisymmetric geometry

ABSTRACT

The paper describes a novel implementation of the piecewise linear interface-capturing volume-of-fluid method (PLIC-VOF) in axisymmetric cylindrical coordinates. The principal innovative feature involved in this work is that both the forward and inverse reconstruction problems are solved analytically, resulting in an appreciable speed-up in computing time in comparison with an iterative approach. All reconstruction formulae are introduced explicitly, and an example illustrating their derivation is included for clarity. The numerical implementation of the PLIC-VOF interface tracking method developed here is described in detail, as well as its coupling with the 3D incompressible Navier-Stokes solver PSI-BOIL, which features a finite-volume approach based on a fixed, rectangular grid. This coupling includes a method to calculate the surface tension force within an axisymmetric VOF framework by means of height functions. The method is first verified to ensure its correct implementation in the code, and to evaluate its performance, and several advection tests are employed to demonstrate successful solution of the basic transport problem. It is shown that convergence to equilibrium may be achieved for static problems, involving the control of parasitic currents, for a variety of grid arrangements, and for different material properties. Several axisymmetric dam-break and rising bubble problems, for which high-quality measured data are available, are also presented to serve as validation tests. In all cases, very good agreement of simulation results with experimental data has been recorded, and the dynamics of the problem well reproduced.

1. Introduction

In the field of multiphase computational fluid dynamics in the bubble and droplet regimes, capturing the interface between individual phases is crucial for performing numerical simulations of the phenomena, and capturing the morphology of the flow. A prominent example of an inherently mass-conservative Interface Tracking Method (ITM) is the Volume-of-Fluid (VOF) approach, in which the phases are represented in terms of a cell-wise phasic volume-fraction function, ϕ. For a computational cell of volume V_c [m^3], the volume fraction of Phase 1, ϕ_1, is defined as:

$$\phi_1 = \frac{V_1}{V_c}, \qquad (1)$$

[*] Corresponding author at: Nuclear Energy and Safety, Paul Scherrer Institute, 5232 Villigen PSI, Switzerland.
E-mail address: lubomir.bures@psi.ch (L. Bureš).

https://doi.org/10.1016/j.jcp.2021.110291

Principal symbols used

α	Normalised line constant [-]
β	Line constant [m]
η	Reduced axial coordinate [-]
κ	Curvature [1/m]
μ	Dynamic viscosity [Pa·s]
ρ	Density [kg/m^3]
σ	Surface tension [N/m]
ξ	Normalised coordinate [-]
c	Area fraction, colour [-]
ϕ	Volume fraction [-]
$\vec{n}$	Unit normal vector [-]
$\vec{m}$	General normal vector with positive components [-]
$\vec{u}=(u,w)$	Velocity [m/s], u - radial component, w - axial component
$\vec{x}=(x,z)$	Vector of radial (x) and axial (z) coordinates [m]
CFL	Courant number, $CFL=\max\left(u/\Delta x,\, w/\Delta z\right)\cdot\Delta t$

where V_1 is the volume of the cell occupied by Phase 1. If only two phases are present, as considered here, $\phi_2 = 1-\phi_1$; therefore, we drop the subscript 1 for brevity. The basis for any VOF method is then the solution of the continuity equation (written here in conservative form, with no source terms):

$$\frac{\partial\phi}{\partial t}+\nabla\cdot(\phi\vec{u})=0, \tag{2}$$

where $\vec{u}$ represents the velocity field. Equation (2) is valid for incompressible flow, for which the velocity field satisfies the divergence constraint:

$$\nabla\cdot\vec{u}=0. \tag{3}$$

The original algebraic VOF method [1], and many of its successors [2,3], attempt to solve Eq. (2) within the framework of the solver: i.e. without recourse to independent reconstruction of the interface. As a result, the interface region will have a finite thickness, determined by the underlying degree of grid refinement, often covering several computational cells. In contrast, geometric VOF methods involve advective transport of the volume fraction field based on a geometrical reconstruction of the interface within the computational cells. Typical reconstruction techniques include the simple linear interface calculation (SLIC) [4], which crudely assumes that the interface is parallel to one of the cell faces, and the more advanced piecewise linear interface calculation (PLIC) [5], which reconstructs the interface as one or multiple lines (in 2D), or one or multiple planes (in 3D), with arbitrary orientation with respect to that of the computational cells. Advection of the interface can then be simulated using directional-split or unsplit schemes: in the former approach, the phasic volume is transported in one direction at a time [6], while in the latter approach it is transported in all directions simultaneously [6,7].

Geometric PLIC-VOF methods were originally developed only for structured Cartesian meshes, due to the difficulty in deriving appropriate interface reconstruction formulae. Subsequently, however, the methods have been extended to axisymmetric cylindrical [8], full cylindrical [9], and axisymmetric spherical [10,11] structured geometries. PLIC-VOF methods have also been implemented on unstructured meshes at various levels of geometrical complexity, [12–20]; see also the recent review by Maric et al. [21].

The interfacial geometry is usually described in terms of the pair $\{\vec{n},\beta\}$, where $\vec{n}$ is the unit vector normal to the interface and β is the line (plane) constant satisfying the equation:

$$\vec{n}\cdot\vec{x}=\beta, \tag{4}$$

in which $\vec{x}$ is the vector of coordinates. Under the assumption that the normal vector is calculated via a numerical scheme, the volume fraction ϕ can be computed from β and $\vec{n}$ alone. This is the so-called *forward reconstruction*, which can be formally represented as:

$$\phi=f_{fwd}(\beta;\vec{n}). \tag{5}$$

Conversely, β can be computed from ϕ and $\vec{n}$ using the *inverse reconstruction*:

$$\beta=f_{inv}(\phi;\vec{n}). \tag{6}$$

Here, “forward” and “inverse” are chosen as descriptive labels only, which does not imply that the forward reconstruction is performed first, followed by the inverse reconstruction. Instead, both inverse and forward reconstruction are used in different parts of the overall PLIC-VOF advection algorithm: the former for positioning of the interface after volume fraction advection, and the latter typically for the calculation of fluid fluxes during the advection step [22].

For rectangular, cuboidal, triangular and tetrahedral meshes, simple relations connecting the volume fraction ϕ and the interfacial geometry exist, e.g. [22–24], and both the forward and inverse reconstructions can then be implemented analytically. The geometrical complexity associated with arbitrary interfaces intersecting general non-Cartesian meshes presents a challenge to the derivation of analogous relations in this context. Although Dyadechko and Shashkov [25] proved that relation (6) is strictly monotonic, and that the pair $\{\vec{n}, \phi\}$ is sufficient to compute β, no analytical relation for the inverse reconstruction currently exists for an arbitrarily shaped cell [21]. Thus the inverse problem is usually solved using either an iterative root-finding algorithm or by use of a *bracketing* method [21]. The pioneering work on this topic is the algorithm of Rider and Kothe [8,26], which uses bracketing to obtain an initial guess followed by a Brent iteration [27]. The iterative step within this algorithm was later replaced by a direct analytical solution utilising the results of the bracketing procedure [28]. A similar semi-analytic approach was developed by Diot et al. [29] for 2D planar and axisymmetric convex cells of arbitrary shape, and later extended to arbitrary convex 3D cells [30]; however, a Newton iteration was still necessary in axisymmetric geometry. The bracketing-analytic approach was further optimised by Lopez et al. [31], and extended to non-convex polyhedra by the same authors [20].

A specific example of a reconstruction method is the *Moment-of-Fluid* (MoF) reconstruction of Dyadechko and Shashkov [25,32], which has the ability to reconstruct the interface without reference to the details of the neighbouring cells, relying only on the volume fraction within the cells and geometric centroids of the phasic volumes; i.e. the first and second moments of the volume fraction distribution [25]. The MoF reconstruction algorithm involves minimisation of an objective function quantifying the distance of the reconstructed centroid to the tracked one; the original scheme employed a bracketing-analytic approach within the iterative procedure [25]. This technique was later extended to axisymmetric and full cylindrical coordinates [33,34]. The MoF reconstruction was also further enhanced by introducing analytical expressions for the objective function gradient relations for convex polyhedra [35], 3D rectangular hexahedra [36], and general polyhedra, as well as for axisymmetric polygons [37]. For 2D Cartesian meshes, Lemoine et al. [38] were able to replace the minimisation procedure by a direct inversion; the roots of the polynomials arising in the solution process were found iteratively followed by *a posteriori* elimination of the unsuitable ones. Although the authors hinted at application to rectangular meshes in other geometries, only 2D Cartesian meshes were treated in detail [38]. Note that the MoF method conceptually differs from the classical VOF approach due to the added necessity to explicitly track the advection of the phasic geometric centroids [25].

The iterative procedures usually employed for the inverse reconstruction, without the need for semi-analytical techniques, have also been developed further. These include the secant/bisection method proposed by Ahn and Shashkov [32,39], which remains in common use today [40], and the efficient Predicted-Newton method of Chen and Zhang [40].

In spite of the advancements in the development of solution techniques for the inverse reconstruction problem, a direct analytic solution still offers clear benefits in terms of speed and accuracy. For this reason, we present in this paper a novel implementation of the PLIC-VOF method in axisymmetric cylindrical coordinates including a closed-form analytic solution of the inverse reconstruction. Aside from simple laminar flows, the two-dimensional geometry assumption finds application for problems for which a high degree of axial symmetry can be guaranteed, it being currently too computationally expensive for a full three-dimensional simulation to be undertaken. For example, direct numerical simulation (DNS) of pool boiling, performed recently by Urbano et al. [41] using the Level Set implementation of the ITM [42], required more than 16 million computational cells in an axisymmetric configuration. Evidently, a 3D domain for the same problem would encompass more than 64 billion cells, unfeasible in the context of current computer technology.

To the best of our knowledge, our direct analytic solution of the inverse reconstruction problem has never been attempted for this geometry, and the full implementation of PLIC-VOF in an axisymmetric, cylindrical geometry has never been reported in the open literature. These aspects constitute the innovation of the present work.

The paper is organised as follows: in Section 2, the computational method is described. Firstly, analytic relations for both the forward and inverse problem in cylindrical, axisymmetric geometry are developed in Section 2.1, thereby removing the need for in-cell iterative reconstruction; appreciable speed-up with respect to the iterative approach is demonstrated. Then, in Section 2.2, details of the interface tracking algorithm and its numerical implementation in the code PSI-BOIL are described, and in Section 2.3 a height-function-based approach for the interfacial curvature calculation is presented. Finally, in Section 3, several verification and validation cases are presented, with final, overall conclusions presented in Section 4.

2. Mathematical model

The interface reconstruction method is detailed in Section 2.1, where the analytical relations for the forward and inverse reconstructions are developed. In Section 2.2, the interface tracking algorithm for two-phase flow is described. Finally, the height function method for the calculation of curvature in axisymmetric geometry is presented in Section 2.3.

2.1. Analytic relations for interface reconstruction

Axisymmetric representation of a three-dimensional object corresponds to its projection on a two-dimensional x-z plane, with x representing the radial and z the axial coordinate, respectively. This transformation is shown schematically in Fig. 1. It can be seen that a “linear” interface in an axisymmetric cylindrical geometry is, in the general case, a conical shell with the two extremes being a disc perpendicular to the z-axis and a cylindrical shell. Furthermore, projected rectangular cells represent hollow cylinders in three dimensions (see Fig. 1).

Within this framework, it is possible to define an *area fraction*, or *colour function*, c, of the projection with respect to a given reference control area. Due to the radius effects, the colour function is not equal to the volume fraction; nonetheless, c is a convenient variable to describe the interfacial geometry, as it allows us to re-use the theoretical framework developed

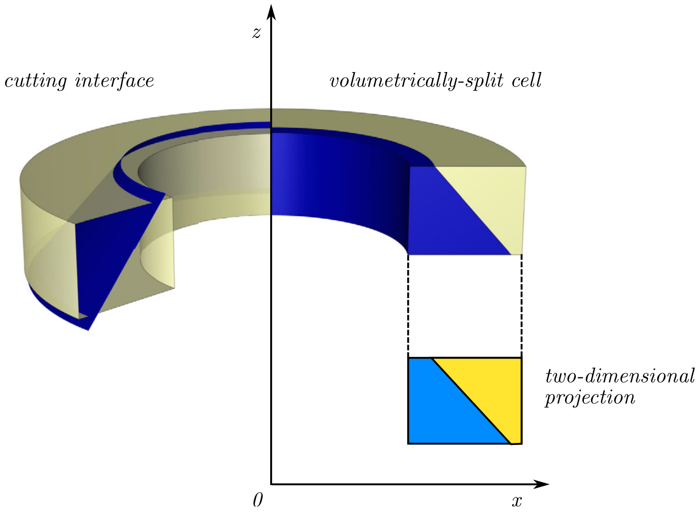


Fig. 1. Schematic representation of the projection of a typical computational cell in axisymmetric cylindrical geometry onto a two-dimensional *x*-*z* plane, with *x* representing the radial and *z* the axial coordinate, respectively.

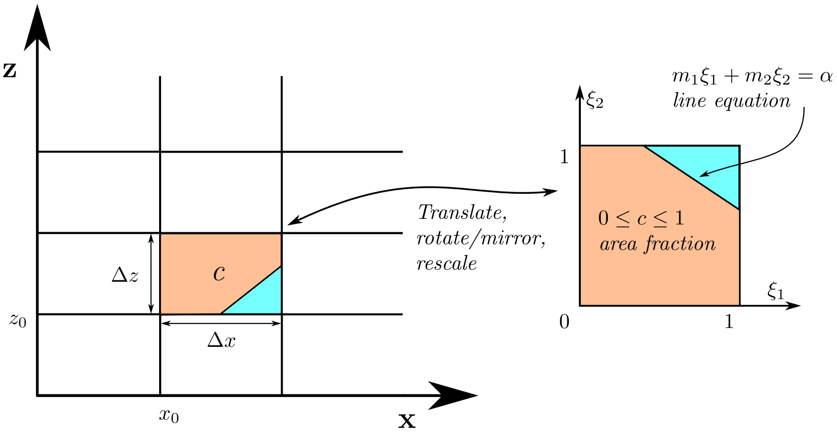


Fig. 2. Schematic representation of the 'standardisation' of the interfacial geometry.

for 2D Cartesian configurations, for which $c \equiv \phi$. Specifically, we can treat the *x*-*z* projections of the computational cells as rectangular cells in a 2D Cartesian geometry. As shown in Fig. 2, we can then translate, rotate/mirror and rescale the projected geometry such that Eq. (4) becomes:

$$m_1\xi_1 + m_2\xi_2 = \alpha, \tag{7}$$

where $m_1 \geq 0$, $m_2 \geq 0$, $m_1 + m_2 = 1$, $\xi_{1,2} \in [0,1]$ and $\alpha \in [0,1]$. For the example illustrated in Fig. 2, $|n_z| > |n_x|$, $n_x > 0$, and $n_z < 0$. Thus, using basic geometric manipulations:

$$m_1 = \frac{|n_z|\Delta z}{|n_z|\Delta z + |n_x|\Delta x};\ m_2 = \frac{|n_x|\Delta x}{|n_z|\Delta z + |n_x|\Delta x};\ \xi_1 = 1 - \frac{z - z_0}{\Delta z};\ \xi_2 = \frac{x - x_0}{\Delta x};$$

with $\vec{x}_0 = (x_0, z_0)$ and $\Delta\vec{x} = (\Delta x, \Delta z)$, as shown in Fig. 2. This 'standardised' representation of the interfacial geometry was first presented in [22,43] and makes use of the invariance of *c* with respect to the mirroring of the PLIC-line. With this representation, the results derived in [22] are valid, and will only be reproduced verbatim here for the sake of completeness:

$$c = \frac{\alpha^2}{2m(1-m)},\ 0 \leq \alpha < \alpha_1; \tag{8}$$

$$c = \frac{1}{1-m}\left(\alpha - \frac{m}{2}\right) = \frac{\alpha}{1-m} - c_1,\ \alpha_1 \leq \alpha \leq \frac{1}{2}; \tag{9}$$

$$\alpha = \sqrt{2m(1-m)c},\ 0 \leq c < c_1; \tag{10}$$

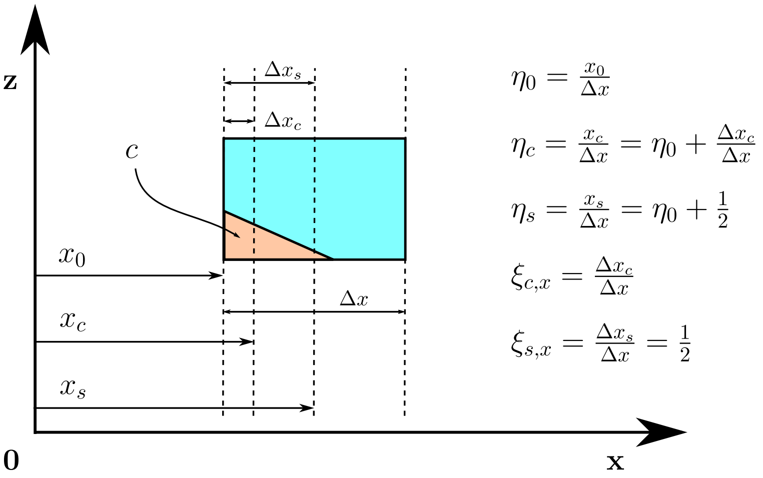

z
0
x
c
x_0
x_c
x_s
Δx_s
Δx_c
Δx
$\eta_0 = \frac{x_0}{\Delta x}$
$\eta_c = \frac{x_c}{\Delta x} = \eta_0 + \frac{\Delta x_c}{\Delta x}$
$\eta_s = \frac{x_s}{\Delta x} = \eta_0 + \frac{1}{2}$
$\xi_{c,x} = \frac{\Delta x_c}{\Delta x}$
$\xi_{s,x} = \frac{\Delta x_s}{\Delta x} = \frac{1}{2}$

Fig. 3. Schematic representation of the nomenclature used to describe the geometry. Note that while the areas to the left of $x = x_s$ and to the right of it are the same, their respective volumes of revolution are not equal due to the radius effects. (For interpretation of the colours in the figure(s), the reader is referred to the web version of this article.)

$$\alpha = (1-m)c + \frac{m}{2} = (1-m)(c + c_1),\ c_1 \le c \le \frac{1}{2}; \tag{11}$$

$$\alpha_1 = m; \tag{12}$$

$$c_1 = \frac{m}{2(1-m)}; \tag{13}$$

where $m = \min(m_1, m_2)$. Equations (8) and (9) represent the solution of the forward problem in 2D Cartesian geometry: that is, calculating the area fraction c from the line constant α given a vector $\vec{m}$ according to $c = f_{fwd}(\alpha; \vec{m})$. Equations (10) and (11) represent the solution of the inverse problem in 2D Cartesian geometry: that is, calculating the line constant α from the area fraction c given a vector $\vec{m}$ according to $\alpha = f_{inv}(c; \vec{m})$.

Even though the functions f_{fwd} and f_{inv} are monotonic [21,22], it can be seen that both the forward and the inverse problems feature two solution branches. The transition between these branches occurs at the point $[\alpha_1, c_1]$. The values of these two parameters can be found by equating the two branches of the solution, i.e. by solving (8) = (9) and (10) = (11).

Furthermore, it can be seen that Eqs. (8)-(11) are valid only on the intervals $c \in [0, \frac{1}{2}]$ and $\alpha \in [0, \frac{1}{2}]$. The complementary solutions of the problems for which $c \in (\frac{1}{2}, 1]$ and $\alpha \in (\frac{1}{2}, 1]$ can be achieved simply by point reflection around $[\frac{1}{2}, \frac{1}{2}]$ as:

$$c(\alpha, \vec{m}) = \tilde{c}(\tilde{\alpha}, \tilde{\vec{m}}) = 1 - c(1-\alpha, \vec{m}), \tag{14}$$

$$\alpha(c, \vec{m}) = \tilde{\alpha}(\tilde{c}, \tilde{\vec{m}}) = 1 - \alpha(1-c, \vec{m}), \tag{15}$$

as described in [22]. The above relations should be understood as descriptions of the transformations:

$$c \to \tilde{c} = 1 - c,$$
$$\alpha \to \tilde{\alpha} = 1 - \alpha,$$
$$\vec{m} \to \tilde{\vec{m}} = \vec{m},$$

which is equivalent to the swapping of the two phases in any given cell.

2.1.1. Forward problem in axisymmetric geometry (calculate ϕ from α and $\vec{n}$)

With reference to Fig. 3, the relation between the volume fraction ϕ in axisymmetric geometry and the area fraction or colour function c in its two-dimensional projection is given by the Second Theorem of Pappus [44] as:

$$\phi = c\frac{\eta_c}{\eta_s} = c\frac{\eta_0 + \xi_{c,x}}{\eta_0 + \xi_{s,x}} = c\frac{\eta_0 + \xi_{c,x}}{\eta_0 + 1/2} \equiv K(\alpha, \vec{n}, \eta_0) \cdot c, \tag{16}$$

where $\eta_c = x_c/\Delta x$ is the reduced radial coordinate of the geometric centroid of the orange-coloured area; likewise, η_s is the reduced radial coordinate of the geometric centroid of the bounding cell in the x-z coordinate system. The parameter η_0 is then the reduced radial coordinate of the inward cell face, i.e. the face closer to the axis of symmetry. Also, $\xi_{c,x}$ and $\xi_{s,x}$ are the reduced in-cell radial coordinates, these assuming values between 0 and 1. Again, the meanings of the individual symbols are illustrated in Fig. 3.

Table 1
Values of $\xi_{c,x}$ for Eq. (16).

	$m_x \le m_z$	$m_x > m_z$
$n_x \ge 0$	$\xi_{c,x} = \frac{\alpha}{3m_x},\ 0 \le \alpha < m_x$ $\xi_{c,x} = \frac{1}{3}\frac{3\alpha - 2m_x}{2\alpha - m_x},\ m_x \le \alpha \le \frac{1}{2}$	$\xi_{c,x} = \frac{\alpha}{3m_x},\ 0 \le \alpha < m_z$ $\xi_{c,x} = \frac{1}{3m_x}\left[(2\alpha - m_z) - \frac{\alpha - m_z}{2\alpha - m_z}\alpha\right],\ m_z \le \alpha \le \frac{1}{2}$
$n_x < 0$	$\xi_{c,x} = \frac{3m_x - \alpha}{3m_x},\ 0 \le \alpha < m_x$ $\xi_{c,x} = \frac{1}{3}\frac{3\alpha - m_x}{2\alpha - m_x},\ m_x \le \alpha \le \frac{1}{2}$	$\xi_{c,x} = \frac{3m_x - \alpha}{3m_x},\ 0 \le \alpha < m_z$ $\xi_{c,x} = \frac{1}{3m_x}\left[(3m_x - 2\alpha + m_z) + \frac{\alpha - m_z}{2\alpha - m_z}\alpha\right],\ m_z \le \alpha \le \frac{1}{2}$

Table 2
Solution of the inverse problem for axisymmetric geometry for $\phi \le \phi_{max}$.

		$\alpha(\phi)$
$m_x \le m_z$ $n_x \ge 0$	$\phi < \phi_{tr}$ $\phi < \phi_{crit}$	$\left(\frac{M\phi_{crit}}{2}\right)^{1/3}\left[-1+2\cos\left(\frac{\varphi_0}{3}+\frac{\pi}{6}\right)\right]$
	$\phi < \phi_{tr}$ $\phi \ge \phi_{crit}$	$\left(\frac{M}{2}\right)^{1/3}\left[\phi_{crit}^{1/3}+\left\{2\phi-\phi_{crit}+2\sqrt{\phi(\phi-\phi_{crit})}\right\}^{1/3}+\left\{2\phi-\phi_{crit}-2\sqrt{\phi(\phi-\phi_{crit})}\right\}^{1/3}\right]$
	$\phi \ge \phi_{tr}$	$m_z\phi+\frac{m_x}{2}\frac{\eta_0+2/3}{\eta_0+1/2}$
$m_x \le m_z$ $n_x < 0$	$\phi < \phi_{tr}$	$-\left(\frac{M\phi_{crit}}{2}\right)^{1/3}\left[-1+2\sin\left(\frac{\varphi_0}{3}\right)\right]$
	$\phi \ge \phi_{tr}$	$m_z\phi+\frac{m_x}{2}\frac{\eta_0+1/3}{\eta_0+1/2}$
$m_x > m_z$ $n_x \ge 0$	$\phi < \phi_{tr}$ $\phi < \phi_{crit}$	$\left(\frac{M\phi_{crit}}{2}\right)^{1/3}\left[-1+2\cos\left(\frac{\varphi_0}{3}+\frac{\pi}{6}\right)\right]$
	$\phi < \phi_{tr}$ $\phi \ge \phi_{crit}$	$\left(\frac{M}{2}\right)^{1/3}\left[\left\{2\phi-\phi_{crit}+2\sqrt{\phi(\phi-\phi_{crit})}\right\}^{1/3}+\left\{2\phi-\phi_{crit}-2\sqrt{\phi(\phi-\phi_{crit})}\right\}^{1/3}-\phi_{crit}^{1/3}\right]$
	$\phi \ge \phi_{tr}$	$\frac{1}{2}\left[m_z-2m_x\eta_0+\sqrt{4m_x^2\eta_0^2+8m_x^2\left(\eta_0+\frac{1}{2}\right)\phi-\frac{1}{3}m_z^2}\right]$
$m_x > m_z$ $n_x < 0$	$\phi < \phi_{tr}$	$-\left(\frac{M\phi_{crit}}{2}\right)^{1/3}\left[-1+2\sin\left(\frac{\varphi_0}{3}\right)\right]$
	$\phi \ge \phi_{tr}$	$\frac{1}{2}\left[m_z+2m_x\left(\eta_0+1\right)-\sqrt{4m_x^2\left(\eta_0+1\right)^2-8m_x^2\left(\eta_0+\frac{1}{2}\right)\phi-\frac{1}{3}m_z^2}\right]$

With the help of the basic geometrical definitions, the following relations can be derived for the geometric centroid of the orange-coloured area in the standardised geometry described above:

$$\vec{\xi}_c = (\xi_{c,1}, \xi_{c,2}) = \frac{\alpha}{3}\left(\frac{1}{m_1}, \frac{1}{m_2}\right),\ 0 \le \alpha < m_1; \tag{17}$$

$$\vec{\xi}_c = (\xi_{c,1}, \xi_{c,2}) = \frac{1}{3}\left(\frac{3\alpha - 2m_1}{2\alpha - m_1}, \frac{1}{m_2}\left[(2\alpha - m_1) - \frac{\alpha - m_1}{2\alpha - m_1}\alpha\right]\right),\ m_1 \le \alpha \le \frac{1}{2}; \tag{18}$$

with $m_1 \le m_2$, and with $\xi_{c,i}$ associated with its respective m_i for $i \in \{1, 2\}$. As the correspondence between $\xi_{c,x}$ in the real geometry and either $\xi_{c,1}$ or $\xi_{c,2}$ in the standardised representation is orientation dependent, four cases must be distinguished, based on the conditions $|n_x| \lessgtr |n_z|$ and $n_x \gtrless 0$. Here n_x is the original x-component of the normal vector: $|n_x| \le |n_z|$ results in $\xi_{c,x} = \xi_{c,1}$ and $|n_x| > |n_z|$ results in $\xi_{c,x} = \xi_{c,2}$; $n_x < 0$ results in the transformation $\xi_{c,x} \to 1 - \xi_{c,x}$. As an example, in the configuration shown in Fig. 3, $|n_x| < |n_z|$ and $n_x > 0$, implying $m_1 = m_x$, $m_2 = m_z$, and $\xi_{c,x} = \xi_{c,1}$.

Table 1 lists the $\xi_{c,x}$ values for all given cases. With $\xi_{c,x}$ known in all cases in terms of the interface geometry, Eq. (16), together with Eqs. (8) and (9), represent the solution of the forward problem in axisymmetric geometry: i.e. $\phi = f_{fwd}(\alpha; \vec{n}, \eta_0)$, for $\alpha \le \frac{1}{2}$.

2.1.2. Inverse problem in axisymmetric geometry (calculate α from ϕ and $\vec{n}$)

The solution of the inverse problem, $\alpha = f_{inv}(\phi; \vec{n}, \eta_0)$, is dependent on the geometric orientation of the interface with respect to the coordinate system, and must be developed independently for the four cases listed in Table 1. Table 2 presents

Table 3
Auxiliary values required for the inverse problem for axisymmetric geometry.

	$m_x \le m_z$	$m_x > m_z$
$n_x \ge 0$	$\phi_{tr} = \dfrac{m_x}{2m_z}\dfrac{\eta_0+1/3}{\eta_0+1/2}$ $\phi_{crit} = \dfrac{2m_x}{3m_z}\dfrac{\eta_0^3}{\eta_0+1/2}$	$\phi_{tr} = \dfrac{m_z}{2m_x}\dfrac{\eta_0+m_z/3m_x}{\eta_0+1/2}$ $\phi_{crit} = \dfrac{2m_x}{3m_z}\dfrac{\eta_0^3}{\eta_0+1/2}$
$n_x < 0$	$\phi_{tr} = \dfrac{m_x}{2m_z}\dfrac{\eta_0+2/3}{\eta_0+1/2}$ $\phi_{crit} = \dfrac{2m_x}{3m_z}\dfrac{(\eta_0+1)^3}{\eta_0+1/2}$	$\phi_{tr} = \dfrac{m_z}{2m_x}\dfrac{\eta_0+1-m_z/3m_x}{\eta_0+1/2}$ $\phi_{crit} = \dfrac{2m_x}{3m_z}\dfrac{(\eta_0+1)^3}{\eta_0+1/2}$
All cases	$M = 3m_x^2 m_z\left(\eta_0+\dfrac{1}{2}\right)$ $\varphi_0 = \arcsin\left(1-\dfrac{2\phi}{\phi_{crit}}\right), \quad \phi_{max} = \dfrac{K(1/2,\vec{n},\eta_0)}{2}$ $m_x = \dfrac{\lvert n_x\rvert\Delta x}{\lvert n_x\rvert\Delta x+\lvert n_z\rvert\Delta z}, \quad m_z = \dfrac{\lvert n_z\rvert\Delta x}{\lvert n_x\rvert\Delta x+\lvert n_z\rvert\Delta z}$	

a summary of the different branches of the solution of the inverse problem for axisymmetric geometry for $\phi \le \phi_{max}$. The parameter ϕ_{max} is case-dependent, and can be calculated by solving the forward problem according to:

$$\phi_{max} = f_{fwd}(1/2;\vec{n},\eta_0) = K(1/2;\vec{n},\eta_0)\cdot c_{max} = \frac{K(1/2;\vec{n},\eta_0)}{2}. \tag{19}$$

This is in direct analogy with the Cartesian situation, for which c_{max} is given simply by $c_{max} = 1/2$. Table 3 lists the auxiliary values required for the individual branches. As illustration, the solution process is described for one of the branches in Appendix A.

2.1.3. Complementary problem in axisymmetric geometry

So far, only the cases $0 \le \alpha \le \alpha_{max} = \frac{1}{2}$ and $0 \le \phi \le \phi_{max} = f_{fwd}(1/2;\vec{n},\eta_0)$ have been considered. To extend the solution to the full intervals $\alpha \in [0,1]$ and $\phi \in [0,1]$, we can use the principle of complementarity, analogously to the two-dimensional Cartesian situation [22]. The equalities (analogous to Eqs. (14) and (15)):

$$\phi(1-\alpha,-\vec{n},\eta_0) = 1-\phi(\alpha,\vec{n},\eta_0), \text{ and} \tag{20}$$

$$\alpha(1-\phi,-\vec{n},\eta_0) = 1-\alpha(\phi,\vec{n},\eta_0), \tag{21}$$

hold, as the swapping of the two phases in a given cell is equivalent to the transformation:

$$\begin{aligned}
\phi &\to \tilde{\phi} = 1-\phi,\\
\alpha &\to \tilde{\alpha} = 1-\alpha,\\
c &\to \tilde{c} = 1-c,\\
\vec{n} &\to \tilde{\vec{n}} = -\vec{n},\\
\eta_0 &\to \tilde{\eta}_0 = \eta_0.
\end{aligned}$$

Equation (20) is then a representation of the statement that the sum of the volume fractions of both phases is equal to one. To prove Eq. (21), we use the fact that Eq. (15) corresponds to the same transition, and so:

$$\begin{aligned}
\alpha(\phi,\vec{n},\eta_0) &= \alpha[c(\phi;\vec{n},\eta_0),\vec{n},\eta_0] = \tilde{\alpha}[\tilde{c}(\tilde{\phi};\tilde{\vec{n}},\eta_0),\tilde{\vec{n}},\eta_0] = 1-\alpha[\tilde{c}(\tilde{\phi};\tilde{\vec{n}},\eta_0),\tilde{\vec{n}},\eta_0]\\
&= 1-\alpha(\tilde{\phi},\tilde{\vec{n}},\eta_0) = 1-\alpha(1-\phi,-\vec{n},\eta_0).
\end{aligned} \tag{22}$$

The relation $c = f(\phi;\vec{n},\eta_0)$ is not developed specifically in this paper, since it is not needed in the formulation, though it could be derived by analogy to the other derivations presented. For the proof to hold, however, it is sufficient to realise that such a relation exists and is unique. Using Eqs. (20) and (21), solution of the inverse problem can now be extended to the full intervals $\alpha \in [0,1]$ and $\phi \in [0,1]$.

2.1.4. Performance evaluation

The motivation for development of a direct solution of the inverse reconstruction problem is *computational efficiency*, since iterative algorithms can achieve the same precision as bracketing or analytical algorithms. In this section, the performance of our method is evaluated. First, a short summary on the iterative methods developed in the past is given, followed by quantitative arguments justifying the performance superiority of our direct analytical method.

For axisymmetric cylindrical geometry, convergence of the inverse problem using the original algorithm of Rider and Kothe is reported to require $\sim$5 iterations after the bracketing procedure in the general case, and up to 15 iterations in cases for which $\phi \approx 0$ [8]. The later semi-analytic methods of Dyadechko and Shashkov [25] and Diot et al. [29] reduced the average number of iterations to $\gtrsim 3$ [29]. Lopez et al. [31] demonstrated that the efficiency of their method for 2D problems is very similar to the one of Diot et al. [29], while being simpler and applicable to 3D problems as well.

Evidently, our direct solution of the inverse problem must be faster than that of the contemporary iterative algorithms to justify its implementation into our code. Thanks to the simplicity of the considered geometry, the *Newton's method* [44] represents an appealing reference approach. In the Newton's method, we define an objective function $F(\alpha)$ as (assuming fixed normal vector $\vec{n}$ and reduced coordinate η_0):

$$F(\alpha) = \phi(\alpha) - \phi_{ref}, \tag{23}$$

where ϕ_{ref} is the volume fraction, for which we want to solve the inverse reconstruction problem. We then construct and repeatedly solve an iterative map:

$$\alpha^{n+1} = \alpha^n - \frac{F(\alpha^n)}{F'(\alpha^n)} = \alpha^n - \frac{\phi(\alpha^n) - \phi_{ref}}{\phi'(\alpha^n)} \tag{24}$$

with α^0 being a suitable initial guess and $'$ indicating differentiation with respect to α. The root of the equation $F(\alpha) = 0$, i.e. α such that $\phi(\alpha) = \phi_{ref}$, represents a superstable fixed point of this iterative map [45]; as a result, the Newton's method (iterative solution of Eq. (24)) converges to the true solution of the inverse problem quadratically [44].

The main challenge in implementing the Newton's method lies in the efficient calculation of the derivative in the denominator of Eq. (24). In [40], Chen and Zhang used the mean-value theorem to express this derivative with the help of the area of the phasic interface, developing the so-called Predicted-Newton (PN) method. Its main advantage is the simplicity and speed of evaluation of the derivative, requiring only the identification of points where the interface intersects the bounding cell.

Originally, we planned to compare the speed of execution of the PN method with our direct approach. In 2D axisymmetric geometry, the area of the phasic interface can be obtained as the product of the length of the projected line segment and the radial coordinate of its geometric centroid, as per the First Theorem of Pappus [44]. However, after implementation, we have discovered that the efficiency of the PN method depends strongly on the orientation of the phasic interface with respect to the bounding cell: the less aligned the interface is with the edges of the cell, the more iterations are required to solve the inverse problem with the slowest convergence occurring for the situation $|n_x| \approx |n_z|$, for which multiple oscillations around the true value are observed. This can be explained by the strong variation of the length of the line segment with α for such configuration.

For this reason, we have decided to evaluate the derivative directly by taking advantage of the simplicity of Eq. (16) instead. By differentiating it, we obtain:

$$\phi'(\alpha, \vec{n}, \eta_0) = K'(\alpha, \vec{n}, \eta_0)c(\alpha, \vec{n}) + K(\alpha, \vec{n}, \eta_0)c'(\alpha, \vec{n}) = \frac{\xi'_{c,x}c}{\eta_0 + 1/2} + Kc'. \tag{25}$$

The derivative c' can be evaluated immediately by the differentiation of Eqs. (8) and (9). Similarly, $\xi'_{c,x}$ results from the formulae in Table 1 in a straightforward manner. To evaluate the initial guess α^0, we first wanted to use the method described in [40]; given a Hermite interpolation polynomial $H(\alpha)$:

$$H(\alpha) = -2\alpha^3 + 3\alpha^2 - \phi_{ref}, \tag{26}$$

it is trivial to verify that $H(0) = -\phi_{ref}$, $H'(0) = 0$, $H(1) = 1 - \phi_{ref}$, $H'(1) = 0$, which corresponds to the properties of the objective function $F(\alpha)$ at these points (with the exception of the unlikely situation $n_x = 0$ or $n_z = 0$). Then, the initial guess can be found as the solution of the cubic polynomial equation $H(\alpha) = 0$ using similar methods to those utilised in Appendix A as:

$$\alpha^0 = \frac{1}{2} - \sin\left(\frac{\arcsin\left(1 - 2\phi_{ref}\right)}{3}\right). \tag{27}$$

However, this expression includes a successive evaluation of an inverse trigonometric and a standard trigonometric function. We have ascertained by testing that an initial guess of comparable quality but faster evaluation time can be deduced from a piecewise-quadratic interpolation polynomial as:

$$\alpha^0 = \begin{cases} \sqrt{\frac{\phi_{ref}}{2}} & \text{if } \phi_{ref} < 1/2 \text{ and} \\ 1 - \sqrt{\frac{1-\phi_{ref}}{2}} & \text{otherwise.} \end{cases} \tag{28}$$

Note that this function, as well as its first derivative are continuous.

Although the convergence of the Newton's method for functions with discontinuous second derivative is not guaranteed [44], the interpolation procedure presented in [40] and designed to enforce the convergence has not been found to be necessary during the evaluation of the test set presented below due to the high quality of the initial guess.

In order to compare the CPU time required for our direct approach against the iterative option, we consider a single-cell reconstruction test, in which the volume fraction, ϕ, the normal vector to the interface, $\vec{n}$, and the reduced radial coordinate of the cell face, η_0, are all given. In this test, the corresponding α value is found using both the direct analytic solution of the inverse reconstruction problem and the Newton's iterative method. Our method for solving both the forward and the inverse problems used in the process has been implemented into a C++ program under the C++11 standard and compiled using the Intel® 2021.1.1 compiler.[1] Since the literature is not fully consistent on the topic of compiler optimisation (e.g. Lopez et al. [31]: no optimisation; Diot et al. [29], Chen and Zhang [35]: O2 optimisation), we employ compilation incorporating both the O0 (no optimisation) and O3 (maximum optimisation) flags. The results with the O2 flag lie between these two. Double precision accuracy is used, and time is measured using the `high_resolution_clock` class of the `chrono`[2] library. To derive parameter values for comparison, we consider a test matrix obtained from the Cartesian product of the following parametric spaces:

- 54 values of ϕ, computed according to the formulae:

$$\phi_{i,j,1} = V_i \times 10^{E_j} \text{ and} \tag{29}$$

$$\phi_{i,j,2} = 1 - \phi_{i,j,1}, \tag{30}$$

  where $V_i \in \{1, 2, 4\}$ and $E_j \in \{-9, -8, \ldots, -1\}$. By distributing the points in this manner, the execution speed can be evaluated even for fringe cases, such as $\phi = 10^{-8}$ or $\phi = 1 - 10^{-8}$.
- 9 values of η_0: $\eta_0 \in \{0.2, 0.4, 0.8, 2, 4, 8, 20, 40, 80\}$.
- 99 values of n_x, distributed uniformly between -1 and 1. The absolute value $|n_z|$ of the z-component of the normal vector is then equal to $\sqrt{1-n_x^2}$; sign of n_z does not affect the solution of either the forward or the inverse problem.

This procedure corresponds to 48 114 test cases in total, which we run on a workstation with a 2.40 GHz Intel® Xeon® E5-2640 processor. To accurately measure the CPU time required for execution, average values for each of these cases after 30 000 runs are used. The error tolerance for the iterative approach was chosen as 10^{-13}: although the mean of the direct evaluation of $\left|\phi_{ref} - f_{fwd}[f_{inv}(\phi_{ref})]\right|$ is 7×10^{-15} and median 1×10^{-16}, for some cases this expression can attain values of $\mathcal{O}(10^{-13})$. Thus, we consider 10^{-13} to be a reasonable tolerance criterion for the termination of the Newton's method.

Fig. 4 shows the evaluated execution times for both the iterative approach and the direct inversion without and with compiler optimisation, as well as their ratio and also the number of iterations required by the Newton's method after the initial guess; results have been averaged for each volume fraction value considered. The overall results are reported in Table 4. Based on these results, we can conclude that our direct analytic solution computationally outperforms the iterative approach, with a ∼1.4-1.7× speed-up. Use of compiler optimisation reduces the advantage of the direct inversion approach. Note that the decrease of the speed-up for values of ϕ close to 0 or 1 is partly a result of using an absolute tolerance as a stopping criterion, rather than relative one. When only ϕ values between 0.1 and 0.9 are considered, the speed-up becomes 1.81 ± 0.42 with O0 optimisation and 1.68 ± 0.44 with O3 optimisation.

2.2. Interface tracking method

The interface reconstruction relations developed above have been implemented into the in-house code PSI-BOIL[3] as a component of its interface tracking algorithm. Assuming two phases (liquid and gas), with $\phi = 1$ corresponding to the liquid phase and $\phi = 0$ to the gas phase, the governing equation for the liquid volume fraction is just Eq. (2), which is reproduced here for clarity:

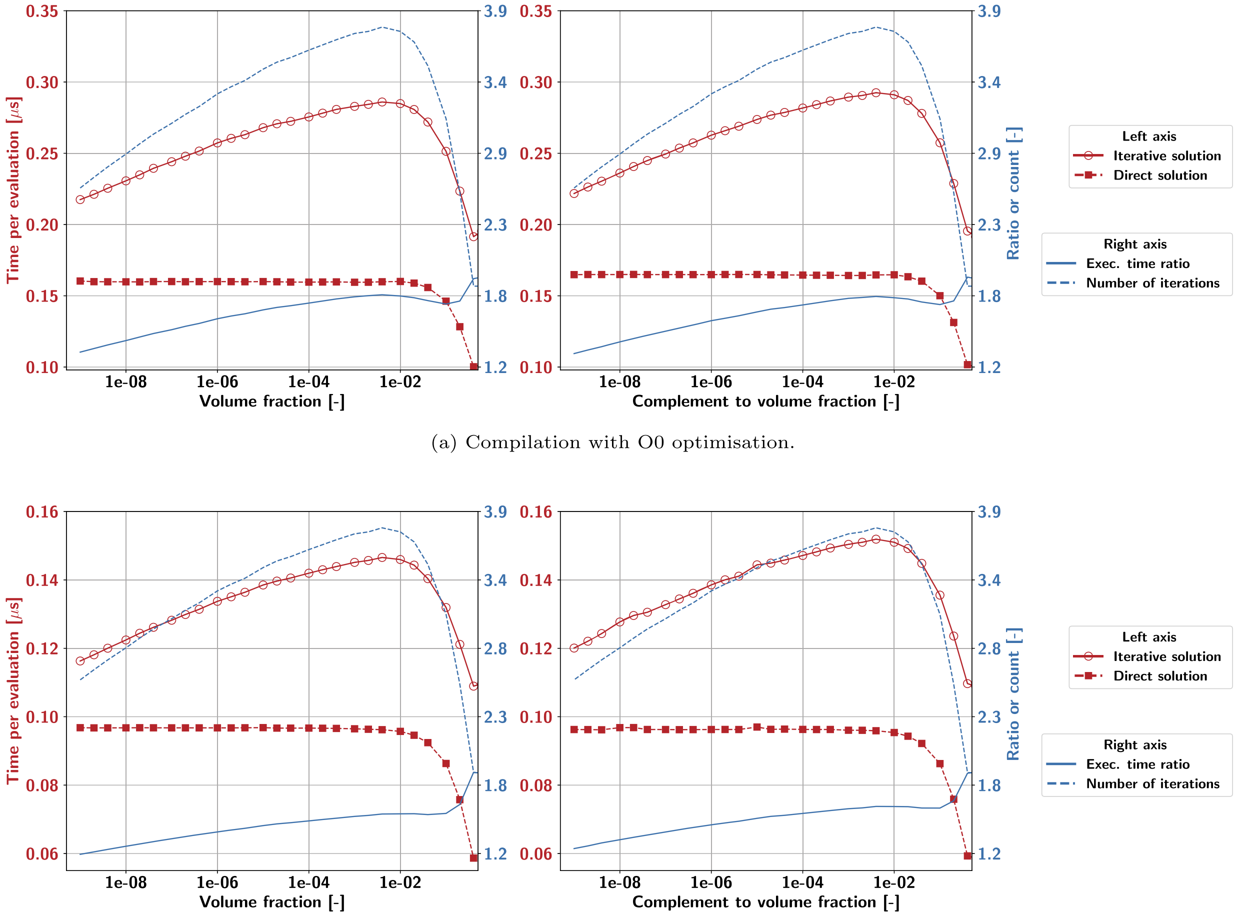
$$\frac{\partial \phi}{\partial t} + \nabla \cdot (\phi \vec{u}) = 0. \tag{31}$$

The equation is solved together with the single-field momentum conservation equations [46]:

[1] https://software.intel.com/content/www/us/en/develop/documentation/cpp-compiler-developer-guide-and-reference/.

[2] http://www.cplusplus.com/reference/chrono/high_resolution_clock/.

[3] https://github.com/PSI-NES-LSM-CFD/PSI-Boil.



(a) Compilation with O0 optimisation.

(b) Compilation with O3 optimisation.

Fig. 4. Execution times for the iterative approach and the direct analytic inversion, as well as their ratio and number of iterations required by the Newton's method. Left: $\phi \in (0, 1/2)$, right: $(1-\phi) \in (0, 1/2)$.

Table 4
Performance evaluation results for the iterative approach and for the direct analytic inversion. Lines designated by ⋆ correspond to the iterative approach with initial guess obtained by Eq. (27); the last line gives the time necessary to compute this guess.

	O0 optimisation			O3 optimisation		
	Mean	Std. dev.	Median	Mean	Std. dev.	Median
Number of iterations [-]	3.3	1.0	3	3.3	1.0	3
Iterative exec. time [μs]	0.258	0.059	0.250	0.135	0.026	0.133
Direct exec. time [μs]	0.158	0.022	0.164	0.095	0.014	0.099
Ratio it./dir. [-]	1.65	0.37	1.59	1.45	0.32	1.40
⋆Number of iterations [-]	3.33	0.96	3	3.33	0.96	3
⋆Iterative exec. time [μs]	0.286	0.055	0.272	0.163	0.024	0.160
⋆Initial guess eval. time [μs]	0.0797	0.0037	0.0768	0.0741	0.0018	0.0739

$$\frac{\partial(\rho\vec{u})}{\partial t} + \nabla\cdot(\rho\vec{u}\otimes\vec{u}) = -\nabla p + \nabla\cdot\left[\mu\left(\nabla\vec{u} + (\nabla\vec{u})^\top\right)\right] + \rho\vec{g} + \vec{f}_\sigma, \tag{32}$$

in which ρ [kg/m^3] and μ [Pa·s] are the mixture density and dynamic viscosity, respectively. The former can be calculated from its single-phase counterparts according to:

$$\rho = \phi\rho_l + (1-\phi)\rho_g, \tag{33}$$

and the latter according to either the weighted-harmonic mixing rule derived by Prosperetti [47] from a single-field representation of tangential stresses at the interface:

$$\frac{\rho}{\mu} = \phi\frac{\rho_l}{\mu_l} + (1-\phi)\frac{\rho_g}{\mu_g}, \tag{34}$$

or the classical harmonic mixing rule:

$$\frac{1}{\mu} = \phi\frac{1}{\mu_l} + (1-\phi)\frac{1}{\mu_g}. \tag{35}$$

Note that there is no unique definition of the mixture dynamic viscosity arising from the derivation of the single-field momentum equations [47]; as such, the final choice must be ultimately motivated by the quality of the numerical results [47].

Furthermore, in Eq. (32), p [Pa] refers to the pressure, $\vec{g}$ [m/s^2] the gravitational acceleration, and $\vec{f}_\sigma$ [N/m^3] the surface tension force density. The pressure is obtained via the projection method of Chorin [48], to satisfy the adiabatic incompressibility condition:

$$\nabla \cdot \vec{u} = 0. \tag{36}$$

Brackbill's Continuum Surface Force (CSF) model [49] is used to estimate the surface tension force density, $\vec{f}_\sigma$.

The solution algorithm for the above governing equations starts from the discretised form of Eq. (2) (or Eq. (31)):

$$\frac{\phi^{n+1} - \phi^n}{\Delta t} = -\nabla \cdot (\phi\vec{u}), \tag{37}$$

and can be summarised in the procedural steps listed below:

Step 1. Reconstruct interfacial geometry, calculating normal vectors and line constants.
Step 2. Advect the volume fraction field based on the reconstructed geometry.
Step 3. Calculate curvatures and surface tension force.
Step 4. Solve the momentum conservation equations to obtain a new velocity field (here, backward and forward Euler methods are used for the discretisation of the diffusion and advection terms, respectively).
Step 5. Advance the time step, and go back to Step 1.

Step 1, Step 2 and Step 3 are detailed below.

2.2.1. *Normal vector calculation and global iterative reconstruction*

In the interface reconstruction procedure, knowledge of the normal vector is a pre-requisite for the solution of both the forward and inverse problems. In 2D Cartesian geometry, algorithms for calculating $\vec{n}$ based on a c-value stencil are widely available; e.g. via the Young method [50], the Centred Columns method [50], LVIRA [50,51] or ELVIRA [51]. In axisymmetric geometry, the colour function c must be obtained from the volume fraction ϕ in order to accurately calculate the normal vector, since c, rather than ϕ, characterises the geometry of the interface. An iterative technique is therefore used to compute c and reconstruct the full geometry: i.e. to determine the complete ($\vec{n}$, α) pair, especially since the normal vector in general depends on the values of c in multiple cells. The iterative reconstruction algorithm has been implemented according to the following prescription, in which subscripts indicate the iteration count.

Step 1a. Set $c_0 = \phi$ everywhere.
Step 1b. Calculate the normal vector field $\vec{n}_0(c_0)$ under this assumption.
Step 1c. Solve the inverse problem $\alpha_0 = f_{inv}(\phi, \vec{n}_0)$ for the line-constant field.
Step 1d. Calculate c_1 (Eqs. (8) and (9)).
Step 1e. Calculate the updated normal vector field $\vec{n}_1(c_1)$.
Step 1f. Solve the forward problem $\phi_0^\star = f_{fwd}(\alpha_0, \vec{n}_1)$ to obtain an estimate of the volume fraction field, given the current $(\alpha, \vec{n})$ pair.
Step 1g. If the chosen *cost function* $Err(\phi_0^\star, \phi) < \varepsilon$, with ε being a prescribed tolerance, terminate the iteration. Candidate cost functions include the L^2-norm (N is the total number of grid cells):

$$L^2\text{-error} = \frac{1}{\sqrt{N}}\sqrt{\sum_{i=1}^{N}\left(\phi_{0,i}^\star - \phi_i\right)^2}, \tag{38}$$

or the L^∞-norm:

$$L^\infty\text{-error} = \max_{\forall\,\text{cells}} |\phi_0^\star - \phi|. \tag{39}$$

Step 1h. If not satisfied, increment the iteration count and go back to Step 1c.

The ELVIRA algorithm [51] has been implemented for the $\vec{n}$ calculations performed here, due to its overall accuracy, and its ability to reconstruct any linear interface exactly [51].

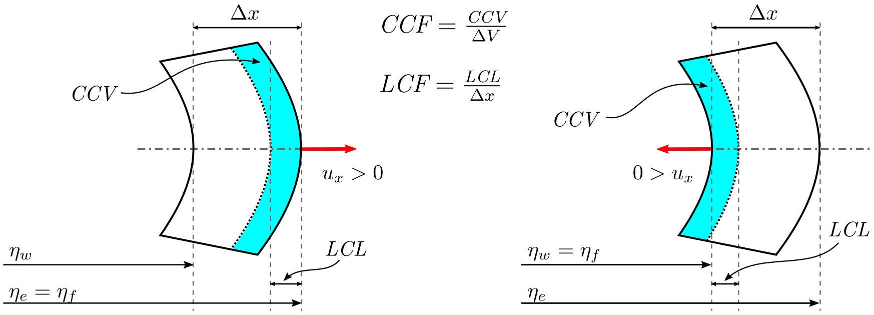


Fig. 5. Schematic representation of the nomenclature used to describe advection in the radial direction for cases with $u_x \gtrless 0$. In this representation, the z-axis is perpendicular to the page.

2.2.2. Volume fraction advection

The directional-split method [6] has been adopted here for the flux calculations in the solution of the VOF advection equation (Eq. (31)). Both the *sequential flux-splitting* approach with independent splitting and the *bounded conservative flux-splitting* approach described in [52] have been modified for application to axisymmetric geometry. In the former method, the flux in each direction is calculated independently, and large undershoots and overshoots of the volume fraction (i.e. values outside the physically meaningful interval [0,1]) can occur. The latter method was designed to resolve this problem [52]; however, it requires solving the inverse reconstruction problem after each directional sweep, and introduces a corrective term to account for the non-zero divergence of the individual velocity components; see [52] for a detailed discussion.

For both approaches, the liquid volume advected in the given direction (either x or z) must be calculated for each cell to obtain the flux value. To this end, the forward reconstruction problem must be solved within a segment of the computational cell delineated by the so-called *cut-cell fraction* (*CCF*), which is calculated in 3D Cartesian VOF formulations according to [52]:

$$CCF_{cart,s} = \frac{u_s \Delta t}{\Delta s}, \tag{40}$$

in which u_s is the advective velocity in the direction $s \in \{x, y, z\}$, Δt is the time step, and Δs is the cell extent in the given direction (i.e. Δx, Δy or Δz). This equation is also applicable for advection in the axial direction in axisymmetric geometries, but the *CCF* for the radial advection then requires modification. It can be derived from the continuity equation as:

$$CCF_{radial} = \frac{\eta_f^2 - (\eta_f - LCF_{radial})^2}{\eta_e^2 - \eta_w^2}, \tag{41}$$

where η_w and η_e are the reduced radial coordinates of the inward (west) and outward (east) faces of the cell, η_f is the reduced radial coordinate of the particular face (i.e. either η_w or η_e), and LCF_{radial} is the radial *line cut fraction* in the given cell, as calculated from:

$$LCF_{radial} = \eta_f - \sqrt{\eta_f^2 - 2\eta_f \frac{u_x \Delta t}{\Delta x}}. \tag{42}$$

The meaning of the individual symbols is illustrated schematically in Fig. 5.

2.3. Curvature estimation using height functions

A fundamental property of interface tracking in the context of bubble/droplet multiphase flows is the ability to estimate the cell-wise curvature κ of the interface, which is needed for estimating the surface tension force included in the momentum conservation equations. In axisymmetric geometry, κ can be expressed in terms of the principal radii:

$$\kappa = \frac{1}{R_{xz}} + \frac{1}{R_z} = \kappa_{xz} + \frac{\tilde{\kappa}_z}{x_r}, \tag{43}$$

where x_r is the radial position of the interface, and represents the radius of revolution around the z-axis. The first term in Eq. (43) is just the curvature calculated in a two-dimensional Cartesian projection of the geometry on the x-z plane, while the second arises from the revolution around the z-axis, and is of course identically zero in the Cartesian case. In the context of VOF methods generally, the current state-of-the-art for curvature estimation is by means of the height-function method [53], first comprehensively presented by Sussman [54] and Cummins et al. [55]; the review paper of Popinet [53] describes the height-function method in detail. Using basic differential geometry, both the Cartesian curvature κ_{xz} and the

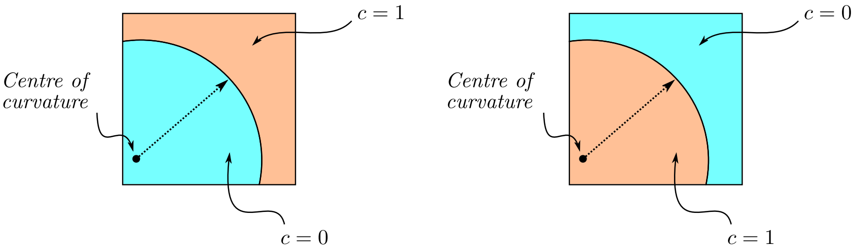


Fig. 6. Illustration of the curvature sign convention. Curvature is positive in the left configuration and negative in the right one.

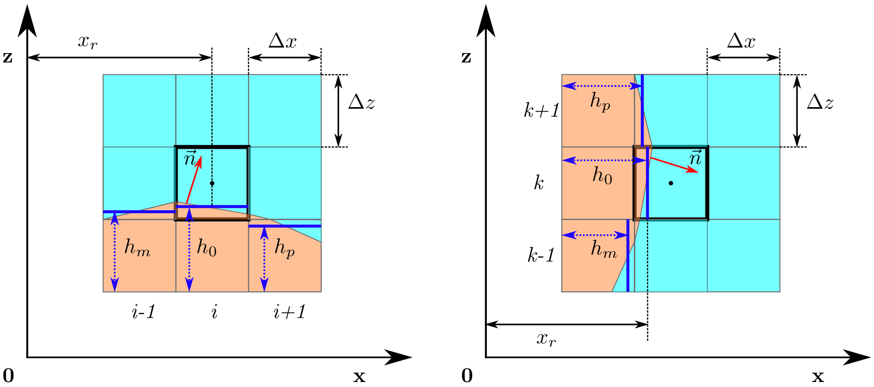


Fig. 7. Schematic representation of the height function method for the calculation of curvature in the highlighted cell. Only 3 × 3 stencils with uniform grid spacing are shown, for clarity. Left: $|n_z| > |n_x|$, implying the use of a stencil aligned with the z-axis. Right: $|n_x| > |n_z|$, implying the use of a stencil aligned with the x-axis.

non-dimensional factor $\tilde{\kappa}_z$ can be easily calculated (h_z and h_x are local representations of the interface as graphs of the functions $z = h_z(x)$ and $x = h_x(z)$, respectively). Viz:

$$\kappa_{xz} = \frac{\partial^2 h_z}{\partial x^2} \Big/ \left[1 + \left(\frac{\partial h_z}{\partial x}\right)^2\right]^{1.5} = \frac{\partial^2 h_x}{\partial z^2} \Big/ \left[1 + \left(\frac{\partial h_x}{\partial z}\right)^2\right]^{1.5}, \tag{44}$$

$$\tilde{\kappa}_z = \frac{\partial h_z}{\partial x} \Big/ \sqrt{1 + \left(\frac{\partial h_z}{\partial x}\right)^2} = -1 \Big/ \sqrt{1 + \left(\frac{\partial h_x}{\partial z}\right)^2}. \tag{45}$$

Under the chosen sign convention, which is standard for evaluating curvatures, the radius of curvature is positive if the centre of curvature is located "below" the surface, i.e. in the region for which $c < \frac{1}{2}$. This is illustrated in Fig. 6. Note that in general the centre of curvature may not lie in the same cell as the arc of the surface under consideration.

In the two-dimensional Cartesian implementation of the height function method, the functions h_x and h_z in Eqs. (44) and (45) are replaced by the area fraction field, integrated in a given direction on a local stencil [53]. In axisymmetric geometry, the area fraction is represented by the colour function c used in our calculations. A 3 × 7 stencil with the local topology adaptation of Lopez et al. [56] is used for the height integrations, in which the major stencil direction is determined by the dominant normal vector component: i.e. $|n_x| > |n_z|$ results in alignment of the stencil with the x-direction, and $|n_x| < |n_z|$ results in alignment of the stencil with the z-direction. For the highly unlikely situation in which $|n_x| = |n_z|$, the major stencil direction is undefined; so in this case we choose x, just for simplicity.

Fig. 7 illustrates the curvature calculation in schematic form. For clarity, 3 × 3 stencils with uniform grid spacing are shown, but a 3 × 7 stencil is actually implemented in PSI-BOIL. In the example on the left, $|n_z| > |n_x|$, implying the use of a stencil aligned with the z-axis. In the example on the right, $|n_x| > |n_z|$, implying the use of a stencil aligned with the x-axis. Note that to maintain consistent second-order accuracy of the curvature estimation using height functions in axisymmetric geometry, the radius of revolution, x_r in Eq. (43), must take into account the position of the interface with respect to the given stencil alignment, as indicated in Fig. 7.

3. Verification and validation

For verification purposes, artificial advection problems, and problems involving the generation of parasitic currents, have been selected for simulation. For the validation, the well-known cylindrical dam-break problem, and bubble rise due to buoyancy, have been chosen. Note that, barring influence of the outlet boundary conditions, exact conservation of total volumetric content of both phases has been confirmed in all simulations. The weighted-harmonic viscosity mixing rule (Eq. (34)) has been used in calculations in which the Navier-Stokes equations were solved, if not stated otherwise.

Table 5
Simulation parameters of reversible advection verification cases.

	Radial extent [m]	Axial extent [m]	Initial circle radius [m]	Initial circle location (x, z) [m]	Total time [s]
Stretching field	$(0, L)$	$(-2.5L, 2.5L)$	$0.1L$	$(0.7L, 0)$	3.20 L/A
Contracting field	$(0, L)$	$(-0.25L, 0.25L)$	$0.1L$	$(0.1L + \Delta x, 0)$	2.64 L/A
Circular vortex	$(0, L)$	$(0, L)$	$0.2L$	$(0.4L, 0.4L)$	15 L/A

3.1. Reversible advection problems

To verify the correct implementation of our advection algorithm in the code PSI-BOIL, the volume fraction field has been artificially transported by means of prescribed divergence-free velocity fields in different geometries for a pre-set number of time steps, and then run in reverse for the same duration. The differences between the initial and final configurations are then quantified, together with any under/overshoots of the volume fraction ϕ with respect to its physical range [0,1], i.e. $-\phi_{min}$ and $(\phi_{max} - 1)$. Note that these are very challenging tests of the model implementation. The following velocity fields have been considered:

1. Stretching field:

$$u(x, z) = -\frac{Ax}{L}; \tag{46}$$

$$w(x, z) = 2\frac{Az}{L}. \tag{47}$$

2. Contracting field:

$$u(x, z) = \frac{Ax}{L}; \tag{48}$$

$$w(x, z) = -2\frac{Az}{L}. \tag{49}$$

3. Circular vortex:

$$u(x, z) = -A\cos\left(\frac{\pi z}{L}\right)\sin\left(\frac{\pi x}{L}\right); \tag{50}$$

$$w(x, z) = \frac{AL}{\pi x}\sin\left(\frac{\pi z}{L}\right)\left[\sin\left(\frac{\pi x}{L}\right) + \frac{\pi x}{L}\cos\left(\frac{\pi x}{L}\right)\right]. \tag{51}$$

In all cases, advection is applied to analytically initialised circles (corresponding to tori in axisymmetric representation), and the domain dimensions taken as suitable multiples of L to accommodate the full advection process. Table 5 lists details of the chosen simulation parameters. Uniform discretisation of the grid has been adopted in all cases. Grid resolution is measured in terms of the number of cells per initial circle radius, with assumed values ranging from 20 to 240. The time step is taken as constant for each simulation, so that the *CFL* condition:

$$\frac{v_{max}\Delta t}{\Delta x} = CFL_{lim} \tag{52}$$

could be satisfied, with v_{max} the maximum of $\sqrt{u^2 + w^2}$ in the computational domain. Multiple values of the *CFL* limit in the range $[0.02, 0.4]$ have been adopted to test the effect of time step on the simulation results. Both “droplet-like” and inverted “bubble-like” tori have been tested: the effects of colour inversion, i.e. whether liquid or gas is considered as the primary fluid, were found to be negligible. For a qualitative illustration of the advection process, Figs. 8, 9, and 10 give snapshots of the simulations for selected grid resolutions and CFL_{lim} values. It can be seen that in all cases the initial and final states bear much similarity, despite the challenging transient conditions imposed.

3.1.1. Comparison of flux calculation methods

We have performed calculations using both the sequential and bounded conservative flux-splitting methods described in Section 2.2. For a quantitative comparison of their convergence properties, Figs. 11, 12 and 13 show the simulation errors as functions of CFL_{lim} and grid resolution. The L^∞-errors of undershoot and overshoot in Fig. 12 are defined as:

$$L^\infty\text{-error (undershoot)} = \max_{\forall\,\text{cells}}(-\phi),\ \text{and} \tag{53}$$

$$L^\infty\text{-error (overshoot)} = \max_{\forall\,\text{cells}}(\phi - 1), \tag{54}$$

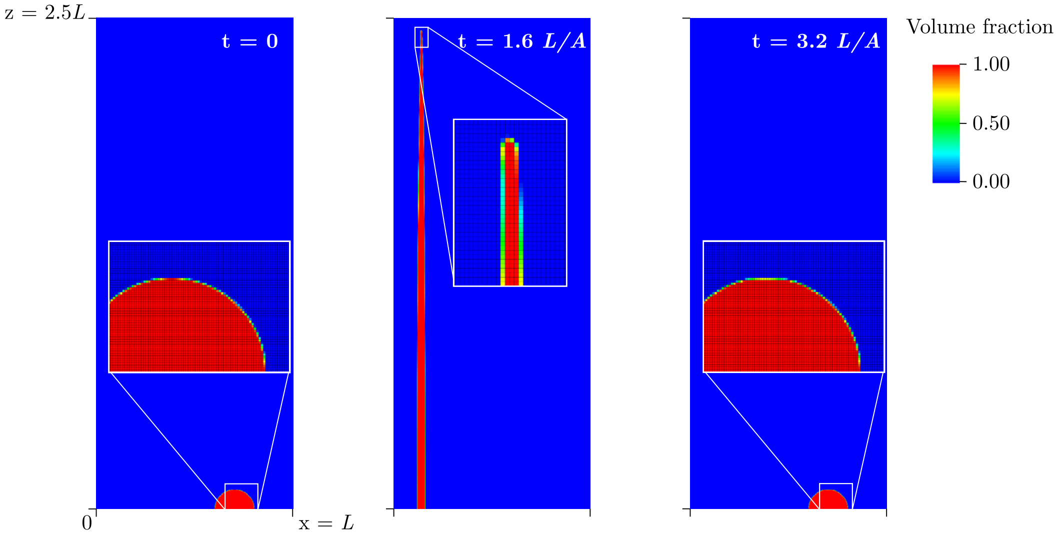


Fig. 8. From left to right: initial, midpoint and final states of the domain in the "droplet-like" stretching field reversible advection test case with bounded conservative flux splitting. There are 40 cells per initial circle radius; $CFL_{lim} = 0.2$. Only one half of the domain is shown.

while the final L^1-error in Figs. 12 and 13 is calculated according to:

$$L^1\text{-error} = \frac{1}{V}\sum_{i=1}^{N}\left|\phi_i^{final} - \phi_i^{initial}\right| V_i, \tag{55}$$

where N is the total number of grid cells, V the total domain volume, and V_i the volume of cell i.

It can be noticed in Fig. 11 that for the sequential flux-splitting method the undershoots $-\phi_{min}$ and overshoots $\phi_{max} - 1$ are quadratically dependent on CFL_{lim} as a result of the over-estimation of the flux caused by the independent splitting. However, there is no significant difference between these two error estimates. For the bounded conservative method, no such trend can be discerned. This is not surprising as this method was specifically designed to prevent any overshoot/undershoot error resulting from the independent splitting for $CFL < 0.5$ [52]. Thus, these errors remain negligible, independently of the CFL number.

The final L^1-error varies strongly with CFL for the sequential flux-splitting method. For example, as can be discerned from Fig. 12, the CFL-dependence of the error for the stretching and contracting field cases could be described as essentially linear, with a non-zero offset. This offset most probably derives from errors induced by interface reconstruction, and as such would be independent of the CFL number. While the final shape of the advected object is continuous and close to circular in the stretching field and contracting field cases, strong fragmentation is observed in the circular vortex cases, with the degree of severity increasing with increasing CFL number. This feature tends to dominate the final error and, as a result, the L^1-error is seen to be unacceptably high for $CFL_{lim} \gtrsim 0.2$. This is a direct result of the over-estimation of the computed fluxes.

For the bounded conservative method, the final error depends only weakly on CFL_{lim}. It can be seen that it is lower than the error for the sequential flux-splitting method for all CFL values, with the two apparently converging for $CFL_{lim} \to 0$, which is to be expected. The non-zero error offset as $CFL_{lim} \to 0$ corresponds to the accumulated reconstruction error for the given grid resolution, as the PLIC method with a finite grid spacing can never represent curved interfaces exactly; as a result, the L^1-error tends to a non-zero value as $CFL_{lim} \to 0$. Note that due to the very good performance of the bounded conservative flux-splitting algorithm, the total error actually slightly decreases with increasing CFL number. This could be attributed to the reduced number of time steps, thereby reducing the number of interfacial reconstructions needed to be performed. Indeed, for a method in which the L^1-error is dominated by the reconstruction error, this would lead to a reduction of the overall error.

For the dependence of the L^1-error on grid resolution, the following observation can be made: assuming that cells in the vicinity of the interface account for the major contribution to the error, which is reasonable, and that the contribution ε of an individual cell is approximately constant, the L^1-error should be proportional to G^{-1}, where G is the grid level quantifying the number of cells per dimension, in our case the number of cells per initial circle radius. This deduction is a consequence of the number of cells near the interface N_γ being $\mathcal{O}(G)$, and the total number of cells N being $\mathcal{O}(G^2)$. Hence:

$$L^1\text{-error} \approx \frac{N_\gamma}{N}\varepsilon \propto \frac{G}{G^2}\varepsilon = G^{-1}\varepsilon. \tag{56}$$

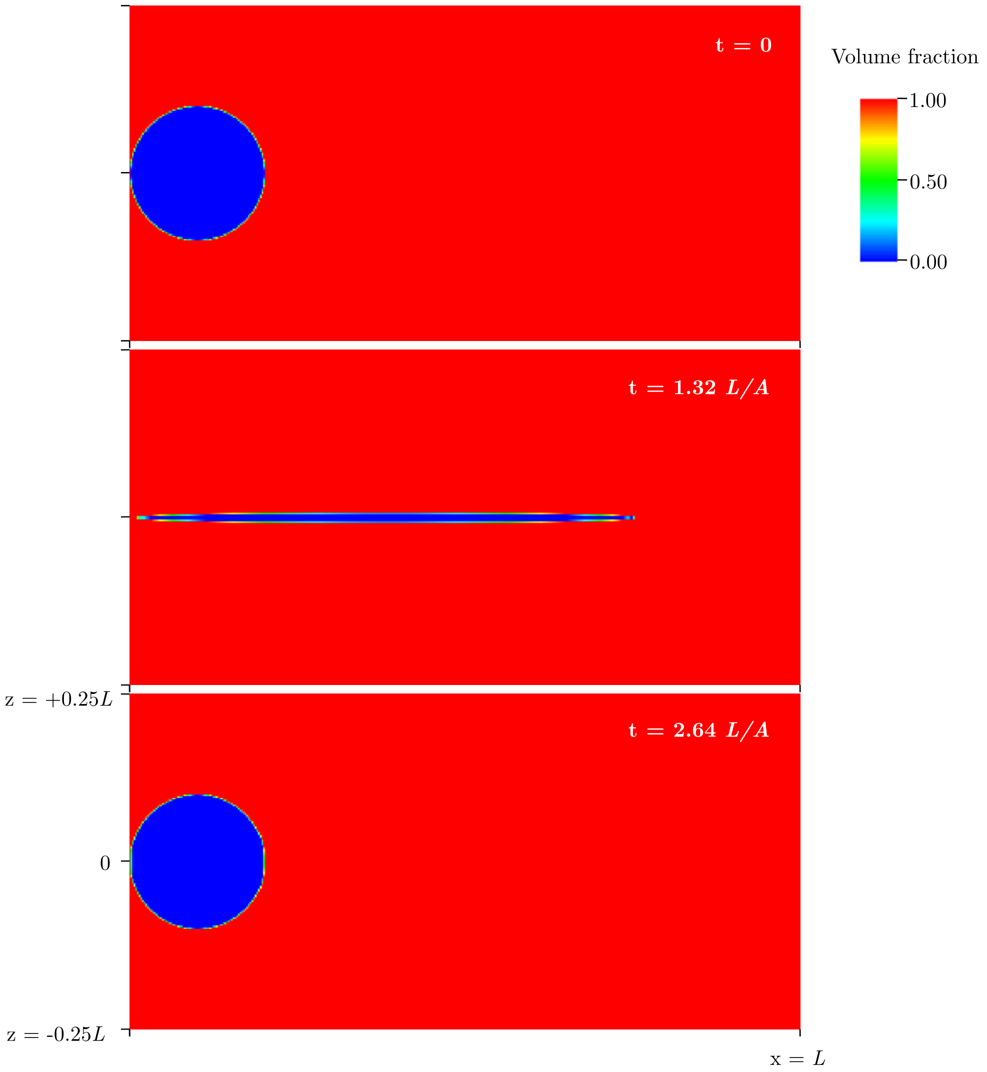


Fig. 9. From top to bottom: initial, midpoint and final states of the domain in the contracting field "bubble-like" reversible advection test case with bounded conservative flux splitting. There are 40 cells per initial circle radius; $CFL_{lim} = 0.2$.

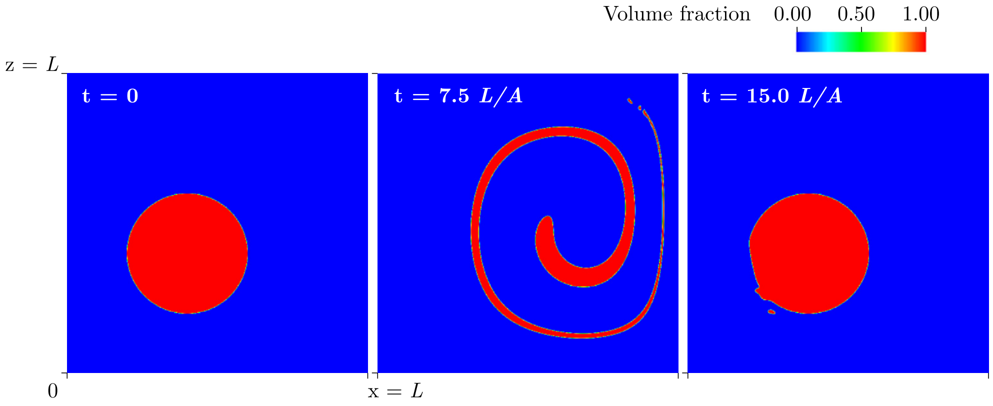


Fig. 10. From left to right: initial, midpoint and final states of the domain in the circular vortex field "droplet-like" reversible advection test case with bounded conservative flux splitting. There are 80 cells per initial circle radius; $CFL_{lim} = 0.2$.

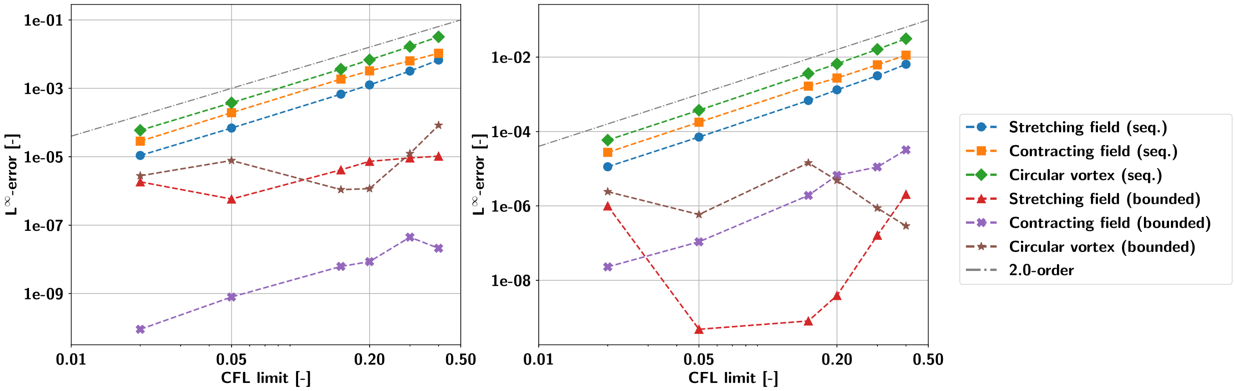


Fig. 11. Maximum undershoot (left) and overshoot (right) of the volume fraction during the simulations as functions of the imposed CFL_{lim} for all reversible advection cases: grid resolution 20 cells per initial circle radius for the stretching and contracting field cases, and 80 cells per initial circle radius for the circular vortex.

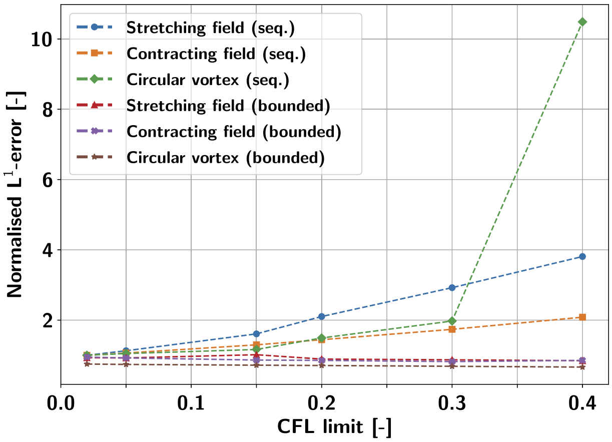


Fig. 12. Final L^1-errors of the volume fraction field as functions of the imposed CFL_{lim} for all reversible advection cases: grid resolution is 20 cells per initial circle radius for the stretching and contracting field cases, and 80 cells per initial circle radius for the circular vortex. Values are normalised in terms of the L^1-error of the respective case, for $CFL_{lim} = 0.02$ using the sequential flux-splitting method.

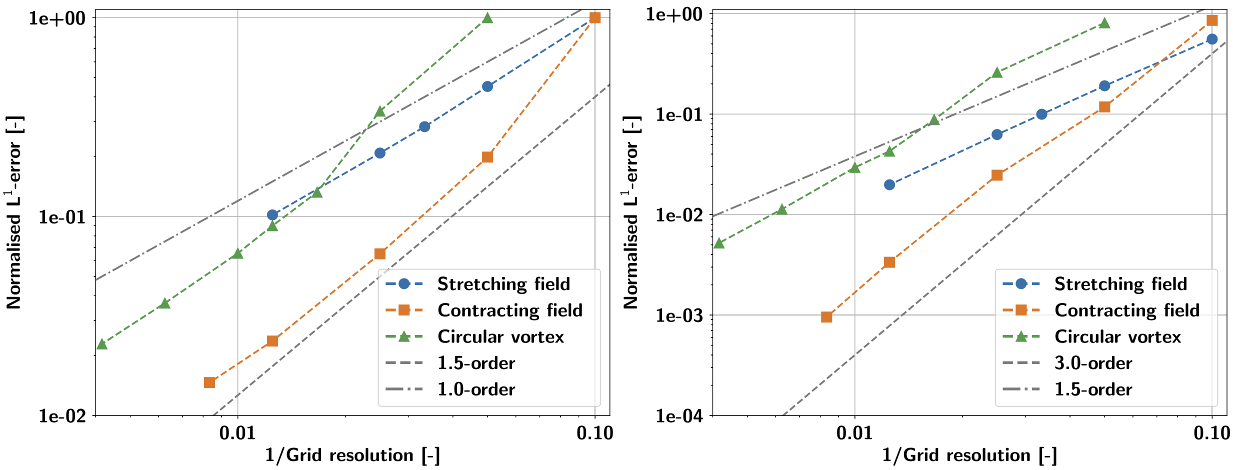


Fig. 13. Final L^1-errors of the volume fraction field as functions of the reciprocal of the grid resolution (measured in cells per initial circle radius) for all reversible advection cases for $CFL_{lim} = 0.2$; left: sequential flux-splitting method, right: bounded conservative flux-splitting method. Values are normalised in terms of the L^1-error of the respective case for the coarsest grid resolution considered using the sequential flux-splitting method.

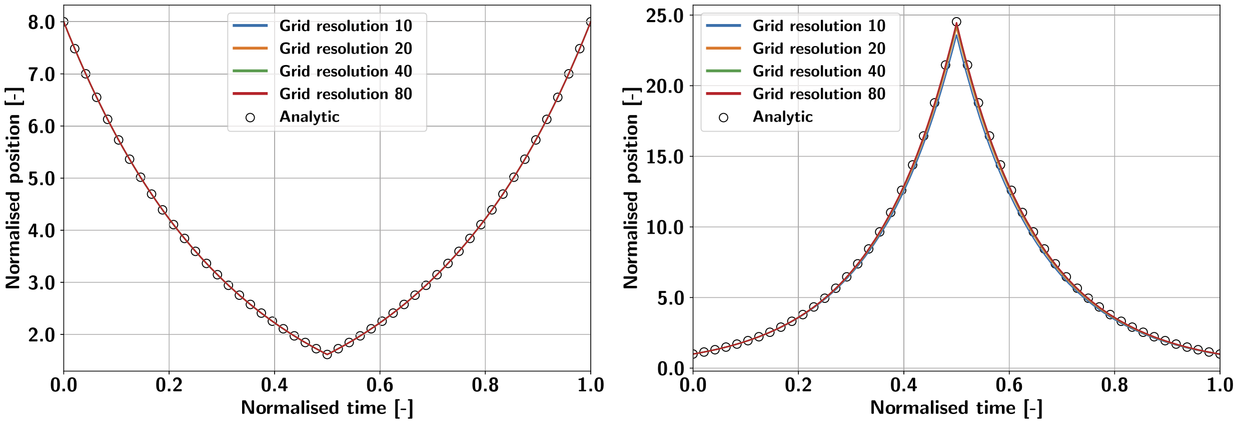


Fig. 14. Position of the left (x_{max}, shown in the left figure) and top (z_{max}, shown in the right figure) edges of the advected circle as functions of time for the stretching-field/reversible advection test case; results of four different grid resolutions are shown as well as the analytic solutions (Eqs. (59) and (60)). Position is normalised by the initial circle radius $R = 0.1L$, and time is normalised in terms of the total time (see Table 5).

Fig. 13 shows that, for the sequential flux-splitting method, this dependence is exactly that observed for the stretching field case, as well as for the circular vortex case (after some initial non-monotonicity). In this case, the main source of error most probably results from surface fragmentation. The initial non-monotonic behaviour of the error corresponds to a qualitative change resulting from grid refinement, since for the two coarsest grid spacings very strong fragmentation occurs at the midpoint of the simulation in the tail of the spiral, which significantly increases the final error. Similarly, for the contracting field case, with the sequential flux-splitting method, the midpoint geometry is under-resolved for coarse grid spacings, whereas for fine grid spacings 1.5-order accuracy is achieved. Switching to the bounded conservative flux-splitting method reduces the error significantly; furthermore, the order of the asymptotic accuracy increases to ∼1.6 for the stretching field case, ∼1.9 for the circular vortex case, and ∼3 for the contracting field case. Evidently, using this method, the error in interfacial cells reduces with increasing grid resolution.

As the bounded conservative flux-splitting method outperforms the sequential one in all cases, this option will be adopted for the rest of the simulations described in this paper.

3.1.2. Evolution of interfacial position

For the stretching and contracting field cases, the u and w components of velocity are only dependent on the x and z coordinates, respectively. Thus, for an infinitesimal fluid parcel located originally at $[x_0, z_0]$, its position as a function of time can be calculated analytically by solving two first-order linear differential equations:

$$\frac{dx}{dt} = u(x,z) = -S\frac{Ax}{L}, \text{ and} \tag{57}$$

$$\frac{dz}{dt} = w(x,z) = 2S\frac{Az}{L}, \tag{58}$$

in which $S = 1$ for the stretching field case and $S = -1$ for the contracting field case. For both, the solution is:

$$x(t) = x_0 \exp\left(-S\frac{A}{L}t\right), \text{ and} \tag{59}$$

$$z(t) = z_0 \exp\left(2S\frac{A}{L}t\right). \tag{60}$$

Figs. 14 and 15 compare the position of the left (x_{max}) and top (z_{max}) edges of the advected circle as functions of time, obtained from the simulations at various levels of grid refinement, with the analytical solution. It can be observed that near-perfect agreement has been achieved overall, the only exception being the contracting field advection with 10 cells per initial circle radius: here, due to under-resolution of the midpoint condition at $t = 1.32L/A$ (only 2 cells per axial extent of the stretched circle), the algorithm fails to capture advection in the reverse direction. Note that no analytical solution is available for the circular vortex case, but almost perfect recovery of the initial conditions, as in the stretching and contracting field cases, gives strong indications that the algorithm has been coded correctly.

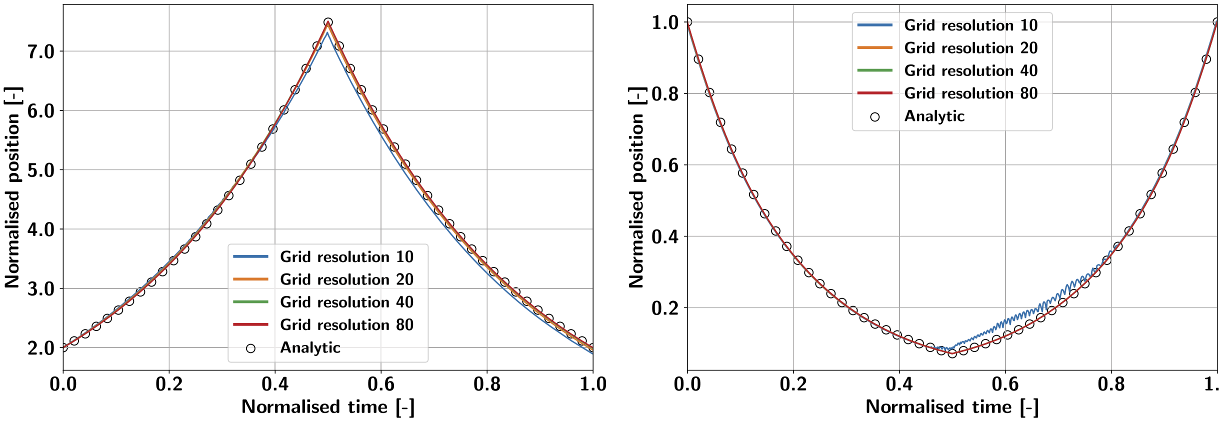


Fig. 15. Position of the left (x_{max}, shown in the left figure) and top (z_{max}, shown in the right figure) edges of the advected circle as functions of time for the contracting field reversible advection test case; results of four different grid resolutions are shown as well as the analytic solutions (Eqs. (59) and (60)). Position is normalised by the initial circle radius $R = 0.1L$, and time is normalised by the total time (see Table 5).

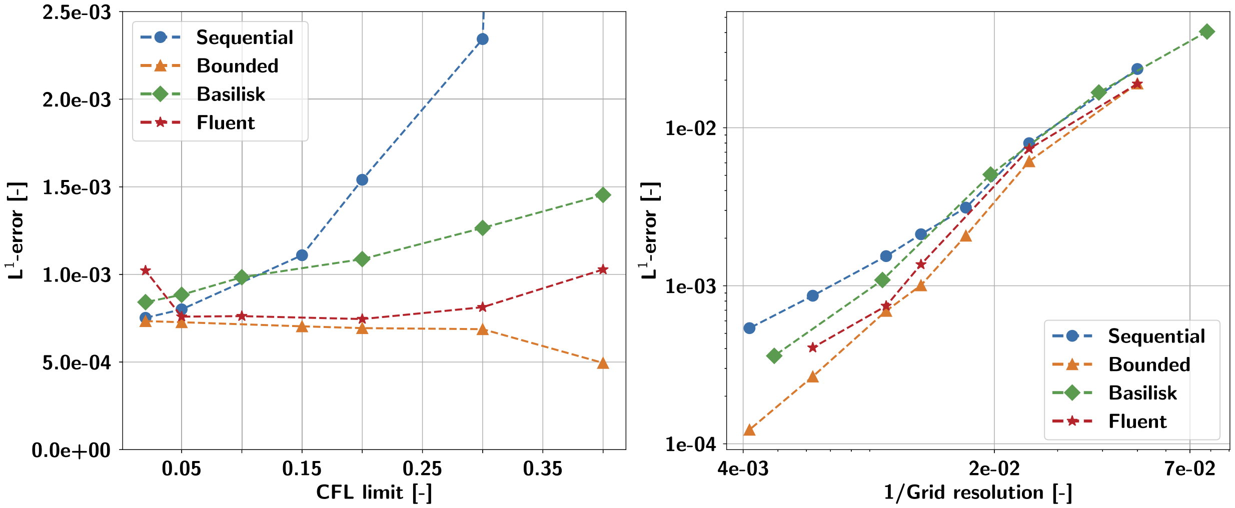


Fig. 16. Final L^1-errors of the volume fraction field as functions of imposed CFL_{lim} (left) and the reciprocal of the grid resolution measured in cells per initial circle radius (right). Results for the circular vortex case are shown using our approach with the sequential flux-splitting method, our approach with the bounded conservative flux-splitting method, Basilisk and Ansys® Fluent.

3.1.3. Comparison with contemporary VOF methods

In order to compare our method with existing implementations of the (axisymmetric) PLIC-VOF method, we have simulated the circular vortex case using the commercial CFD software Ansys® Fluent 2020 R1[4] [57] and the open-source CFD code Basilisk.[5] Note that both Fluent and Basilisk incorporate the PLIC geometric reconstruction algorithm: see the Fluent Theory Guide [57] and the Basilisk source code at the referenced website. Basilisk uses the bounded conservative flux-splitting advection method of Weymouth and Yue [52]; but we have not been able to find out the precise details of the Fluent advection algorithm, either in the documentation supplied with the code, nor in the open literature.

Fig. 16 shows a comparison of the final L^1-errors using our implementation with the sequential flux-splitting method, our implementation with the bounded conservative flux-splitting method, Basilisk and Ansys® Fluent. For the simulations with variable CFL_{lim}, a grid resolution of 100 cells per initial circle radius is used for our method, and for Fluent, and 102.4 cells for Basilisk; this minor discrepancy resulting from the use of factors of 2 for domain discretisation in Basilisk, has a negligible effect on the comparisons. For the simulations with variable grid resolution, $CFL_{lim} = 0.2$ in all cases. It can be seen that our approach with the bounded conservative flux-splitting method outperforms both Fluent and Basilisk under all conditions, meaning L^1-errors are always smaller in our case. Furthermore, Fig. 17 compares the overshoot and undershoot errors of our approach against those of Basilisk, as functions of CFL_{lim}. Our approach, with the bounded conservative

[4] https://www.ansys.com/resource-library/brochure/ansys-fluent.

[5] http://basilisk.fr.

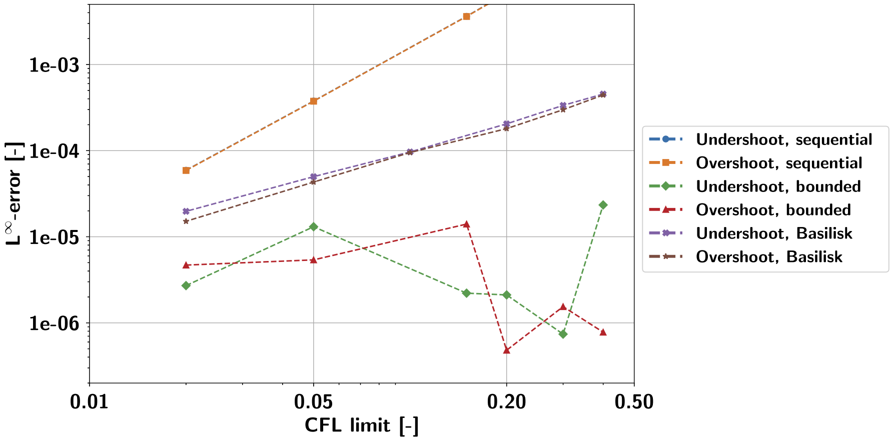


Fig. 17. Maximum undershoot and overshoot of the volume fraction during the circular vortex simulations as functions of the imposed CFL_{lim}. Results obtained using Basilisk and our approach with the sequential and the bounded conservative flux-splitting methods are shown. The results for our approach with the sequential flux-splitting method overlap.

Table 6
Simulation parameters of the Rudman [2] advection verification case. Initial location refers to the coordinates of the centre of area of the advected circle.

Radial extent [m]	Axial extent [m]	Initial location (x, z) [m]	Total time [s]
$(0, 2.8)$	$(0, 4)$	$(1.36, 1.88)$	4

flux-splitting method, can again be seen to achieve superior performance in terms of L^∞-error. Note that with Fluent, no overshoots and undershoots of the volume fraction occur, possibly due to clipping and filling of cells; as a result, the total volume of the phases is not exactly conserved during the simulation.

3.2. Rudman advection problem

In the test cases described in the previous section, the imposition of exact flow reversal can obscure the effects of the accumulating reconstruction errors. To evaluate the stability of the advection algorithm with respect to the reconstruction method, we have considered one of the forced advection test cases of Rudman [2], also used, for example, by Harvie and Fletcher [58]. In this test case, an analytically-initialised circle is advected using a constant, unidirectional velocity field and, after a given number of time steps, the volumetric fraction distribution is compared to the analytical solution. Note that for unidirectional flow, the sequential and bounded conservative flux-splitting methods coincide.

In axisymmetric geometry, only advection in the axial z-direction can be considered in order to avoid deformation of the advected objects due to radial stretching. In analogy to the original 2D Cartesian setup of Rudman [2], we consider an advection velocity of $w = 1$ m/s in the positive z-direction for a hollow circle with outer diameter of 1.6 m and inner diameter of 1.2 m. This shape represents a body of rotation in axisymmetric geometry with circular cross-section. Table 6 gives details of the simulation parameters adopted here. The domain is taken as periodic in the z-direction and the total time chosen so that the final position of the advected shape coincides with its original location. Uniform grid discretisation is used in all cases. Grid resolution is measured in terms of the number of cells per domain height, $L_z = 4$ m, ranging from 100 to 800. The time step is taken as constant in each simulation, so that the CFL condition could be satisfied, the velocity being constant and equal to the prescribed value of $w = 1$ m/s. Multiple values of the CFL limit in the range $[0.005, 1]$ are adopted to test the effect of the size of the time step on the simulation results.

Fig. 18 gives snapshots of the simulations for selected grid resolutions and CFL_{lim} values. It can be seen that the initial and final states are almost identical. Table 7 lists the final L^1-error as a function of grid resolution; the order of accuracy is ~1.8. Regarding the dependence of the final L^1-error on the CFL_{lim} number, the total error grows with decreasing CFL_{lim}, due to the increasing number of simulation steps which, in turn, correspond to an increased number of interface reconstructions. Indeed, in the limit of $CFL_{lim} = 1$, the total error is caused only by the repeated reconstructions. However, the L^1-error divided by the number of time steps, a measure representative of error per time step, scales almost linearly with the CFL_{lim} in the asymptotic region (i.e. $CFL_{lim} \lesssim 0.1$); see Fig. 19. The undershoots and overshoots of the solution are negligible for all cases considered: both are results of independent splitting during flux calculations, and these errors do not occur for unidirectional advection.

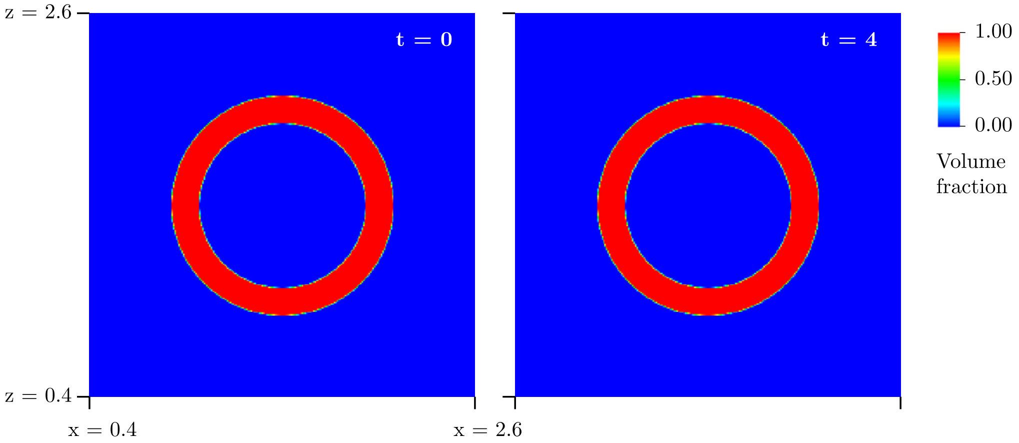


Fig. 18. Initial (left) and final (right) states of the domain in the Rudman circle advection test case. There are 400 cells per domain height; $CFL_{lim} = 0.2$. Only that part of the domain containing the circle is shown. Units in m and s.

Table 7
Final L^1-errors of the volume fraction field for various values of the grid resolution (measured in terms of the number of cells per domain height) for the Rudman circle advection case, with $CFL_{lim} = 0.2$. Values are normalised by the L^1-error for the coarsest grid resolution considered.

Grid resolution [-]	Normalised L^1-error [-]
1	1.00
2	0.282
4	0.0889
8	0.0270

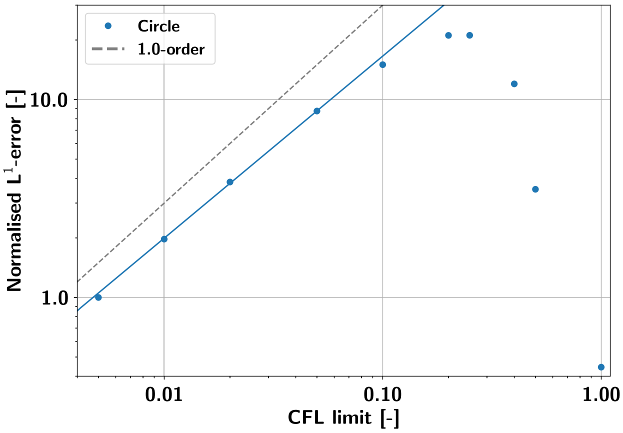


Fig. 19. L^1-errors per time step of the volume fraction field as a function of imposed CFL_{lim} for the Rudman advection test case for circle with grid resolution 200 cells per domain height. Values are normalised in terms of the L^1-error per time step for $CFL_{lim} = 0.005$.

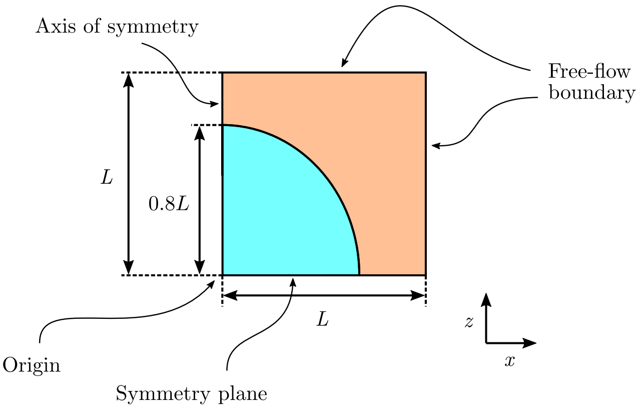


Fig. 20. Schematic representation of the domain used for the static parasitic current problem.

Table 8
Characteristics of surface tension verification cases.

	Laplace number *La* [-]	Density ratio [-]	Viscosity ratio [-]	$C_{\Delta t}$ (Eq. (65)) [-]	Note
'Droplet-like"	$2 \cdot 10^5$	1000	100	0.126	Heavier fluid in the bulk
"Bubble-like"	$2 \cdot 10^6$	1000	100	0.126	Lighter fluid in the bulk
Artificial case	$1 \cdot 10^3$	1	1	0.282	

3.3. Parasitic-current problems

3.3.1. Parasitic-current problem in a stagnant configuration (static)

The parasitic current problem is a standard benchmark for the implementation of the surface tension force in a code [53]. As described in detail by Popinet [59], the test consists of simulating a static sphere inside a stagnant bulk fluid. To represent this problem in axisymmetric geometry, we analytically initialise a circle of radius R in a square domain of side length L, and then calculate the corresponding volume fraction field using forward reconstruction. We choose $L = 2$ mm as a typical droplet/bubble size and $R = 0.8L$, following the choices made in [59]. Making use of the symmetry boundary condition at $z = 0$, only a half of the sphere needs to be simulated. Free-flow boundary conditions are applied at $z = L$ and $x = L$. Fig. 20 is a schematic representation of the domain used.

The simulation is run until several multiples of a "suitable" time scale (see below) have elapsed to allow the system to relax to its equilibrium state. If the surface tension force has been implemented correctly, the amplitude of the velocity field should decrease below the prescribed residual tolerance of the iterative momentum equations solver. A "suitable" time scale T in this case would be:

$$T = \max(T_\sigma, T_\mu), \tag{61}$$

in which T_σ is the capillary-wave time scale, calculated according to [59]:

$$T_\sigma = \sqrt{\frac{\rho d^3}{\sigma}}, \tag{62}$$

where σ is the surface tension coefficient between the two fluids, ρ the density of the fluid inside the sphere, and $d = 2R$ the diameter of the sphere. The viscous time scale, T_μ, is calculated as follows [59]:

$$T_\mu = \frac{\rho d^2}{\mu}, \tag{63}$$

where μ is the dynamic viscosity of the fluid inside the sphere. The Laplace number La,

$$La = \frac{T_\mu^2}{T_\sigma^2}, \tag{64}$$

is an important dimensionless parameter characterising the problem. The two other control parameters are the ratios of the densities and dynamic viscosities for the two fluids. During the simulation performed here, the time step is limited by the capillary wave condition [49]:

$$\Delta t \le C_{\Delta t}\sqrt{\frac{(\rho_1 + \rho_2)\Delta x^3}{\sigma}}, \tag{65}$$

where ρ_1 and ρ_2 are the densities of the fluid on either side of the interface. Even though the condition in Eq. (65) was originally developed for 2D Cartesian geometry [49,60], it is also applicable for spherical surfaces, since in that case it can be derived by the requirement that the time step be smaller than the lowest admitted eigenperiod of the spherical droplet oscillations [61]. Note that the exact value of the coefficient $C_{\Delta t}$ has not been agreed upon in the literature [53]: for example, Brackbill et al. [49] derived $C_{\Delta t} = 1/\sqrt{4\pi} \approx 0.282$, while Sussman and Ohta [60] suggest $C_{\Delta t} = 1/\sqrt{8\pi^3} \approx 0.063$.

Correct implementation of the surface tension force in our code has been verified for three test cases; their characteristics are summarised in Table 8. Four grid resolutions have been employed (16, 32, 48, 64), each measured in terms of number of cells per domain width L. Note that for the "droplet-like" cases, several million time steps are required to reach equilibrium conditions.

In all simulations, decay of parasitic currents has been observed. As illustration, Fig. 21 shows the relaxation process for the "droplet-like" and"bubble-like" cases for different grid resolutions. The capillary velocity scale used here for normalisation is given by [59]:

$$U_\sigma = \frac{d}{T_\sigma}, \tag{66}$$

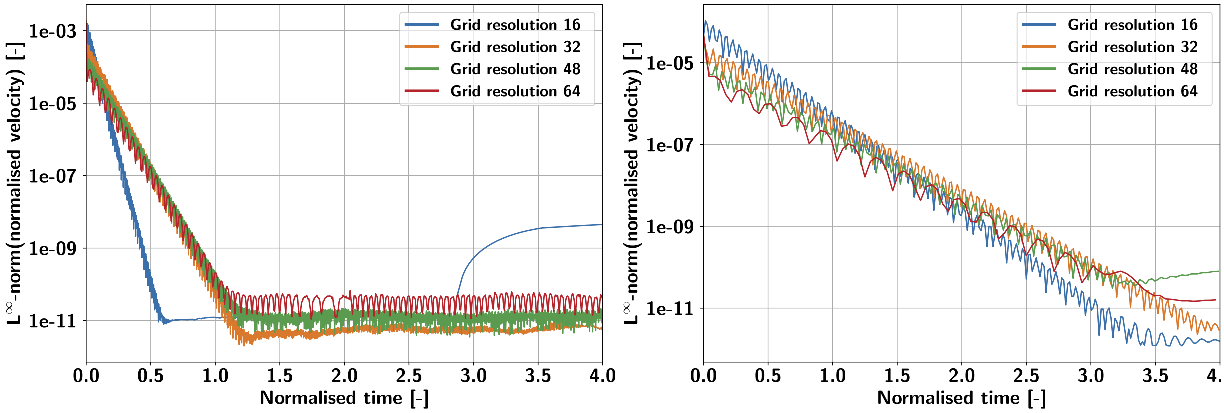


Fig. 21. L^∞-norm of the z-component of the velocity field in the domain, normalised in terms of the capillary velocity, as a function of time, itself normalised in terms of the viscous time scale for the "droplet-like" (left) and "bubble-like" (right) static parasitic current problems; results of four different grid resolutions (measured in terms of number of cells per the domain width L) are shown.

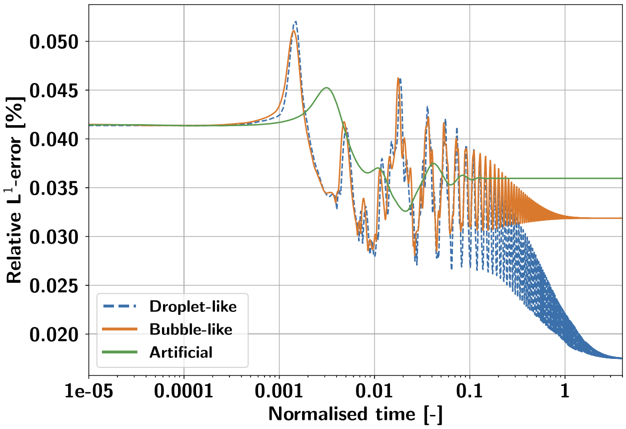


Fig. 22. Relative L^1-error of curvature as a function of time, normalised in terms of the viscous time scale, for the static parasitic current problem. Results with grid resolution 32 are shown.

with the L^∞-norm defined as:

$$L^\infty\text{-norm} = \max_{\forall \text{ cells}} \left|\left|\frac{w}{U_\sigma}\right|\right|. \tag{67}$$

According to Popinet [59], the interfacial shape adjusts itself during the relaxation process to reach the static numerical equilibrium solution of the Young-Laplace equation:

$$\Delta p = \sigma \kappa, \tag{68}$$

where Δp is the pressure jump across the interface. The relative L^1-error of the interfacial curvature in Fig. 22 is defined as:

$$L^1\text{-error} = \frac{1}{N_\gamma} \sum_{i=1}^{N_\gamma} \left|\left|\frac{\kappa_i - \kappa_{theor}}{\kappa_{theor}}\right|\right|, \tag{69}$$

with N_γ being the number of interfacial cells, and κ_{theor} the theoretical value of the curvature of the sphere, equal to:

$$\kappa_{theor} = \frac{2}{R} = \frac{2}{0.8L} = 1250\,\text{m}^{-1}. \tag{70}$$

The evolution of the L^1-error is shown in Fig. 22 for all three cases (droplet-like, bubble-like, artificial), each with a grid resolution of 32. Note that as the Laplace number increases (see Table 8) the relaxation process becomes longer, since the dampening effect of viscosity decreases. It should also be noted that, at the end of the simulation, the L^1-error = L^∞-error for all cases. This provides further evidence that equilibration has indeed been achieved.

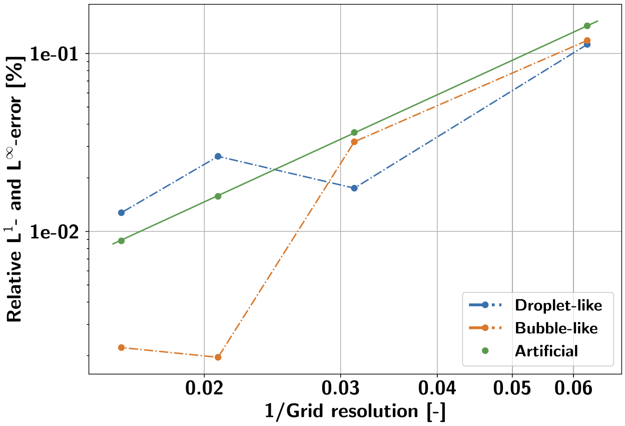


Fig. 23. Relative error of curvature as a function of the reciprocal of grid resolution (measured in terms of number of cells per the domain width L) for the static parasitic current problem; the L^1- and L^∞-errors are numerically identical. The solid line corresponds to a quadratic fit of the data points.

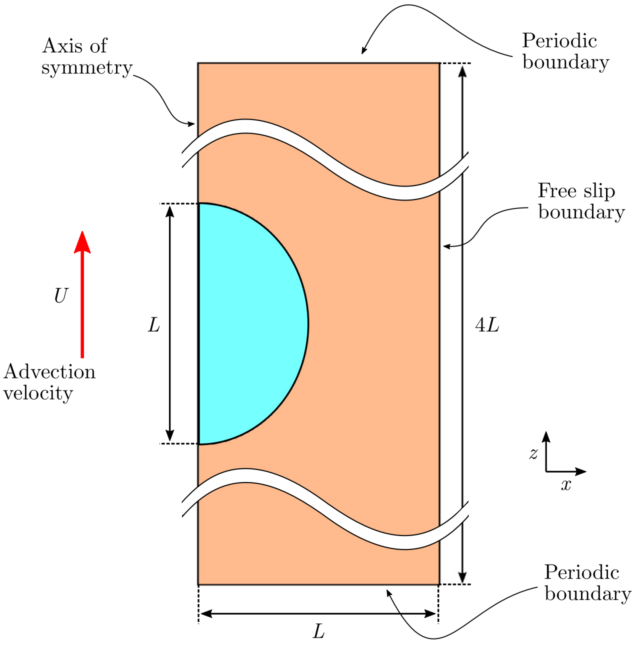
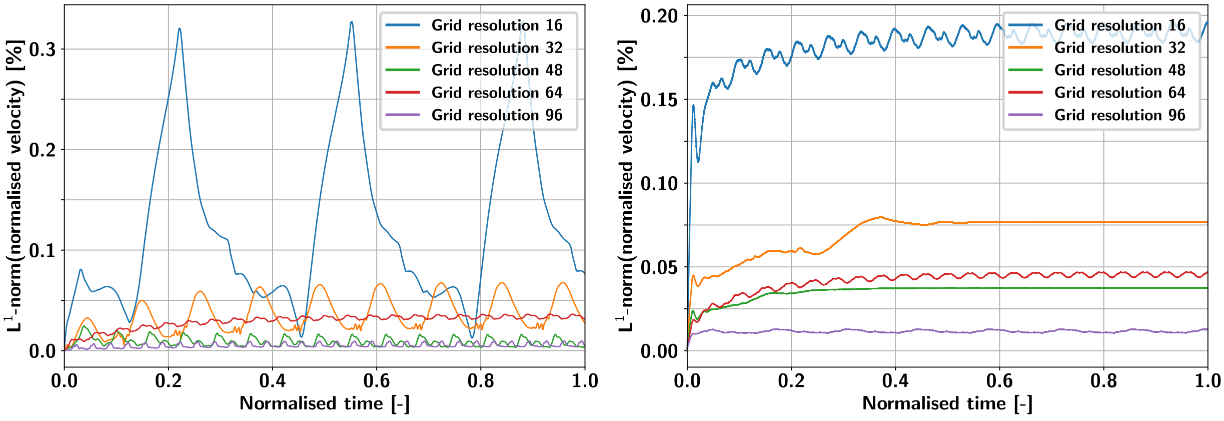


Fig. 24. Schematic representation of the domain used for the dynamic parasitic current problem.

Fig. 23 shows the final relative error in the curvature (Eq. (69)) as a function of the grid resolution for all three cases. It can be seen that for the artificial case, in which the densities and viscosities are equal, second-order accuracy has been achieved. For the bubble-like case, the error decays to negligible levels with higher grid resolution, while for the droplet-like case, the grid dependency cannot be clearly discerned. This adverse behaviour, as well as the observed non-monotonicity, are issues currently under further investigation. However, note that in all cases the error is anyway extremely small ($\ll 1\%$), and the solutions are stable.

3.3.2. Parasitic-current problem in advected configuration (dynamic)

As argued by Popinet [53,59], the parasitic current problem in a stagnant configuration should be complemented by a corresponding dynamic study, in which a sphere is advected with constant velocity. In [59], for example, it was found that the simulation exhibited poor convergence characteristics, as the interface was continuously perturbed during advection. In the present study, advection in the axial direction has been tested for each of the three cases listed in Table 8. The domain is taken as periodic in the z-direction, and a free-slip boundary condition is applied in the radial direction; see Fig. 24. The radius of the advected sphere is taken as $0.5L$, where $L = 2$ mm is the domain radial extent. Two extreme values of the advection velocity are considered: $0.01U_\sigma$ and $10U_\sigma$. Five grid resolutions are adopted (16, 32, 48, 64, 96), each measured in terms of number of cells per domain width L.



(a) Advection velocity $0.01U_\sigma$.

(b) Advection velocity $10U_\sigma$.

Fig. 25. L^1-norm of the velocity field in the domain, normalised in terms of the capillary velocity, as a function of time, itself normalised in terms of the viscous time scale, for the "artificial" dynamic parasitic current problem; results for five grid resolutions are shown.

Table 9
Characteristics of the axisymmetric dam-break problem [62], and experimental results.

ID	Initial radius R_0 [cm]	Initial height H_0 [cm]	Arrival time at wall [s]	Time of max. height [s]	Maximal height [cm]
D1-1	5.5	5	0.24 ± 0.02	0.34 ± 0.02	3 ± 1
D1-2	5.5	10	0.21 ± 0.02	0.36 ± 0.02	9 ± 1
D1-3	5.5	20	0.20 ± 0.02	0.42 ± 0.02	16 ± 1
D2-1	9.5	10	0.16 ± 0.02	0.34 ± 0.02	14 ± 1
D2-2	9.5	20	0.15 ± 0.02	0.40 ± 0.02	22 ± 1

For the artificial case, the parasitic currents do not decay to zero, but rather display a stable, oscillatory behaviour throughout the simulation: see Figs. 25a and 25b. This can be attributed to the fact that the advection scheme fails to preserve the spherical shape of the advected object, and corrective fluxes induced by the surface tension force arise as a result. The anomalous increase in magnitude of the parasitic currents for grid resolution 64, for both advection velocities, remains under investigation. Nonetheless, the results are encouraging in as far as the present algorithm does not induce divergent solutions.

In cases for which unequal dynamic viscosities are featured, irregular and even divergent behaviour has been observed when the weighted-harmonic mixing rule (Eq. (34)) is used. By switching to the classical harmonic mixing rule (Eq. (35)), stable solutions, with parasitic currents of negligible amplitude, could be achieved for the bubble-like cases. For the droplet-like cases, the solutions were also stable, but with adverse properties:

- For an advection velocity of $0.01U_\sigma$, the magnitude of the parasitic currents was comparable to, or even exceeded, the applied advection velocity, with occasional "spikes" occurring for some of the grid resolutions used.
- For an advection velocity of $10U_\sigma$, the magnitude of the parasitic currents *increased* with the grid refinement.

Note that such problems always diminish if equal dynamic viscosities are imposed for the droplet-like cases. It appears that the representation of the stress balance at the phasic interface under dynamic conditions requires further investigation, which remains a task for future work.

3.4. Cylindrical dam break validation test

The dam-break problem is a commonly-used experimental benchmark for testing interfacial tracking methodology [64]. In this problem, a free-standing column of liquid, centrally located in a cylindrical vessel, is allowed to collapse under gravity. A diverging wave spreads out radially at the base of the column. Then, depending on the particular experimental set-up, sloshing, with different characteristics, takes place. The experimental study was performed at the Karlsruhe Institute of Technology by Maschek et al. [62,63]; the set of centralised dam-break experiments without obstacles, presented in Table V.1.1 of their report [62], is used here for validation purposes.

The computational domain has dimensions $R \times H$, where $R = 22$ cm is the radius of the container and H is taken as double the height of the initial column H_0, this to ensure that the upper outlet boundary condition has no influence on the subsequent dynamics; the radius of the initial column is R_0. Fig. 26 is a schematic representation of the fluid domain, and Table 9 gives the values of R_0 and H_0 for all the cases studied here. Water, surrounded by air at room temperature,

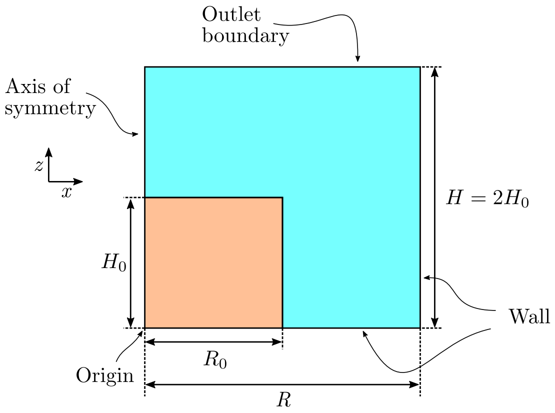


Fig. 26. Schematic representation of the domain used for the dam-break problem.

Table 10
Physical properties of fluids used in the dam break problem.

	Density [kg/m^3]	Dynamic viscosity [Pa·s]	Surface tension [N/m]
Water	997.8	9.53×10^{-4}	0.072
Air	1.2	1.82×10^{-5}	

Table 11
Simulation results for the axisymmetric dam-break problem.

ID	Initial radius R_0 [cm]	Initial height H_0 [cm]	Arrival time at wall [s]	Time of max. height [s]	Maximal height [cm]
D1-1	5.5	5	0.23	0.31-0.32	5.0
D1-2	5.5	10	0.20	0.32-0.34	13.5
D1-3	5.5	20	0.20		
D2-1	9.5	10	0.15	0.30-0.33	17.5
D2-2	9.5	20	0.15		

and at atmospheric pressure, was used as the test liquid [62]. Physical properties of both fluids are presented in Table 10, and reflect normalised atmospheric conditions; surface tension is considered in the simulation. Due to the negligible effect of surface wetting on the problem dynamics, the wall adhesion force is neglected on the container walls. A uniform grid discretisation is employed for all cases. A variable time step is used, with the limit imposed by the Courant number, here taken as $CFL < 0.2$. A second upper limit is given by the capillary wave condition (Eq. (65)), with $C_{\Delta t} = 0.126$: the absolute minimum of these two criteria denoting the time step actually used in the simulation.

In the original experiments [62], the macroscopic characteristics (time of arrival of liquid at the outer wall, time of maximal ascent at the outer constraining wall, etc.) have all been recorded, and have provided valuable data for validation exercises. Some of these quantities have been reproduced in Table 9, just for reference purposes. Note that the time of maximal ascent at the outer constraining wall represents the limit of applying the axisymmetric assumption to the problem, as confirmed by the photographs presented in [62], in which three-dimensional effects are observed.

The time of arrival at the outer wall is a well-defined quantity, which was measured with high precision in the experiments. Moreover, the flow is essentially perfectly axisymmetric before impact on the container wall. Thus, predictions of this quantity have been recorded for all the simulations performed here, and are presented in Table 11. As can be seen from the comparison with the measured values in Table 9, very good agreement has been achieved, all predictions falling within the error bounds of the experimental values.

Note that this problem exhibits non-monotonic convergence behaviour with respect to the degree of grid refinement, as illustrated in Fig. 27. With a grid resolution less than 880 cells per domain radius R (i.e. grid spacing $\Delta x = 0.25$ mm), almost no variation in the arrival time of the wave at the confining wall is observed as a function of grid refinement. However, by further increasing the grid resolution (i.e. using ever smaller meshes), significant improvement in overall agreement with measured data is achieved. This can be explained by considering that the arrival time is affected primarily by the overall momentum of the radially expanding wave, and only secondarily by its actual physical shape. To capture the scales required to resolve the overall momentum, it appears only a coarse grid is necessary. But, to capture the fine details of the wave, a high level of grid refinement must be employed. A qualitative comparison of the shape of the diverging wave as a function of grid resolution is shown in Fig. 28. Evidently, the leading edge is accurately resolved only in the case of fine grid resolution, which, as a consequence, improves the accuracy of the simulation results.

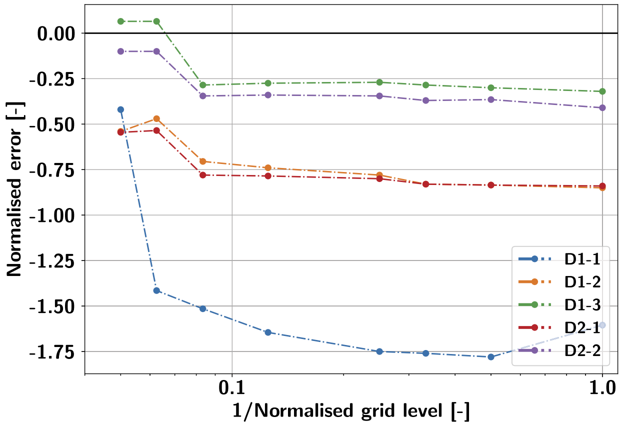


Fig. 27. Deviation of calculated wall-arrival times from experimental values as functions of the reciprocal of grid resolution (measured in terms of number of cells per domain radius R) for the cylindrical dam-break problem. Values on the abscissa are normalised in terms of the coarsest grid resolution considered (88 cells per domain radius R) and values on the ordinate in terms of the experimental standard deviation (0.02 s). Refer to Table 9 for an explanation of the legend.

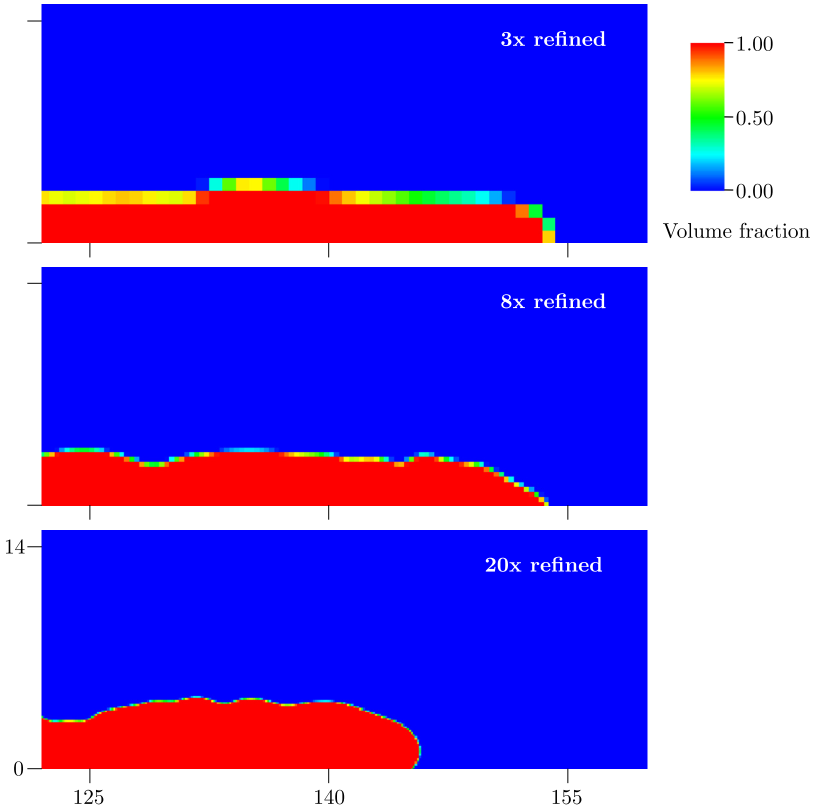


Fig. 28. Details of the leading edge of the diverging wave for the D1-1 axisymmetric dam-break problem. Three different levels of grid refinement are shown; the base grid resolution is 88 cells per domain radius R. Units are in mm.

The importance of properly resolving the leading edge of the diverging wave diminishes with increasing initial column height in the simulations, as demonstrated by the diminishing effect of grid refinement on error shown in Fig. 27. This can be attributed to the fact that the higher the initial column height, the higher the initial potential energy of the column, and the greater the kinetic energy of the advancing radial wave, with the dynamics of the situation becoming progressively more momentum-dominated.

After the impact of the liquid on the outer wall, the complexity of the interface structure gradually increases with the corresponding loss of axial symmetry. This becomes even more pronounced with increasing energy of the wave. Furthermore, corresponding generation of high-velocity eddies in the gas phase imposes limitations on the time step determined by

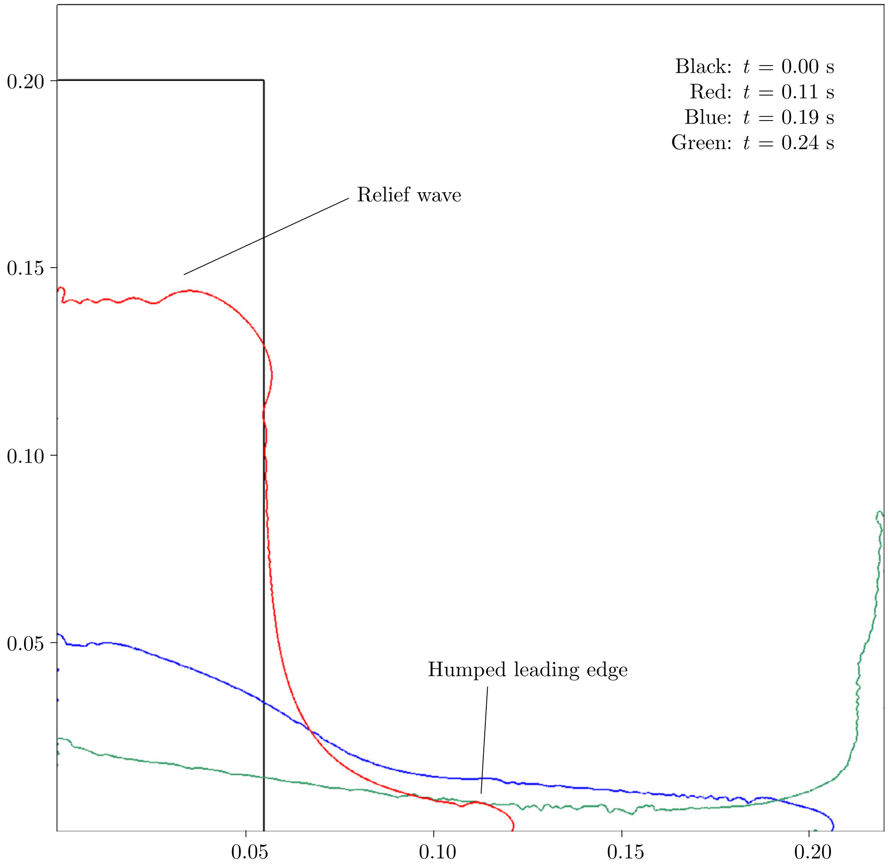


Fig. 29. Evolution of the phasic interface in the D1-3 axisymmetric dam-break problem; grid resolution 1760 cells per domain radius R. Units are in m.

the *CFL* condition. For these reasons, present simulations have been continued until the time of maximal height on the outer containing cylinder only for the three cases with less energetic impacts, and earlier times of maximal height (D1-1, D1-2 and D2-1). Predictions for the maximal height of the outer sloshing, as well as its time, are presented in Table 11. While the times to maximum height are quite well-predicted, the heights themselves are over-estimated somewhat. This could be attributed to the fact that the higher the symmetry of the configuration, the more synchronised the system dynamics become [62], and in the case of our simulations, perfect cylindrical symmetry is inherently assumed, of course. Note that this could also explain the under-estimation of the arrival times and times of maximal height obtained from the simulations (although these values remain within the range of experimental uncertainty).

To illustrate the overall evolution of the flow, Fig. 29 shows several snapshots from the numerical simulation for the D1-3 case, for which experimental photographs are available [62]. Several characteristics of the problem have been described in [62], in particular the relief wave on top of the collapsing column and the water "hump" at the leading edge during the horizontal outward spreading at the base: both aspects can be clearly identified from the simulation. For qualitative comparison purposes only, selected experimental snapshots of this case are reproduced in Fig. 30.

3.5. Single air bubble rise in liquid due to buoyancy

Simulating the rise of a single, initially spherical, bubble in a stagnant liquid is a challenging test of the stability and effectiveness of the coupling of the interface tracking method with the underlying momentum solver, and the treatment of surface tension. In particular, bubbles rising in a highly viscous liquid (e.g. a water-glycerin mixture) deform significantly, and a variety of bubble shapes are often observed [65]. In order to accurately reproduce the terminal rise velocity, and shape of the bubble, effects of external acceleration, viscosity and surface tension must be correctly accounted for, as reported in [64,66].

To represent this problem in axisymmetric geometry, we analytically initialise a circle of radius R at the bottom of a rectangular domain of dimensions $L \times H$, and then calculate the corresponding volume fraction field using our forward reconstruction algorithm, as explained earlier. In order to minimise the wall effects on the computational results, we choose $L = 8R$, as per the recommendations of Hua et al. [66] and the experimental findings of Krishna et al. [67]. The domain height in the axial direction H is taken as a suitable multiple of R to accommodate the full motion of the bubble in a

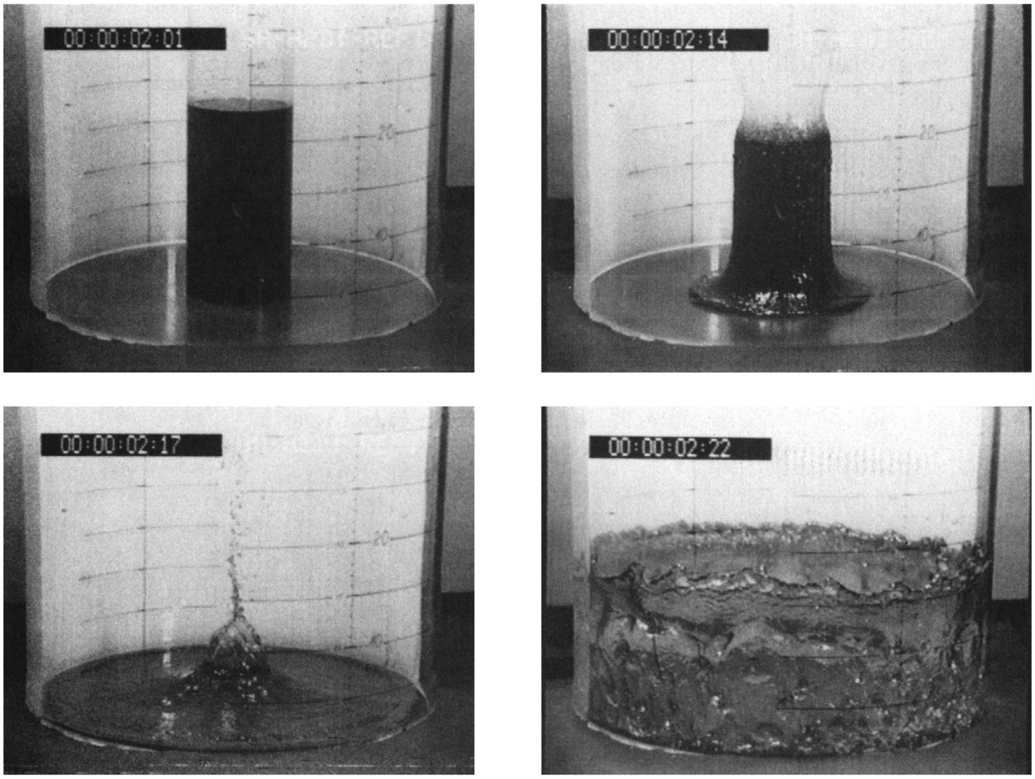


Fig. 30. Snapshots of the D1-3 axisymmetric dam-break experiment [63][6].

static reference frame; here, we choose $H = 30R$. A free-flow outlet boundary condition is applied at $z = H$, and no-slip wall boundary conditions are applied at $z = 0$ and $x = L$. Gravity is considered to act in the negative z-direction. Fig. 31 is a schematic representation of the fluid domain.

A uniform grid discretisation is employed with 33 cells per initial bubble radius, corresponding to 261 360 grid cells in total. Grid convergence at this resolution was first confirmed by means of grid sensitivity studies performed for several of the considered cases. A variable time step is used, with the limit imposed by the Courant number, taken here as $CFL < 0.25$. A second upper limit on the time step is given by the capillary wave condition (Eq. (65)) with $C_{\triangle t} = 0.126$, and the absolute minimum of these two criteria is the time step actually used in the simulation.

To characterise the problem, the following dimensionless groups are commonly formed from the liquid properties (gas properties have a negligible effect on the dynamics of the problem [65]):

- Morton number Mo:

$$Mo = \frac{g\mu_l^4}{\rho_l\sigma^3}, \tag{71}$$

- Eötvös number Eo:

$$Eo = \frac{gD_0^2\rho_l}{\sigma}, \tag{72}$$

- Terminal Reynolds number Re_t:

$$Re_t = \frac{\rho_l D_0 U_t}{\mu_l}, \tag{73}$$

- Terminal Weber number We_t:

$$We_t = \frac{\rho_l D_0 U_t^2}{\sigma} = Re_t^2\sqrt{\frac{Mo}{Eo}}. \tag{74}$$

[6] Reprinted from [63], p. 31, Copyright (1992), with permission from Taylor & Francis.

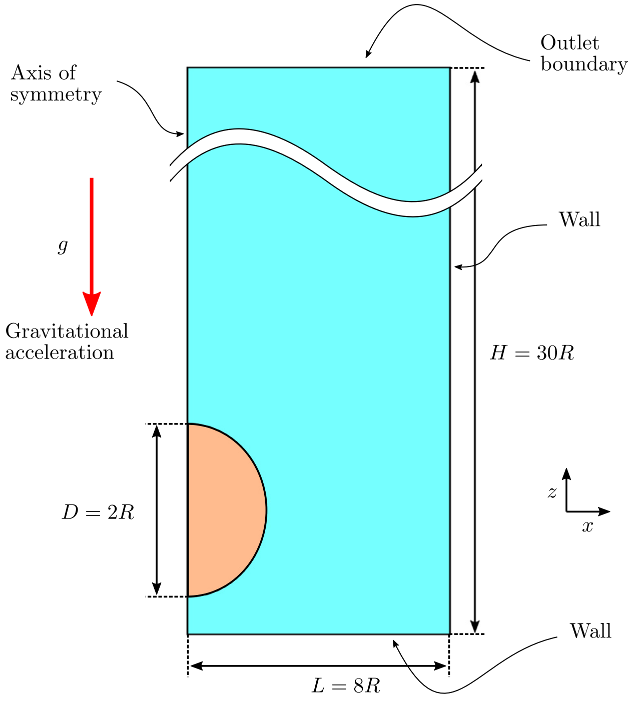


Fig. 31. Schematic representation of the domain used for the bubble rise problem.

Table 12
Characteristics of the cases for the bubble rise in a stagnant liquid together with calculated Reynolds and Weber numbers.

Case	Ref.	D_0 [mm]	ρ_l [kg/m^3]	μ_l [Pa·s]	σ [N/m]	Mo [-] (Eq. (71))	Eo [-] (Eq. (72))	Re_t [-] (Eq. (73))	We_t [-] (Eq. (74))
1	[65]	3.9	1258	0.4578	0.0588	1.68	3.19	0.488	0.173
2	[65]	5.9	1258	0.4578	0.0588	1.68	7.31	1.51	1.10
3	[65]	5.9	1242	0.1735	0.0626	0.0292	6.78	6.81	3.05
4	[65]	5.9	1225	0.0800	0.0627	0.00133	6.67	18.2	4.66
5	[65]	9.0	1258	0.4578	0.0588	1.68	17.0	3.91	4.81
6	[65]	9.0	1242	0.1735	0.0626	0.0292	15.8	13.0	7.31
7	[65]	9.0	1225	0.0800	0.0627	0.00133	15.5	29.1	7.81
8	[68]	12.2	875.5	0.118	0.0322	0.0651	39.4	20.1	16.4
9	[69]	26.1	1350	2.722	0.0778	848	116	3.31	29.7
10	[69]	26.1	1350	2.038	0.0778	266	116	4.92	36.7
11	[69]	26.1	1350	1.277	0.0778	41.1	116	8.63	44.3
12	[69]	26.1	1350	0.7728	0.0778	5.51	116	14.8	47.6

D_0: initial bubble diameter, ρ_l: liquid density, μ_l: liquid dynamic viscosity, σ: surface tension, Mo: Morton number, Eo: Eötvös number, Re_t: terminal Reynolds number, We_t: terminal Weber number.
Gas density ρ_g [kg/m^3]: 1.204.
Gas dynamic viscosity μ_g [Pa·s]: 1.85×10^{-5}.

In these groups, D_0 is the diameter of a volume-equivalent sphere (initial bubble diameter in our simulations), ρ_l the liquid density, μ_l the liquid dynamic viscosity, σ the surface tension, g the gravitational acceleration, and U_t the bubble terminal rise velocity.

The experimental data of Legendre et al. [65], Hnat and Buckmaster [68] and Bhaga and Weber [69] are taken for comparison purposes. Table 12 lists the characteristics of all the cases considered, together with calculated terminal Reynolds and Weber numbers. Fig. 32 shows a visual comparison of the computed bubble shapes against measured experimental data. Note that to obtain snapshots of the entire bubble cross-section, the plots of the simulation results have been mirrored around the axis of symmetry during post-processing. It can be seen that the various types of deformed bubble shapes (i.e. spherical, ellipsoidal and spherical-cap) are all well predicted. Fig. 33 is a plot of the calculated terminal Reynolds numbers against the correlation group of Rastello et al. [70]:

$$Re_{t,cor} = 2.05 We_t^{2/3} Mo^{-1/5}. \tag{75}$$

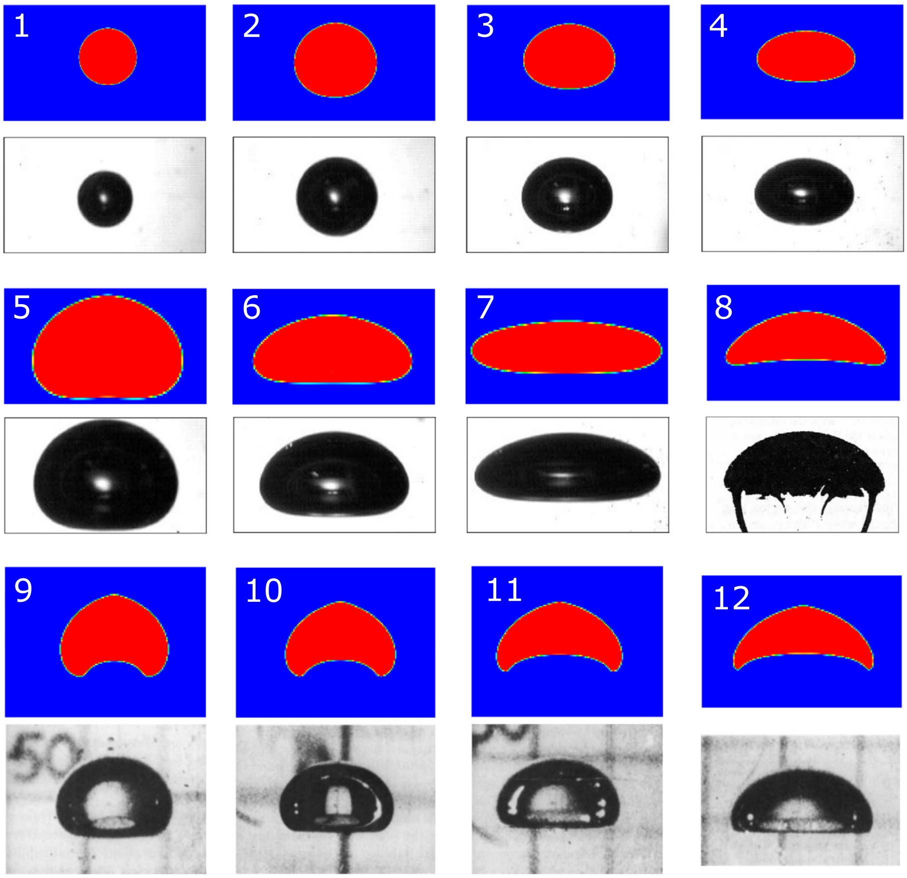


Fig. 32. Comparison of simulated and experimental bubble shapes.[7] Numbers correspond to case identifiers listed in Table 12. The shadow observed below the bubble in Case 8 is an artefact of the experimental measurement technique [68].

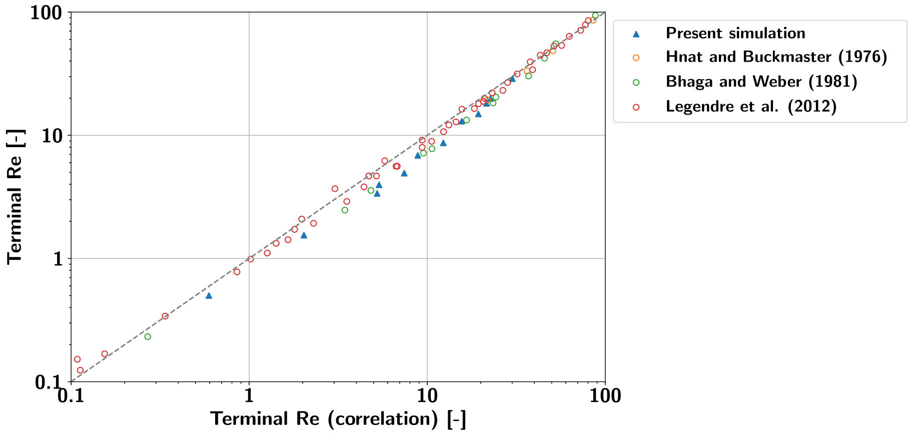


Fig. 33. Comparison of terminal Reynolds numbers against the correlation group of Rastello et al. (Eq. (75)).

Selected experimental results from [65,68,69] are also shown. Good agreement between our computed values and experimental data has clearly been achieved.

[7] Reprinted from [65], p. 3, Copyright (2012), with permission from AIP Publishing, from [68], p. 182, Copyright (1976), with permission from AIP Publishing, and from [69], p. 66, Copyright (1981), with permission from Cambridge University Press.

4. Conclusions

In this work, analytical solutions of the forward and inverse PLIC-VOF interface reconstruction procedure in axisymmetric, cylindrical geometry have been presented. To the best of authors' knowledge, this represents the first solution of its type in such a configuration. Our direct analytic solution procedure outperforms the iterative solution method with a speed-up of about 1.4-1.7. Full implementation of the PLIC-VOF algorithm for axisymmetric cylindrical geometry is also presented here for the first time in the open literature, including a method for curvature estimation using height functions.

A number of benchmark simulations have been performed to test the efficacy of the numerical algorithm. First, a verification exercise has been performed using a set of forced advection cases, as well as the *standard parasitic current problem*. The advection problems successfully demonstrate the solution of the transport of the free surface under a prescribed velocity field, with numerical predictions in good agreement with analytical results. For the circular vortex advection case, a comparison of error measures of our method and contemporary implementations of the axisymmetric PLIC-VOF method in the commercial CFD software Ansys® Fluent and open-source CFD code Basilisk demonstrates the superior performance of our method in terms of accuracy.

The static parasitic current problems considered here show that the discrete form of the Young-Laplace equilibrium condition can be achieved, after a sufficiently long relaxation time, for a variety of grids and material properties. For configurations in which both phases have the same density and dynamic viscosity, second-order accuracy is achieved. Furthermore, for these configurations, a stable solution of the dynamic parasitic problem involving repeated perturbation of the phasic interface by continuous advection could be demonstrated, with parasitic currents shown to be negligible. However, grid-convergent solutions of the dynamic problem could not be attained for all configurations involving non-trivial viscosity ratios. It appears that the representation of the stress balance at the phasic interface under dynamic conditions requires further investigation.

To validate the algorithm developed here, the well-known axisymmetric dam-break problem and bubble rise in a quiescent liquid problem have also been simulated. For the dam-break problem, very good agreement of simulation results with experimental data has been attained, and the dynamic characteristics of the problem well reproduced. For the bubble rise problem, bubble shapes at various degrees of deformation have been well predicted, and good agreement between computed values of terminal Reynolds numbers with those derived from the experimental data has been achieved.

In future work, the wall-adhesion force, represented using height functions, will be included in the algorithm. This will enable wall-bounded flows to be simulated. Subsequently, the algorithm will be extended to treat problems involving heat, mass and species transfer, by incorporating a sharp-interface phase change model and species transport solver.

CRediT authorship contribution statement

Lubomír Bureš: Conceptualization, Methodology, Software, Validation, Visualization, Writing – original draft. **Yohei Sato:** Conceptualization, Funding acquisition, Project administration, Supervision, Writing – review & editing. **Andreas Pautz:** Project administration, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The authors would like to thank Dr. Brian Smith of Goldsmith Transactions for valuable technical discussions, advice and English corrections. This work is supported by the Swiss National Science Foundation (SNSF) under Grant No. 200021_175893.

Appendix A. Solution of the inverse problem

With reference to the terminology introduced in Section 2.1.2, we demonstrate the solution of the inverse problem for the case $m_x \leq m_z$, $n_x \geq 0$. The derivation for other cases follows analogously. At the transition point, which it will be recalled is the point where two different branches of the solution connect, the following relation holds:

$$\phi_{tr} = \frac{m_x}{2m_z}\frac{\eta_0 + 1/3}{\eta_0 + 1/2}. \tag{A.1}$$

A.1. $\phi < \phi_{tr}$

For $\phi < \phi_{tr}$, $\alpha(\phi)$ is the solution of the cubic equation:

$$\left(\eta_0 + \frac{1}{2}\right)\phi = \frac{\alpha^2}{2m_x m_z}\left(\eta_0 + \frac{\alpha}{3m_x}\right), \tag{A.2}$$

which can be re-arranged (using standard notation for the cubic equation coefficients a, b, c) as:

$$\alpha^3 + a\alpha^2 + c = \alpha^3 + 3m_x\eta_0\alpha^2 - 6m_x^2 m_z\left(\eta_0 + \frac{1}{2}\right)\phi = 0. \tag{A.3}$$

Note that the coefficient b is absent here, since there is no linear term in the equation. According to the Descartes' rule of signs [71], only one solution of this equation is positive; this is the one being sought. Following the standard rules for the solution of cubic equations [44,71], we define:

$$p := -\frac{a^2}{3} = -3m_x^2\eta_0^2; \tag{A.4}$$

$$q := \frac{2a^3}{27} + c = 2m_x^3\eta_0^3 - 6m_x^2 m_z\left(\eta_0 + \frac{1}{2}\right)\phi = 2m_x^2\left[m_x\eta_0^3 - 3m_z\left(\eta_0 + \frac{1}{2}\right)\phi\right]. \tag{A.5}$$

The cubic determinant Δ is then determined according to:

$$\Delta = \frac{q^2}{4} + \frac{p^3}{27} = 3m_x^4 m_z\phi\left(\eta_0 + \frac{1}{2}\right)\left[3m_z\left(\eta_0 + \frac{1}{2}\right)\phi - 2m_x\eta_0^3\right]. \tag{A.6}$$

It can be seen that the determinant can attain negative, zero or positive values; the sign can be decided from the critical value (note that here $m_z > 0$):

$$\phi_{crit} = \frac{2m_x}{3m_z}\frac{\eta_0^3}{\eta_0 + 1/2} = \frac{4}{3}\frac{\eta_0^3}{\eta_0 + 1/3}\phi_{tr}. \tag{A.7}$$

Using the above definition of ϕ_{crit}, together with the definition

$$M := 3m_x^2 m_z\left(\eta_0 + \frac{1}{2}\right), \tag{A.8}$$

q can be rewritten as

$$q = 2m_x^2\left[m_x\eta_0^3 - 3m_z\left(\eta_0 + \frac{1}{2}\right)\phi\right] = M\left(\phi_{crit} - 2\phi\right), \tag{A.9}$$

p as

$$p = -3m_x^2\eta_0^2 = -3\left(\frac{M\phi_{crit}}{2}\right)^{2/3}, \tag{A.10}$$

and finally the determinant Δ as

$$\Delta = 3m_x^4 m_z\phi\left(\eta_0 + \frac{1}{2}\right)\left[3m_z\left(\eta_0 + \frac{1}{2}\right)\phi - 2m_x\eta_0^3\right] = M^2\phi\left(\phi - \phi_{crit}\right). \tag{A.11}$$

A.1.1. $\Delta < 0, \phi < \phi_{crit}$

For $\phi < \phi_{crit}$, the determinant is negative. Then, three real solutions exist:

$$\alpha_1 = \frac{a}{3}\left[-1 + 2\sin\left(\frac{\varphi_0}{3}\right)\right], \tag{A.12}$$

$$\alpha_2 = \frac{a}{3}\left[-1 - 2\sin\left(\frac{\varphi_0}{3} + \frac{\pi}{3}\right)\right], \tag{A.13}$$

$$\alpha_3 = \frac{a}{3}\left[-1 + 2\cos\left(\frac{\varphi_0}{3} + \frac{\pi}{6}\right)\right], \tag{A.14}$$

with φ_0 calculated according to:

$$\varphi_0 = \arcsin\left(\frac{27q}{2a^3}\right) = \arcsin\left(1 - \frac{2\phi}{\phi_{crit}}\right), \tag{A.15}$$

in which the coefficient $a/3$ may be expressed in terms of M and ϕ_{crit} as

$$\frac{a}{3} = \left(\frac{M\phi_{crit}}{2}\right)^{1/3}. \tag{A.16}$$

It can be readily proved that $\alpha_3 \geq \alpha_1$ and $\alpha_3 \geq \alpha_2$; therefore, α_3 is the only positive root, which is the one sought for in this algorithm.

A.1.2. $\Delta > 0$, $\phi > \phi_{crit}$

For $\phi > \phi_{crit}$, the determinant is positive. Then, only one real solution exists, and this has the form:

$$\begin{aligned}\alpha_{sol} &= \left(-\frac{q}{2}+\sqrt{\Delta}\right)^{1/3}+\left(-\frac{q}{2}-\sqrt{\Delta}\right)^{1/3}-\frac{a}{3} \\ &= \left(\frac{M}{2}\right)^{1/3}\left[\left\{2\phi-\phi_{crit}+2\sqrt{\phi(\phi-\phi_{crit})}\right\}^{1/3}+\left\{2\phi-\phi_{crit}-2\sqrt{\phi(\phi-\phi_{crit})}\right\}^{1/3}-\phi_{crit}^{1/3}\right].\end{aligned} \tag{A.17}$$

A.1.3. $\Delta = 0$, $\phi = \phi_{crit}$

For the case $\phi = \phi_{crit}$, the determinant is zero. Then, two real solutions exist:

$$\alpha_1 = -2\left(\frac{q}{2}\right)^{1/3}-\frac{a}{3} = \left(\frac{M}{2}\right)^{1/3}\left[2\left\{2\phi-\phi_{crit}\right\}^{1/3}-\phi_{crit}^{1/3}\right], \tag{A.18}$$

$$\alpha_2 = \left(\frac{q}{2}\right)^{1/3}-\frac{a}{3} = \left(\frac{M}{2}\right)^{1/3}\left[-\left\{2\phi-\phi_{crit}\right\}^{1/3}-\phi_{crit}^{1/3}\right]. \tag{A.19}$$

Evidently, $\alpha_2 \le 0$ and $\alpha_2 = 0$ if and only if $\alpha_1 = 0$. Consequently, α_1 is the desired solution. It can also be recognised that this is just a special case of the formula in which $\Delta > 0$, A.1.2, so these two cases do not have to be treated separately.

A.2. $\phi \ge \phi_{tr}$

For $\phi \ge \phi_{tr}$, $\alpha(\phi)$ is the solution of the linear equation:

$$\left(\eta_0+\frac{1}{2}\right)\phi = \frac{1}{2m_z}\left(2\alpha-m_x\right)\left(\eta_0+\frac{1}{3}\frac{3\alpha-2m_x}{2\alpha-m_x}\right) = \frac{1}{2m_z}\left[(2\alpha-m_x)\eta_0+\frac{1}{3}(3\alpha-2m_x)\right], \tag{A.20}$$

for which there is just one solution:

$$\alpha_{sol} = m_z\phi+\frac{m_x}{2}\frac{\eta_0+2/3}{\eta_0+1/2}. \tag{A.21}$$

This concludes the solution of the inverse problem for all branches of the case in which $m_x \le m_z$, $n_x \ge 0$. Though the solutions appear rather cumbersome, once they have been derived and implemented in the code, major savings in computational efficiency result.

References

[1] C. Hirt, B. Nichols, Volume of fluid (vof) method for the dynamics of free boundaries, J. Comput. Phys. 39 (1) (1981) 201–225, https://doi.org/10.1016/0021-9991(81)90145-5.

[2] M. Rudman, Volume-tracking methods for interfacial flow calculations, Int. J. Numer. Methods Fluids 24 (7) (1997) 671–691, https://doi.org/10.1002/(SICI)1097-0363(19970415)24:7<671::AID-FLD508>3.0.CO;2-9.

[3] O. Ubbink, R. Issa, A method for capturing sharp fluid interfaces on arbitrary meshes, J. Comput. Phys. 153 (1) (1999) 26–50, https://doi.org/10.1006/jcph.1999.6276.

[4] W.F. Noh, P. Woodward, Slic (simple line interface calculation), in: A.I. van de Vooren, P.J. Zandbergen (Eds.), Proceedings of the Fifth International Conference on Numerical Methods in Fluid Dynamics, June 28–July 2, 1976, Twente University, Enschede, Springer Berlin Heidelberg, Berlin, Heidelberg, ISBN 978-3-540-37548-7, 1976, pp. 330–340.

[5] D. Youngs, Time-dependent multi-material flow with large fluid distortion, in: K. Morton, M. Baines (Eds.), Numerical Methods for Fluid Dynamics, Academic Press, 1982, pp. 273–285.

[6] S. Bna, A. Cervone, V. Le Chenadec, S. Manservisi, R. Scardovelli, Review of split and unsplit geometric advection algorithms, AIP Conf. Proc. 1558 (1) (2013) 875–878, https://doi.org/10.1063/1.4825636.

[7] R. Comminal, J. Spangenberg, J.H. Hattel, Cellwise conservative unsplit advection for the volume of fluid method, J. Comput. Phys. 283 (2015) 582–608, https://doi.org/10.1016/j.jcp.2014.12.003.

[8] D. Kothe, W. Rider, S. Mosso, J. Brock, J. Hochstein, Volume tracking of interfaces having surface tension in two and three dimensions, in: 34th Aerospace Sciences Meeting and Exhibit, 1999, AIAA 96-0859.

[9] M. Mehrabadi, M. Bussmann, A 3d vof model in cylindrical coordinates, in: 479th Euromech Colloquium: Numerical Simulation of Multiphase Flow with Deformable Interfaces, August 14–August 16, 2006, Delft University of Technology, Scheveningen, 2006, pp. 120–124.

[10] E.S.C. Fan, M. Bussmann, Piecewise linear volume tracking in spherical coordinates, Appl. Math. Model. 37 (5) (2013) 3077–3092, https://doi.org/10.1016/j.apm.2012.07.006.

[11] S. Mehrabian, N. Abbaspour, M. Bussmann, E. Acosta, The effect of viscosity ratio on the hydrodynamics of separation from an oil-coated particle, in: Fluids Engineering Division Summer Meeting, Vol. 1A, ISBN 978-0-7918-4621-6, 2014, p. 7.

[12] P. Liovic, M. Rudman, J.L. Liow, D. Lakehal, D. Kothe, A 3d unsplit-advection volume tracking algorithm with planarity-preserving interface reconstruction, Comput. Fluids 35 (10) (2006) 1011–1032, https://doi.org/10.1016/j.compfluid.2005.09.003.

[13] L. Jofre, O. Lehmkuhl, J. Castro, A. Oliva, A plic-vof implementation on parallel 3d unstructured meshes, in: Proc. of V European Conference on Computational Fluid Dynamics, ECCOMAS CFD, 2010, pp. 1–15.

[14] M. Huang, L. Wu, B. Chen, A piecewise linear interface-capturing volume-of-fluid method based on unstructured grids, Numer. Heat Transf., Part B, Fundam. 61 (5) (2012) 412–437, https://doi.org/10.1080/10407790.2012.672818.

[15] T. Maric, H. Marschall, D. Bothe, voFoam - a geometrical volume of fluid algorithm on arbitrary unstructured meshes with local dynamic adaptive mesh refinement using openfoam, arXiv:1305.3417, 2013.
[16] L. Jofre, O. Lehmkuhl, J. Castro, A. Oliva, A 3-d volume-of-fluid advection method based on cell-vertex velocities for unstructured meshes, Comput. Fluids 94 (2014) 14–29, https://doi.org/10.1016/j.compfluid.2014.02.001.
[17] C.B. Ivey, P. Moin, Conservative and bounded volume-of-fluid advection on unstructured grids, J. Comput. Phys. 350 (2017) 387–419, https://doi.org/10.1016/j.jcp.2017.08.054.
[18] D. Dai, A.Y. Tong, An analytical interface reconstruction algorithm in the plic-vof method for 2d polygonal unstructured meshes, Int. J. Numer. Methods Fluids 88 (6) (2018) 265–276, https://doi.org/10.1002/fld.4664.
[19] T. Maric, H. Marschall, D. Bothe, An enhanced un-split face-vertex flux-based vof method, J. Comput. Phys. 371 (2018) 967–993, https://doi.org/10.1016/j.jcp.2018.03.048.
[20] J. López, J. Hernandez, P. Gómez, F. Faura, Non-convex analytical and geometrical tools for volume truncation, initialization and conservation enforcement in vof methods, J. Comput. Phys. 392 (2019) 666–693, https://doi.org/10.1016/j.jcp.2019.04.055.
[21] T. Maric, D.B. Kothe, D. Bothe, Unstructured un-split geometrical volume-of-fluid methods – a review, J. Comput. Phys. 420 (2020) 109695, https://doi.org/10.1016/j.jcp.2020.109695.
[22] R. Scardovelli, S. Zaleski, Analytical relations connecting linear interfaces and volume fractions in rectangular grids, J. Comput. Phys. 164 (1) (2000) 228–237, https://doi.org/10.1006/jcph.2000.6567.
[23] R. Scardovelli, S. Zaleski, Interface reconstruction with least-square fit and split Eulerian–Lagrangian advection, Int. J. Numer. Methods Fluids 41 (3) (2003) 251–274, https://doi.org/10.1002/fld.431.
[24] X. Yang, A.J. James, Analytic relations for reconstructing piecewise linear interfaces in triangular and tetrahedral grids, J. Comput. Phys. 214 (1) (2006) 41–54, https://doi.org/10.1016/j.jcp.2005.09.002.
[25] V. Dyadechko, M. Shashkov, Moment-of-fluid interface reconstruction, Tech. Rep., Los Alamos National Laboratory, 2005, LA-UR-05-7571.
[26] W.J. Rider, D.B. Kothe, Reconstructing volume tracking, J. Comput. Phys. 141 (2) (1998) 112–152, https://doi.org/10.1006/jcph.1998.5906.
[27] R.P. Brent, An algorithm with guaranteed convergence for finding a zero of a function, Comput. J. 14 (4) (1971) 422–425, https://doi.org/10.1093/comjnl/14.4.422.
[28] J. Lopez, J. Hernandez, Analytical and geometrical tools for 3d volume of fluid methods in general grids, J. Comput. Phys. 227 (12) (2008) 5939–5948, https://doi.org/10.1016/j.jcp.2008.03.010.
[29] S. Diot, M. François, E. Dendy, An interface reconstruction method based on analytical formulae for 2d planar and axisymmetric arbitrary convex cells, J. Comput. Phys. 275 (2014) 53–64, https://doi.org/10.1016/j.jcp.2014.06.060.
[30] S. Diot, M.M. François, An interface reconstruction method based on an analytical formula for 3d arbitrary convex cells, J. Comput. Phys. 305 (2016) 63–74, https://doi.org/10.1016/j.jcp.2015.10.011.
[31] J. Lopez, J. Hernandez, P. Gómez, F. Faura, A new volume conservation enforcement method for plic reconstruction in general convex grids, J. Comput. Phys. 316 (2016) 338–359, https://doi.org/10.1016/j.jcp.2016.04.018.
[32] H.T. Ahn, M. Shashkov, Geometric algorithms for 3d interface reconstruction, in: M.L. Brewer, D. Marcum (Eds.), Proceedings of the 16th International Meshing Roundtable, Springer Berlin Heidelberg, Berlin, Heidelberg, ISBN 978-3-540-75103-8, 2008, pp. 405–422.
[33] H.R. Anbarlooei, K. Mazaheri, 'Moment of fluid' interface reconstruction method in axisymmetric coordinates, Int. J. Numer. Methods Biomed. Eng. 27 (10) (2011) 1640–1651, https://doi.org/10.1002/cnm.1426.
[34] M.B. Friess, J. Breil, S. Galera, P.H. Maire, M. Shashkov, A multi-material CCALE-MOF approach in cylindrical geometry, arXiv:1111.4900, 2011.
[35] X. Chen, X. Zhang, An improved 3d mof method based on analytical partial derivatives, J. Comput. Phys. 326 (2016) 156–170, https://doi.org/10.1016/j.jcp.2016.08.051.
[36] T. Milcent, A. Lemoine, Moment-of-fluid analytic reconstruction on 3d rectangular hexahedrons, J. Comput. Phys. 409 (2020) 109346, https://doi.org/10.1016/j.jcp.2020.109346.
[37] A. Lemoine, Analytic gradient for the moment-of-fluid method in axisymmetric and on general polyhedrons in any dimension, J. Comput. Phys. 422 (2020) 109741, https://doi.org/10.1016/j.jcp.2020.109741.
[38] A. Lemoine, S. Glockner, J. Breil, Moment-of-fluid analytic reconstruction on 2d Cartesian grids, J. Comput. Phys. 328 (2017) 131–139, https://doi.org/10.1016/j.jcp.2016.10.013.
[39] H.T. Ahn, M. Shashkov, Multi-material interface reconstruction on generalized polyhedral meshes, J. Comput. Phys. 226 (2) (2007) 2096–2132, https://doi.org/10.1016/j.jcp.2007.06.033.
[40] X. Chen, X. Zhang, A predicted-Newton's method for solving the interface positioning equation in the mof method on general polyhedrons, J. Comput. Phys. 384 (2019) 60–76, https://doi.org/10.1016/j.jcp.2018.12.038.
[41] A. Urbano, S. Tanguy, G. Huber, C. Colin, Direct numerical simulation of nucleate boiling in micro-layer regime, Int. J. Heat Mass Transf. 123 (2018) 1128–1137, https://doi.org/10.1016/j.ijheatmasstransfer.2018.02.104.
[42] M. Sussman, P. Smereka, S. Osher, A level set approach for computing solutions to incompressible two-phase flow, J. Comput. Phys. 114 (1) (1994) 146–159, https://doi.org/10.1006/jcph.1994.1155.
[43] D. Gueyffier, J. Li, A. Nadim, R. Scardovelli, S. Zaleski, Volume-of-fluid interface tracking with smoothed surface stress methods for three-dimensional flows, J. Comput. Phys. 152 (2) (1999) 423–456, https://doi.org/10.1006/jcph.1998.6168.
[44] D. Zwillinger, CRC Standard Mathematical Tables and Formulae, Chapman and Hall/CRC, 2002.
[45] S.H. Strogatz, Nonlinear Dynamics and Chaos with Student Solutions Manual: With Applications to Physics, Biology, Chemistry, and Engineering, CRC Press, 2018.
[46] Y. Sato, B. Niceno, A sharp-interface phase change model for a mass-conservative interface tracking method, J. Comput. Phys. 249 (2013) 127–161, https://doi.org/10.1016/j.jcp.2013.04.035.
[47] A. Prosperetti, Navier-Stokes numerical algorithms for free-surface flow computations: an overview, in: M. Rein (Ed.), Drop-Surface Interactions, Vienna: Springer Vienna, ISBN 978-3-7091-2594-6, 2002, pp. 237–257.
[48] A.J. Chorin, Numerical solution of the Navier-Stokes equations, Math. Comput. 22 (1968) 745–762, https://doi.org/10.1090/S0025-5718-1968-0242392-2.
[49] J.U. Brackbill, D.B. Kothe, C. Zemach, A continuum method for modeling surface tension, J. Comput. Phys. 100 (1992) 335–354, https://doi.org/10.1016/0021-9991(92)90240-Y.
[50] E. Aulisa, S. Manservisi, R. Scardovelli, S. Zaleski, Interface reconstruction with least-squares fit and split advection in three-dimensional Cartesian geometry, J. Comput. Phys. 225 (2) (2007) 2301–2319, https://doi.org/10.1016/j.jcp.2007.03.015.
[51] J.E. Pilliod, E.G. Puckett, Second-order accurate volume-of-fluid algorithms for tracking material interfaces, J. Comput. Phys. 199 (2) (2004) 465–502, https://doi.org/10.1016/j.jcp.2003.12.023.
[52] G. Weymouth, D.K.P. Yue, Conservative volume-of-fluid method for free-surface simulations on Cartesian-grids, J. Comput. Phys. 229 (8) (2010) 2853–2865, https://doi.org/10.1016/j.jcp.2009.12.018.
[53] S. Popinet, Numerical models of surface tension, Annu. Rev. Fluid Mech. 50 (1) (2018) 49–75, https://doi.org/10.1146/annurev-fluid-122316-045034.
[54] M. Sussman, A second order coupled level set and volume-of-fluid method for computing growth and collapse of vapor bubbles, J. Comput. Phys. 187 (1) (2003) 110–136, https://doi.org/10.1016/S0021-9991(03)00087-1.

[55] S.J. Cummins, M.M. Francois, D.B. Kothe, Estimating curvature from volume fractions, Comput. Struct. 83 (6–7) (2005) 425–434, https://doi.org/10.1016/j.compstruc.2004.08.017.

[56] J. López, C. Zanzi, P. Gómez, R. Zamora, F. Faura, J. Hernández, An improved height function technique for computing interface curvature from volume fractions, Comput. Methods Appl. Mech. Eng. 198 (33) (2009) 2555–2564, https://doi.org/10.1016/j.cma.2009.03.007.

[57] Ansys®Fluent 2020 R1 Theory Guide, Tech. Rep., 2020.

[58] D.J. Harvie, D.F. Fletcher, A new volume of fluid advection algorithm: the stream scheme, J. Comput. Phys. 162 (1) (2000) 1–32, https://doi.org/10.1006/jcph.2000.6510.

[59] S. Popinet, An accurate adaptive solver for surface-tension-driven interfacial flows, J. Comput. Phys. 228 (16) (2009) 5838–5866, https://doi.org/10.1016/j.jcp.2009.04.042.

[60] M. Sussman, M. Ohta, A stable and efficient method for treating surface tension in incompressible two-phase flow, SIAM J. Sci. Comput. 31 (2009) 2447–2471, https://doi.org/10.1137/080732122.

[61] L.D. Landau, E.M. Lifshitz, Course of Theoretical Physics: Fluid Mechanics, Vol. 6; Chap. 7, 2nd ed., Pergamon Press, 1987, pp. 245–247.

[62] W. Maschek, A. Roth, M. Kirstahler, L. Meyer, Simulation experiments for centralized liquid sloshing motions, Tech. Rep., Karlsruhe Institute of Technology, 1992, 32.02.01; LK 01.

[63] W. Maschek, C.D. Munz, L. Meyer, Investigations of sloshing fluid motions in pools related to recriticalities in liquid-metal fast breeder reactor core meltdown accidents, Nucl. Technol. 98 (1) (1992) 27–43, https://doi.org/10.13182/NT92-A34648.

[64] Y. Sato, B. Niceno, A conservative local interface sharpening scheme for the constrained interpolation profile method, Int. J. Numer. Methods Fluids 70 (4) (2012) 441–467, https://doi.org/10.1002/fld.2695.

[65] D. Legendre, R. Zenit, J.R. Velez-Cordero, On the deformation of gas bubbles in liquids, Phys. Fluids 24 (4) (2012) 043303, https://doi.org/10.1063/1.4705527.

[66] J. Hua, J.F. Stene, P. Lin, Numerical simulation of 3d bubbles rising in viscous liquids using a front tracking method, J. Comput. Phys. 227 (6) (2008) 3358–3382, https://doi.org/10.1016/j.jcp.2007.12.002.

[67] R. Krishna, M. Urseanu, J. van Baten, J. Ellenberger, Wall effects on the rise of single gas bubbles in liquids, Int. Commun. Heat Mass Transf. 26 (6) (1999) 781–790, https://doi.org/10.1016/S0735-1933(99)00066-4.

[68] J.G. Hnat, J.D. Buckmaster, Spherical cap bubbles and skirt formation, Phys. Fluids 19 (2) (1976) 182–194, https://doi.org/10.1063/1.861445.

[69] D. Bhaga, M.E. Weber, Bubbles in viscous liquids: shapes, wakes and velocities, J. Fluid Mech. 105 (1981) 61–85, https://doi.org/10.1017/S002211208100311X.

[70] M. Rastello, J.L. Marie, M. Lance, Drag and lift forces on clean spherical and ellipsoidal bubbles in a solid-body rotating flow, J. Fluid Mech. 682 (2011) 434–459, https://doi.org/10.1017/jfm.2011.240.

[71] F.W. Olver, D.W. Lozier, R.F. Boisvert, C.W. Clark, NIST Handbook of Mathematical Functions, Cambridge University Press, 2010.