HAL
open science

Validation of volume-of-fluid OpenFOAM® isoAdvector solvers using single bubble benchmarks

Lionel Gamet, Marco Scala, Johan Roenby, Henning Scheufler, Jean-Lou Pierson

To cite this version:

Lionel Gamet, Marco Scala, Johan Roenby, Henning Scheufler, Jean-Lou Pierson. Validation of volume-of-fluid OpenFOAM® isoAdvector solvers using single bubble benchmarks. Computers and Fluids, 2020, 213, pp.104722. 10.1016/j.compfluid.2020.104722 . hal-03006811

HAL Id: hal-03006811

https://ifp.hal.science/hal-03006811v1

Submitted on 16 Nov 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Validation of volume-of-fluid OpenFOAM® isoAdvector solvers using single bubble benchmarks

Lionel Gamet[a,*], Marco Scala[a], Johan Roenby[b], Henning Scheufler[c], Jean-Lou Pierson[a]

[a] *IFPEN Lyon, Process Experimentation Division, 69360 Solaize, France*
[b] *Department of Mathematical Sciences, Aalborg University, København SV 2450, Denmark*
[c] *DLR German Aerospace Center, Institute of Space Systems, 28359 Bremen, Germany*

Abstract

Free surfaces and fluid interfaces are encountered in a wide variety of gas-liquid configurations. Although many numerical approaches exist to solve such flows, there is still a need for improved simulation methods. Recently, a new efficient geometric VoF method for general meshes, called isoAdvector, was implemented in OpenFOAM®. More recently, the isoAdvector method was significantly improved by introducing a variant using a reconstructed distance function (RDF) in the interface reconstruction step. Elementary quantitative benchmarks are essential for validation and comparison of interfacial solvers. We present here three benchmarks results that were used for validation of the original and new variant of the isoAdvector method. Comparisons are made with reference data and OpenFOAM® VoF original solver, interFoam, employing the MULES limiter. The first case is the static bubble under zero gravity. The RDF reconstruction method demonstrates better prediction of the interface curvature, pressure jump between the phases and strong reduction of the spurious currents. The second and third validation cases are single rising bubbles in a quiescent liquid, with a spiraling path for the third case. Either on hexahedral or tetrahedral grids, the RDF reconstruction method demonstrates a better behavior.

*Corresponding author
Email address: `Lionel.Gamet@ifpen.fr` (Lionel Gamet)
URL: `www.ifpenergiesnouvelles.fr` (Lionel Gamet)

Keywords: CFD, Multiphase flows, Bubbly flows, Incompressible, MULES, isoAdvector, Volume of Fluid, OpenFOAM, Spurious currents

1. Introduction

Gas-liquid interfacial flow systems are often encountered in a wide variety of configurations, in science, engineering or industry. In ocean engineering, floating wind turbines, oil and gas platforms, or more generally coastal and offshore structures are subject to violent waves, which is a paramount concern for their correct dimensioning. In the chemical process industry, various scales of gas-liquid flows are encountered ranging from large bubble columns, plate columns, agitated vessels, surface aerators, jets, static mixers or micro-reactors. Their applications are generally reactive flow systems, mixing, stripping or saturation systems. Complex phenomena of turbulent hydrodynamics involving breakup and coalescence, coupled with gas-liquid mass and heat transfers, appear in such systems. Many other applications of interfacial flows can be found such as inkjet printing, automotive applications (liquid films, fuel injection, aquaplaning, etc), ship maneuvering, tank sloshing, hydraulic pumps, metal casting, fire sprinklers, atomizers and aerosols, etc. The list of domains that may benefit from improved solution methods to the interface advection problem is thus extremely long.

Thanks to the increase of computing resources, highly resolved simulations gain more and more interest as a tool for detailed analysis of the physics of such multiphase flows. Many numerical methods have emerged to attempt to simulate gas-liquid flows. Among these, implicit interface capturing approaches like volume of fluid (VoF) [1], level set (LS) [2] or phase-field [3, 4] have proven to be efficient in simulating multiphase flows. Many different flow solvers, both commercial and open source, implement the VoF approach. Among them, the open source library OpenFOAM® [5] is one of the major actors. This library and its numerous solvers and utilities is widely used both in the industry and in academia.

Roenby et al. [6] have recently developed a new geometric VoF method, called

isoAdvector, for advecting the interface between two incompressible fluids. This method was included in the official release from OpenFOAM-v1706 in the new solver `interIsoFoam`. More recently, Scheufler and Roenby [7] introduced a novel computational interface reconstruction scheme based on the calculation of a reconstructed distance function (RDF). This new scheme has been combined with the interface advection step of the isoAdvector algorithm [7]. Second order convergence with reduced absolute errors is obtained for simple test cases on all mesh types. The computational cost of both isoAdvector variants remains lower [6, 7] than the standard OpenFOAM®'s algebraic interfacial flow solver, called `interFoam`, which was originally developed by Weller [8]. Numerical methods are detailed in section 2.

Before running into complex physics and geometries for gas-liquid flows, elementary quantitative benchmark configurations are essential for validation and comparison of interfacial flow solvers. Although isoAdvector gives a sharper interface than `interFoam` [6, 7], validation data for this new method are still sparse, especially for surface tension dominated flows. The objective of this paper is thus to perform quantitative comparisons of the isoAdvector solvers against benchmark data.

The first benchmark case we consider is the classical static bubble [9, 10]. This case allows to quantify the curvature error and the spurious currents generated by VoF solvers. Results are presented in section 3.

Hysing et al. [11] have proposed a 2D benchmark with a single rising bubble in a quiescent liquid. Two different cases are described in [11], corresponding to different density, viscosity and surface tension ratios. Many authors have used this benchmark for the validation of their solvers. We cite here only some recent works. For level-set (LS) based methods, we can cite for instance the work of Zuzio and Estivalezes [12]. The benchmark of Hysing has also been used to validate coupled LS with VoF solvers [13, 14]. Patel et al. [15] have used the same benchmark for validation of a novel algorithm combining VoF with a staggered/non-staggered method. Manik et al. [16] have applied that benchmark to validate an algebraic VoF algorithm. Febres and Legendre [17] have

validated a new front-tracking algorithm implemented in the JADIM code with the same benchmark [17]. Recently, Castello Branco et al. [18] have discussed 2D results on the same benchmark with an improved curvature computation in isoAdvector. Adelsberger et al. [19] have published a 3D equivalent of the same benchmark, except that lateral no-slip walls have been used in their work, instead of slip walls. For both 2D and 3D benchmarks [11, 19], result data were made available online by the authors at the URLs mentioned in the bibliography. In this paper, we consider only the case 2, where fluid physical parameters are more challenging to predict and closer to real applications. Results are discussed in section 4.

Finally, a freely deforming bubble rising benchmark in a still liquid is studied. We consider the bubble number 26 taken from Cano-Lozano [20]. The path of this bubble is helicoidal, describing a spiraling path. This benchmark assesses the ability of the solvers to capture the transition towards path instability in a three-dimensional context. Results are discussed in section 5.

Validations of the isoAdvector methods are performed against the historical OpenFOAM® VoF solver `interFoam`, against the open source solver Basilisk [21] and against literature results concerning the benchmark of Hysing. Table 1 list the solvers that were operated in the present study.

2. Numerical methods

We consider here an unsteady, laminar, isothermal and incompressible two-phase flow. The flow is supposed without mass transfer across the gas-liquid interface. The governing equations are the continuity and momentum equations. The incompressible mass conservation reads:

$$\nabla \cdot \mathbf{u} = 0 \tag{1}$$

where $\mathbf{u}$ denotes the velocity vector. The Navier-Stokes equations for the momentum evolution are written as:

$$\rho \frac{\partial \mathbf{u}}{\partial t} + \rho \nabla \cdot (\mathbf{u}\,\mathbf{u}) = \nabla \cdot \big(\mu(\nabla \mathbf{u} + \nabla^T \mathbf{u})\big) - \nabla p + \rho \mathbf{g} + \mathbf{F}_\sigma \tag{2}$$

where p, ρ, μ are respectively the pressure, density and viscosity. $\mathbf{g}$ is the gravity. $\mathbf{F}_\sigma$ represents the surface tension force, which is expressed as a source term per unit of volume in the momentum equation.

In the VoF method, the volume fraction field α is introduced, which represents for each cell the fraction of its volume which is occupied by one of the two fluids. Quantities are then defined as volume fraction weighted sums. If α denotes the first phase volume fraction, ρ_1 the first phase density, ρ_2 the second phase density, then the density is defined by the mathematical average $\rho = \alpha\rho_1 + (1-\alpha)\rho_2$. Other quantities like viscosity can be defined similarly. The volume fraction α is obtained by solving the following transport equation:

$$\frac{\partial \alpha}{\partial t} + \nabla \cdot (\alpha\, \mathbf{u}) = 0 \tag{3}$$

Special care must be taken in the numerics to prevent smearing of the α-field and at the same time keeping it bounded ($0 \leq \alpha \leq 1$). In the `interFoam` solver, sharpness is obtained by adding an artificial interface compression term $\nabla \cdot (\alpha(1-\alpha)\, \mathbf{u}_c)$ to the equation 3 (see Weller [8]). Boundedness is ensured by employing the MULES limiter (Multidimensional Universal Limiter with Explicit Solution). More details can be found in Deshpande [22]. In the following, the `interFoam` solver will be used as a reference solver for comparisons and will for the sake of brevity be denoted MULES.

The solver `interFoam` has been widely used and validated [23, 24, 25, 26], but under some conditions the described method may fail in keeping the interface sufficiently sharp. Furthermore, the heuristic nature of the added compression term can lead to inaccurate interface advection and undesirable features such as unphysical ripples on the interface [27, 28]. This motivated the development of the isoAdvector geometric VoF method, which was first presented by Roenby et al. [6]. In the latter reference, isoAdvector was tested with a variety of pure advection cases yielding very good results in terms of volume conservation, interface sharpness, boundedness and shape preservation. The isoAdvector method implements new ideas in both the interface reconstruction step and the interface advection step. The reconstruction step uses efficient isosurface calculations to

compute the distribution of fluids in a grid cell. The interface advection step uses a novel division of a physical time step into sub-time steps on which the volume fraction flux through a cell face can be calculated analytically under the assumption that the interface is moving steadily across the face during the interval. In the development of this procedure, no assumptions are made on the shape of a cell face, which makes the advection step applicable on arbitrary meshes. Except for the interface advection step, the `interIsoFoam` (isoAdvector) solver is identical to the `interFoam` (MULES) solver. They both solve the governing system of equations in a segregated manner using the PIMPLE algorithm (a combination of the SIMPLE and PISO algorithms) for pressure-velocity coupling. Strictly speaking, isoAdvector and MULES also differ in the way `rhoPhi` (used in the momentum convection term) is calculated, which is described in [28].

With recent improvements, the isoAdvector method has been made consistently second order for all mesh types (See Scheufler and Roenby [7]). Scheufler and Roenby [7] have presented an iterative residual based interface reconstruction procedure utilizing a reconstructed distance function (RDF) to estimate the local interface position and orientation from the raw volume fraction data. This new algorithm has been developed in two variants based on RDF isosurface reconstruction and on piecewise linear interface construction (PLIC), respectively. The latter reconstruction method, called plic-RDF, has been used in the present work. Following the nomenclature in Scheufler and Roenby [7], we will refer to the original reconstruction algorithm of Roenby [6] as *isoAdvector isoAlpha* or simply *isoAlpha*. In the present paper, we have used the reconstruction method plic-RDF with 5 iterations from [7]. Corresponding results will be referred to as *isoAdvector plicRDF-5* or simply *plicRDF-5*. Additionally, the RDF of plicRDF-5 method is used to compute curvature [29] instead of using the volume fraction gradient.

In the next sections, comparisons between codes other than the OpenFOAM® flow solvers are exposed, either with the literature [11] or with the Basilisk code (`http://basilisk.fr`). In the Basilisk code, equations 1 and 2 are solved us-

ing the approach implemented by Popinet [21]. The corresponding finite volume spatial discretization makes use of a graded quadtree or octree partitioning depending on the number of dimensions of the problem studied. All variables are collocated at the cell centers. Time advancement of the viscous term in the momentum equation is achieved with an implicit scheme, while the advection equation is solved using the Bell-Colella-Glaz scheme [30]. A piecewise-linear geometrical VoF method is used to solve the advection equation of the volume fraction [9]. Besides, a combination of the height-function curvature estimation and a well balanced scheme for the estimation of the surface tension force is also used [9, 31].

The table 1 summarizes the different solvers used in the present study.

Table 1: List of VoF solvers used in this benchmark study.

VoF solver	**Type**	**References**
MULES	Algebraic VoF	[8, 22]
isoAdvector isoAlpha	Geometric VoF	[27, 28]
isoAdvector plicRDF-5	Geometric VoF	[7]
Basilisk	Geometric VoF with height-function	[9, 21]

For all OpenFOAM® computations, second order schemes were chosen. The Crank-Nicolson second order time scheme with blending coefficient 0.9 has been used. The Gauss linear scheme was used for gradient terms on hexahedral grids, while a least squares gradient scheme (namely `pointCellsLeastSquares`) was used for prismatic or tetrahedral meshes. Laplace operators were discretized with the Gauss linear corrected scheme. The convective term in the momentum equation was treated with a limited Gauss linear scheme specialised for vector fields (`Gauss limitedlinearV 1`). This scheme reduces to an upwind scheme in regions of strong velocity gradient. A single limiter is applied to all components of the vector simultaneously. A Gauss Van Leer scheme was used for the α convection term. In the MULES simulations, the artificial interface compres-

sion term is discretized with the `Gauss interfaceCompression` method that ensures the boundedness of the volume fraction. The reader is referred to the OpenFOAM® user's guide [5] for more details on numerical schemes.

The generalised Geometric-Algebraic Multi-Grid (GAMG) linear solver [5] was used for pressure terms, while the smooth solver was used for the velocity. Constant time steps have been used with OpenFOAM® computations. Small time steps combined with a Crank-Nicolson second order scheme ensure that discretization errors due to time scheme are kept at a very low level. The PIMPLE algorithm [5] was run with 3 PISO correctors (`nCorrectors` set to 3), which means that the pressure field is corrected three times per PISO corrector loop. The overall PIMPLE algorithm was repeated 3 times per timestep (`nOuterCorrectors` set to 3), which means that the calculation of the pressure-momentum coupling was iterated 3 times in a single time step. Setting `momentumPredictor` to `true` was necessary for accuracy with isoAdvector. The `momentumPredictor` is a switch enabling activation/deactivation of the predictor step in the PISO algorithm. The number of non-orthogonal correctors (`nNonOrthogonalCorrectors`) was set to 1 on tetrahedral/prismatic grids and 0 on hexahedral grids.

The initialisation of bubble shapes in the different configurations examined in the next sections has been done using the `setAlphaField` utility. This tool guarantees an accurate initial value for the volume fraction field that respects the percentage of each phase present in a cell crossed by an analytical interface, like for example a cylinder in 2D or a sphere in 3D.

3. The static bubble case

3.1. Definition of test case

The first case we consider is the static bubble [9] under zero gravity. As illustrated on figure 1 in 2D, a cylindrical interface of gas (fluid 2) is initialized inside a continuous liquid phase (fluid 1). Both fluids have the same viscosity μ and density ρ. The density ρ and the surface tension σ between the two fluids

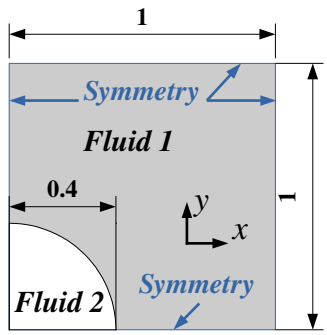

Figure 1: Configuration and boundary conditions for 2D static bubble case under zero gravity.

are both taken as unitary constant values. The bubble has an initial diameter $D_0 = 2R_0 = 0.8\,\mathrm{m}$. Only a quarter of the 2D geometry is simulated in a domain of size $1 \times 1\,\mathrm{m}$. The bubble is placed in the bottom left corner of the domain. All boundary conditions are symmetries. This test case can easily be extended to 3D, where 1/8th of an initially spherical bubble is then simulated.

This test case is ideally not supposed to generate any velocity field and the pressure field should follow the Laplace pressure jump at the interface. In such a static configuration, the Navier-Stokes momentum equations 2 simplify to a balance between pressure gradient and surface tension force. Spurious currents can occur from a numerical imbalance between the discretization errors of those two terms. This numerical imbalance creates a source term in the vorticity production equation, which in turn generates velocities. In our attempt to quantify the spurious currents generated by the OpenFOAM® VoF methods studied here, this elementary case provides a fundamental base of comparison. Besides, the static bubble test case has been widely used in the literature [32, 9, 21, 10] and is mentioned as a reference test case for Laplace equilibrium and spurious currents in the review of Popinet [31].

Following previous references on the static bubble [9, 10], we studied the influence of the grid size. For 2D square (resp. 3D cubic) grids, the number of grid cells along x and y (resp. x, y and z) directions are identical. In 2D, triangular grids have also been considered with the OpenFOAM® solvers. A standard Delaunay algorithm was then used to generate grid cells of mean edge

lengths identical to that of the square grids. The grids were created with the software Pointwise®.

The Laplace number defined as $La = \dfrac{\rho\, D_0\, \sigma}{\mu^2}$ was also varied during this study. As the density ρ and the surface tension σ are constant, an increase of the Laplace number thus corresponds to a decrease of the viscosity.

Different time and velocity characteristic scales can be defined for the static bubble [9, 10]. We have chosen to follow Abadie [10] by defining a capillary time scale as $t_\sigma = \sqrt{\rho\, D_0^3/\sigma}$ and a capillary-viscous velocity scale as $u_\sigma = \sigma/\mu$. These values will be used in the following to present dimensionless data.

VoF methods have the advantage to be mass conservative. This property was firstly verified in the OpenFOAM® computations of the static bubble, where mass conservation relative errors were found inferior to $10^{-5}\%$. This point will thus not be developed in the rest of the article. Next sections quantify curvature and spurious currents errors.

3.2. Pressure jump and curvature error

The Young-Laplace law for the pressure discontinuity due to surface tension can be expressed as:

$$\Delta p_{\mathrm{th}} = \sigma\, \kappa_{\mathrm{th}} = \sigma \left(\frac{1}{R_1} + \frac{1}{R_2} \right) \tag{4}$$

where κ_{th} is the curvature [32]. R_1 and R_2 are curvature radii along two perpendicular directions. For the 2D cylindrical bubble which has infinite length in the z direction, one of the curvature radii can be taken as infinite. The second curvature radius in the perpendicular direction is the bubble radius. The theoretical curvature is thus $\kappa_{\mathrm{th}} = 2.5\,\mathrm{m}^{-1}$ and the pressure jump across the 2D bubble is $\Delta p_{\mathrm{th}} = 2.5\,\mathrm{Pa}$. In the case of the 3D spherical bubble, R_1 and R_2 are equal and the exact pressure jump is $\Delta p_{\mathrm{th}} = 5\,\mathrm{Pa}$.

Example pressure profiles along two sampling directions across a 3D static bubble are shown on figure 2, for the OpenFOAM® solvers. It is clearly visible that plicRDF-5 provides the best fit to the theoretical rectangular function of pressure jump. MULES and isoAlpha solvers underestimate the pressure jump.

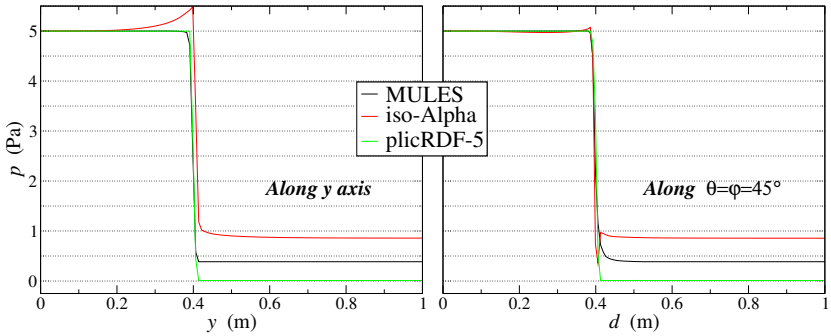

Figure 2: Pressure jump across the static bubble at time $t_* = 100$. Left: pressure along y axis, right: pressure along the domain diagonal at spherical coordinate angles $\theta = \phi = 45^o$. 3D configuration with 128^3 cubic grid cells, $La = 12000$, OpenFOAM® solvers.

isoAlpha presents over or undershoot near the pressure discontinuity at the interface, while MULES smears out the pressure jump.

In a well-balanced solver, this difference between the exact and numerical pressure jumps arises only from errors in the estimation of the curvature. This assumption can easily be verified by imposing the exact curvature κ_{th} inside the solvers source code, as a substitute for the computed curvature. 2D results are shown on figure 3, in terms of time-averaged maximum capillary number (see section 3.3 for capillary number definition and for definition of time averaging), for different grid densities and versus Laplace number. Same trends are observed in 3D. When exact curvature is used, maximum capillary numbers become of the order of numerical precision, slightly vary with grid size and decrease when Laplace number increases. Data for the three solvers are superimposed, which is an expected result as all three solvers use the same pressure-velocity coupling algorithm.

Pressure jump errors, as can be seen in the sample curve of figure 2, can be characterized by comparing the maximum of pressure to the theoretical pressure jump. This was translated to a pressure relative error norm written as:

$$Lp_\infty = \frac{|p_{\max} - \Delta p_{\text{th}}|}{\Delta p_{\text{th}}} \tag{5}$$

with $p_{\max}$ computed as the maximum of pressure field over all the computational domain and averaged over the last third of the computation time (see

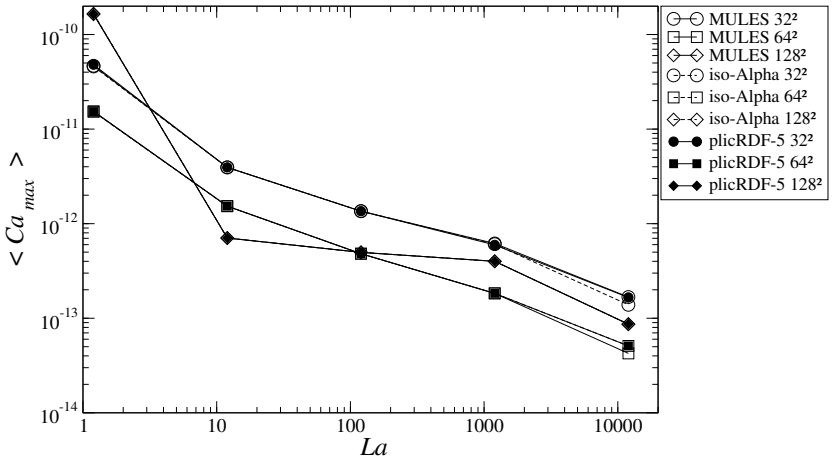

Figure 3: Static bubble on 2D square grids: Mean maximum capillary number versus Laplace number when curvature is imposed to exact curvature. Comparison of OpenFOAM® solvers.

section 3.3 for averaging details). This pressure relative error norm is shown on figure 4, versus Laplace number on the top, and versus density on the bottom, for three 2D grid sizes. We notice the lower errors of plicRDF-5, over MULES and isoAlpha. Note also that changing the density of the fluids has no influence on pressure jump, and more generally on generated spurious currents.

In order to quantify the curvature error, the following curvature relative error norms are introduced:

$$\begin{aligned} L_2 : \qquad & \frac{1}{\kappa_{\text{th}}} \sqrt{\frac{1}{N} \sum_{i=1,N} \left(\kappa_i - \kappa_{\text{th}}\right)^2} \\ L_\infty : \qquad & \frac{1}{\kappa_{\text{th}}} \max_{i=1,N} \left|\kappa_i - \kappa_{\text{th}}\right| \end{aligned} \tag{6}$$

where N is the number of elements in a series of curvature data κ_i.

The evolution of curvature L_2 and L_∞ relative error norms is plotted versus grid size on figure 5 for the three OpenFOAM® solvers and at $La = 1200$. The curvature was extracted at the interface and the resulting data array was processed through the error norms of equations 6. We note that the curvature error is increasing with grid refinement, however it does so to a lesser extent for the plicRDF-5 solver L_2 norm. The RDF curvature method (plicRDF-5 curves) generates errors that are around two orders of magnitude smaller

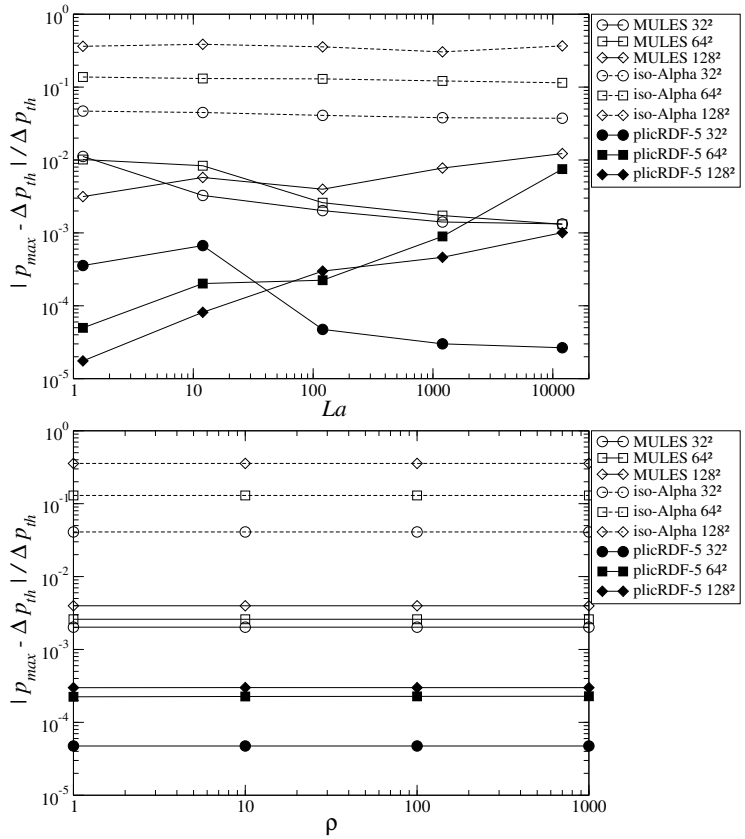

Figure 4: Pressure relative error norms versus Laplace number (top, unitary density) and versus fluid density (bottom, $La = 120$) for the 2D static bubble and for all OpenFOAM® solvers.

than the curvature based on volume fraction gradient used by MULES and isoAlpha. On triangular grids, curvature errors are higher for all solvers. The difference between plicRDF-5 solver and the other methods becomes inferior than on square grids.

3.3. Spurious currents quantification

Figure 6 shows a qualitative view of the spurious currents generated by the different OpenFOAM® solvers used here. Note that the scaling factors applied to make velocity vectors visible are in the same order as the ratios of curvature relative errors between the solvers, as can be seen from figure 5.

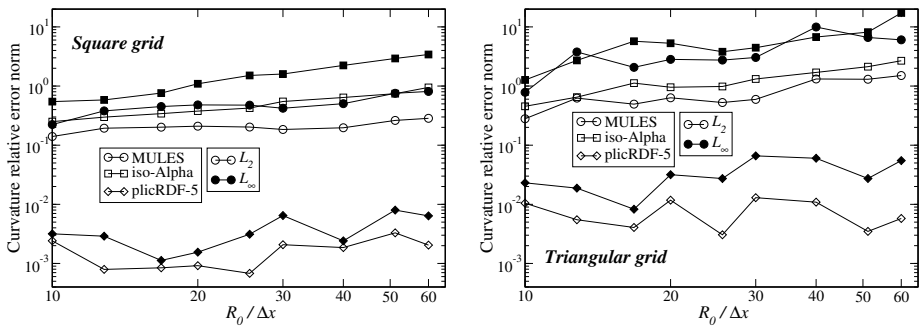

Figure 5: Curvature relative error norms at the interface versus grid size in 2D for the static bubble at time $t_* = 100$. Left: square grids, Right: triangular. Configuration at $La = 1200$, OpenFOAM® solvers.

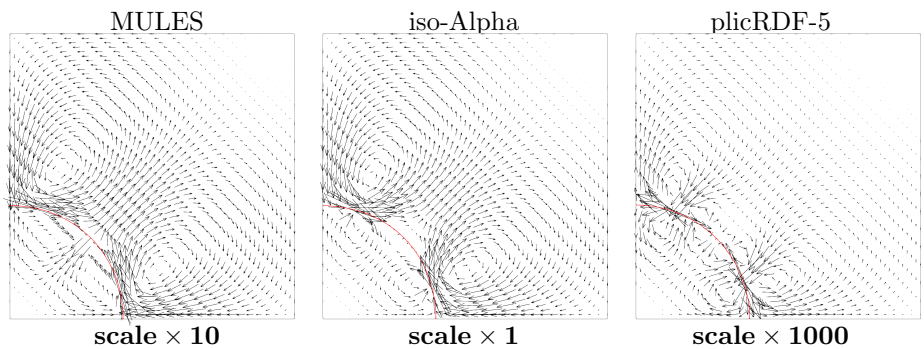

Figure 6: Static bubble in 2D: Visualization of spurious currents at time $t_* = 100$ on 64×64 square grid and for $La = 120$. Vectors on the r.h.s. plicRDF-5 image are scaled by a factor 1000 with respect to isoAlpha. The red line materializes the isoline $\alpha = 0.5$.

In order to quantify spurious velocities, we monitor the maximum of velocity magnitude $U_{\max}$ over the computational domain. $U_{\max}$ is recorded at each time step. Spurious currents magnitude can then be made dimensionless in terms of a capillary number as $Ca_{\max} = (\mu U_{\max})/\sigma = U_{\max}/u_\sigma$. Time is normalized by scaling with t_σ. The resulting maximum capillary number is plotted on figures 7 and 8, for, respectively, 2D and 3D hexahedral grids and for different Laplace numbers. The noticeable point is that plicRDF-5 generates spurious vectors that are two orders of magnitude smaller than the other OpenFOAM® solvers. Basilisk results were obtained without adaptive mesh refinement. Except in the 3D case at $La = 120$ which would require a finer mesh, Basilisk maximum

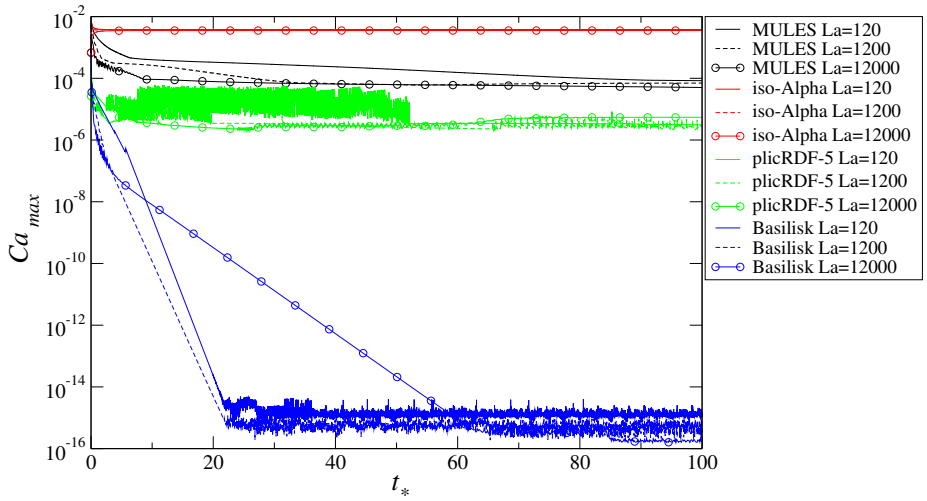

Figure 7: Static bubble in 2D: Maximum capillary number over the computational domain for a hexahedral grid of size 32^2 versus nondimensional time, for different Laplace numbers.

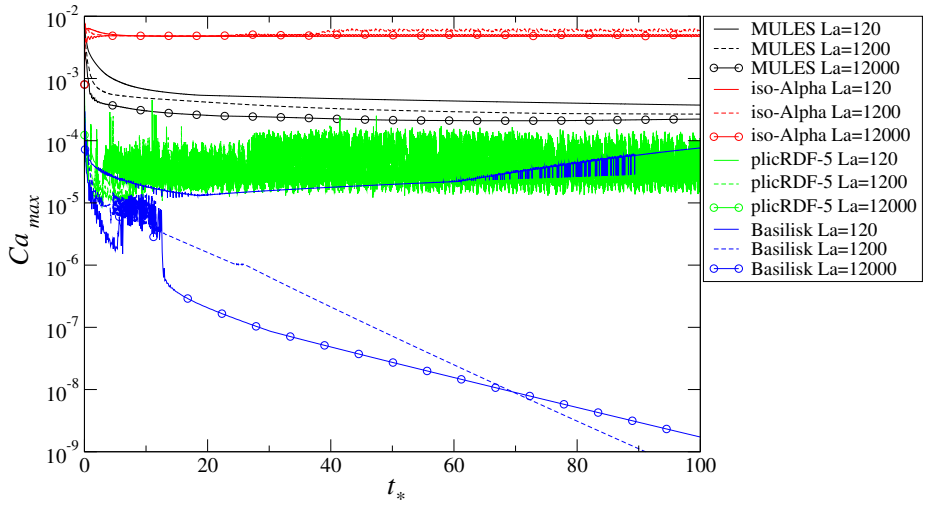

Figure 8: Static bubble in 3D: Maximum capillary number over the computational domain for a hexahedral grid of size 32^3 versus nondimensional time, for different Laplace numbers.

capillary numbers converge to machine precision, which suggests that in that case the measured velocities cannot be considered as actual parasitic currents but rather as the result of a convergence process from an initial solution to a numerical steady state. The imposed initial solution, which does not guarantee the exact balance between the surface tension and pressure gradient, generates

capillary waves at the interface, which are then damped by viscosity towards the exact solution [9]. The Basilik convergence is slower in 3D than in 2D. OpenFOAM® spurious velocities remain rather constant and do not converge to machine precision, even for the long simulated times (almost double compared to Abadie [10]). This means that the OpenFOAM® parasitic currents are continuously fed by an imbalance between pressure gradient and surface tension force.

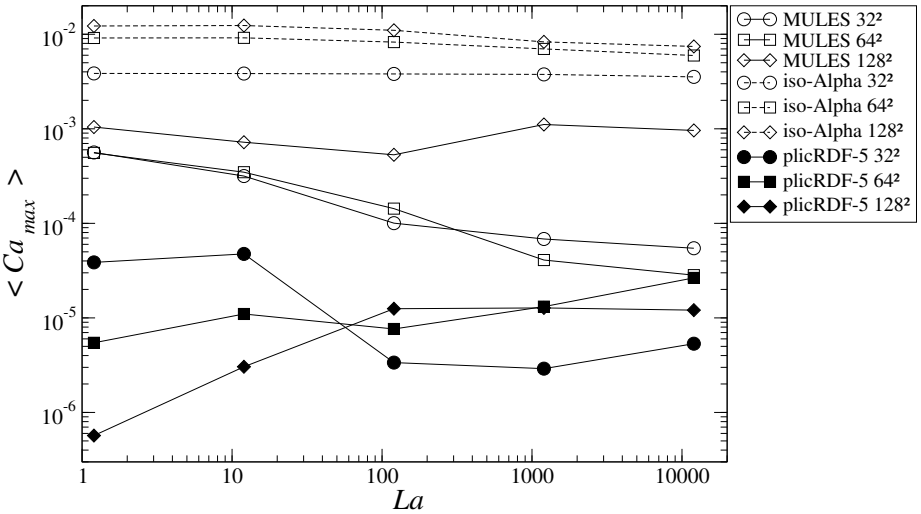

Figure 9: Static bubble in 2D: Mean maximum capillary number versus Laplace number. Comparison of OpenFOAM® solvers.

The effect of Laplace number is shown in figures 9 and 10 for respectively 2D and 3D static bubbles. The time average $< Ca_{\max} >$ of spurious current intensity has been computed over the last third of the time interval of the computations, i.e. between $t_* = 66.66$ and $t_* = 100$. For both 2D and 3D cases, the plicRDF-5 solver produces parasitic currents which are orders of magnitude lower than the other solvers. A general trend of slowly decreasing $< Ca_{\max} >$ with Laplace is observed, particularly in 3D. However, this is not true for plicRDF-5 data in 2D.

The effect of grid size for $La = 1200$ is shown in figure 11 for 2D grids. We can again confirm that the plicRDF-5 solver produces orders of magnitude

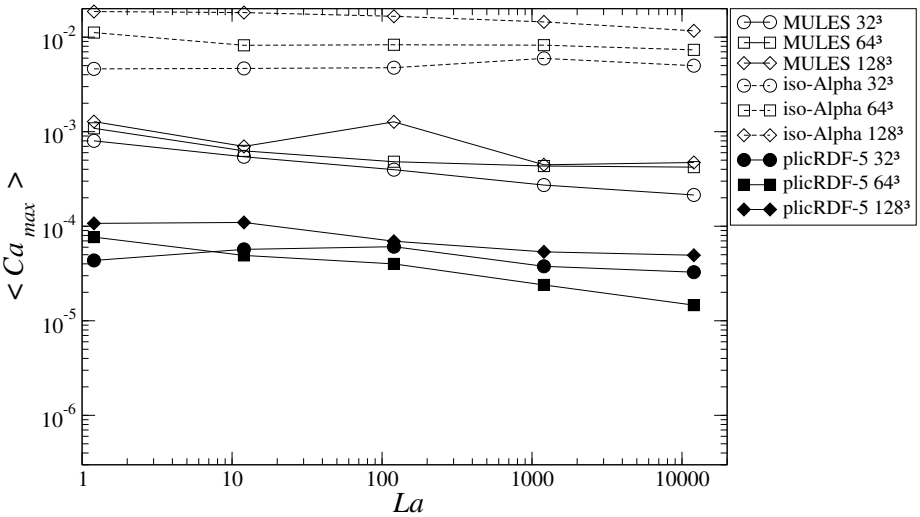

Figure 10: Static bubble in 3D: Mean maximum capillary number versus Laplace number. Comparison of OpenFOAM® solvers.

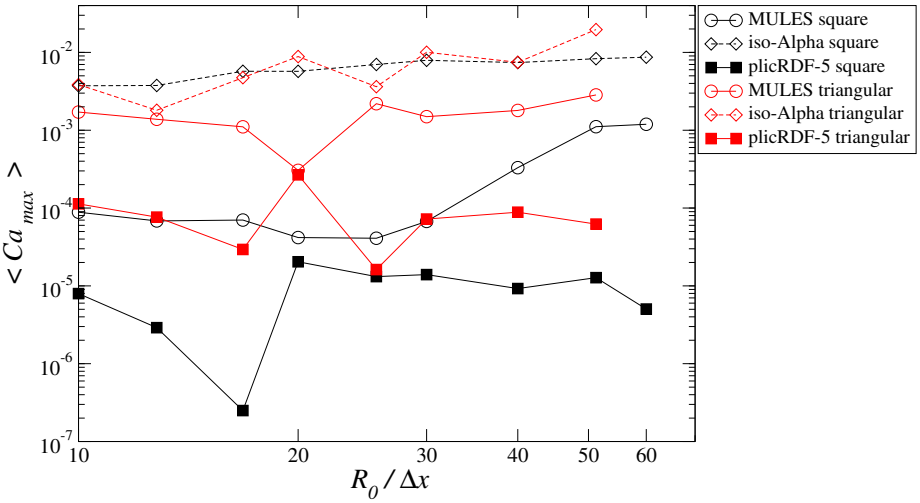

Figure 11: Static bubble in 2D: Mean maximum capillary number versus grid size at $La = 1200$. Comparison of OpenFOAM® solvers on both square and triangle grids.

lower intensities than other solvers. Parasitic currents are around one order of magnitude larger on triangular grids compared to square grids, except for the isoAlpha solver which gives identical levels for both types of grids.

Spurious velocities can have an influence on the shape of the bubble. In the present case with zero gravity, the bubble should remain perfectly cylindrical (or

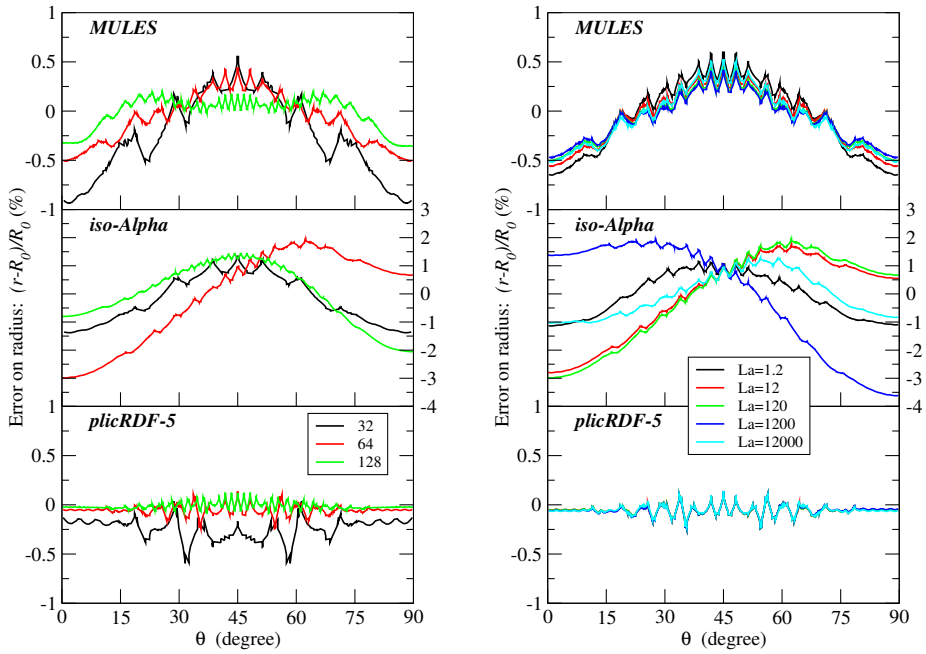

Figure 12: Static bubble in 2D: Error on bubble radius versus angular position for square grids. Left: Effect of grid size at $La = 120$. Right: Effect of Laplace number for 64×64 cells.

spherical in 3D). The graph of figure 12 shows the relative error of the radius in percent on 2D square grids of increasing sizes (left) or when the Laplace number is varied (right). R_0 denotes the bubble radius at time $t = 0$. MULES and isoAlpha show maximal errors along the coordinate axes directions (at 0 and 90 degrees) and also along median directions at 45 degrees. plicRDF-5 errors are smaller and are rather independent of coordinate directions. When the Laplace number is varied, MULES and plicRDF-5 show identical errors, while isoAlpha maximal errors oscillate along axial directions.

4. Single rising bubble benchmark

4.1. Definition of test case

The test case number 2 as described by Hysing et al. [11] has been used here. We have used only this second test case as it was judged more representative of final industrial applications. The 2D case setup is schematized in figure 13.

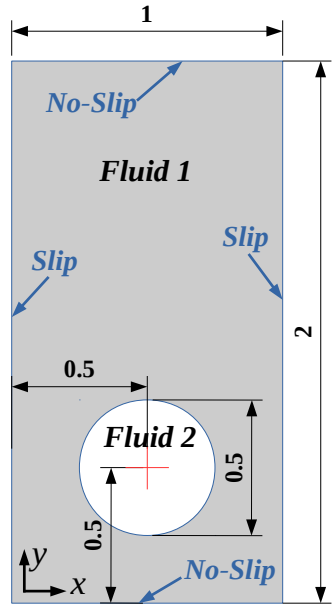

Figure 13: Configuration and boundary conditions for 2D bubble benchmark.

The computational domain of size $1 \times 2\,\mathrm{m}$ is in the plane xy. First phase (liquid) properties are $\rho_1 = 1000\,\mathrm{kg\,m^{-3}}$, $\mu_1 = 10\,\mathrm{kg\,m^{-1}\,s^{-1}}$, while second phase (gas) takes $\rho_2 = 1\,\mathrm{kg\,m^{-3}}$, $\mu_2 = 0.1\,\mathrm{kg\,m^{-1}\,s^{-1}}$, as physical parameters. The surface tension is $\sigma = 1.96\,\mathrm{kg\,s^{-2}}$. The density ratio is thus 1000 and the dynamic viscosity ratio is 100. Gravity is taken as $g = 0.98\,\mathrm{m\,s^{-2}}$ along the $-y$ direction. The bubble thus rises along the positive y direction. Lateral boundary conditions are slip walls, while top and bottom boundaries are no-slip walls. Extension to 3D of this benchmark case along the z direction is straightforward. The bubble is initialised as a cylinder in 2D (or a sphere in 3D) of diameter $D_0 = 0.5\,\mathrm{m}$.

The Bond/Eötvös number $Bo = \rho_1\, g\, D_0^2/\sigma = 125$ and the Galilei number $Ga = \rho_1\, g^{1/2}\, D_0^{3/2}/\mu_1 = 35$ classify the current bubble in the peripheral breakup regime, where inertial forces are dominant [33].

2D square (resp. 3D cubic) or 2D triangular (resp. 3D tetrahedral) grids of different sizes have been created for the simulations. Hexahedral uniform grids were generated with the structured grid generator `blockMesh` from the

OpenFOAM® distribution. Using the software Pointwise®, triangular and tetrahedral grids were generated by a standard Delaunay algorithm, by imposing mean edge lengths identical to that of the square grids of same resolution. For 2D cases, the cell sizes along the non-significant z direction were taken equal to the cell sizes along x or y. We note that OpenFOAM® grids are always three-dimensional. For 2D computations, six grid resolutions have been used, with 20, 40, 80, 160, 320 and 640 cells along the x direction. The number of cells in the y direction is the double of that in the x direction. For 3D computations, the number of cells is identical along x and z, and ranges only up to 320.

Constant time steps have been used, starting at $\Delta t = 0.002\,\mathrm{s}$ for the coarsest level, and reducing by a factor 2 at the next finer grid level. This method kept the maximum CFL number below 0.05 for all grids.

4.2. Post-processing

Post-processing quantities of interest are described in details in [11, 19]. These are the vertical position of the bubble centroid, the bubble rise velocity, the bubble circularity (in 2D) or sphericity (in 3D), area and volume. The volume of the bubble V_b is computed by a integral of the gas volume fraction over the entire domain Ω as:

$$V_b = \int_\Omega (1-\alpha)\, dv \tag{7}$$

The centroid of mass $\mathbf{x}_G$ is computed through:

$$\mathbf{x}_G = \frac{1}{V_b} \int_\Omega (1-\alpha)\mathbf{x}\, dv \tag{8}$$

where $\mathbf{x}$ represents the cell center coordinates. Similarly, the bubble velocity is calculated with:

$$\mathbf{U}_b = \frac{1}{V_b} \int_\Omega (1-\alpha)\mathbf{u}\, dv \tag{9}$$

where $\mathbf{u}$ denotes the velocity. In practice, the integrals are computed by summations of the cell-centered values of α, $\mathbf{x}$ and $\mathbf{u}$, while dv is taken as the cell volume.

Thanks to the calculation of a triangulated isosurface at $\alpha = 0.5$, the bubble area A_b can be determined by summation of the elementary triangles areas of this isosurface. The circularity $\mathcal{C}$ in 2D (resp. sphericity Φ in 3D) is then defined as the (resp. squared) ratio of the equivalent radius of the bubble based on its volume r_V over the equivalent radius of the bubble based on its surface r_A as:

$$\text{In 2D}: \ \mathcal{C} = \frac{r_V}{r_A} = \frac{\sqrt{V_b/(\pi \Delta z)}}{A_b/(2\pi \Delta z)} \qquad \text{In 3D}: \ \Phi = \left(\frac{r_V}{r_A}\right)^2 = \frac{(3V_b/(4\pi))^{2/3}}{A_b/(4\pi)} \tag{10}$$

where Δz denotes the size of the grid cells in the non-significant direction z for 2D calculations. $\mathcal{C}$ and Φ take the value 1 at the beginning of the computation and decrease as the bubble deforms.

4.3. Rising velocity

The figures 14 and 15 show results for the rise velocity in 2D for MULES, isoAdvector isoAlpha and plicRDF-5 on respectively square and triangular grids compared to the highest resolution reference data available in the literature and Basilisk. TP2D, FreeLIFE and MooNMD data are taken from the published results of Hysing [11]. The TP2D code (short for Transport Phenomena in 2D) is a Level Set solver treating immiscible fluids [2, 34]. The FreeLIFE (Free-Surface Library of Finite Element) code is an incompressible flow solver for solving free-surface two-fluid systems by a Level Set method [35]. MooNMD (Mathematics and object-oriented Numerics in MagDeburg) is a solver based on mapped finite-element methods that has been extended to two-phase flows with capillary forces by using the arbitrary Lagrangian-Eulerian approach[36, 37].

The objective of figures 14 and 15 is to discuss grid convergence of all three OpenFOAM® solvers with respect to the literature results (TP2D, FreeLIFE and MooNMD) or to the Basilisk solver. On square grids, MULES and plicRDF-5 results converge similarly to the literature solvers or to Basilik when increasing the number of cells. isoAlpha results converge to a slightly different solution since a lower rising velocity can be observed on the middle plots of figure 14. The figure 15 shows results on triangular grids. At higher grid resolutions, we

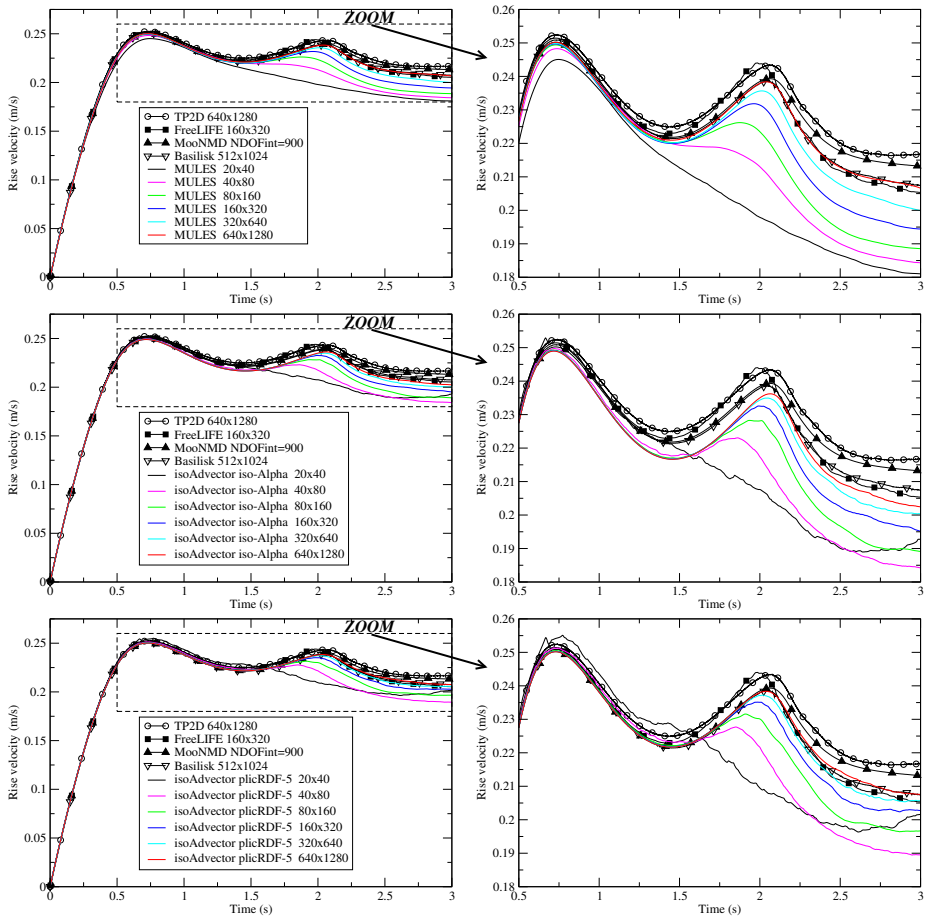

Figure 14: Time evolution of rise velocity on 2D hexahedral grids for solvers MULES (top), isoAdvector with isoAlpha (middle) and plicRDF-5 (bottom) reconstruction schemes.

clearly notice that plicRDF-5 correctly converge towards literature and Basilisk results, but neither MULES nor isoAlpha reach a grid convergence, showing overestimated bubble velocities (see red curve). This behavior can be explained by an increase of spurious currents at highest grid resolutions for these methods. The integral procedure to compute bubble velocity of equation 9 inherently accounts for spurious currents in the solution. As discussed in more details in section 3, parasitic currents are indeed increasing with grid refinement on triangular grids (see also figure 11).

In 3D on a hexahedral 160×320×160 grid (see figure 16), plicRDF-5 bubble

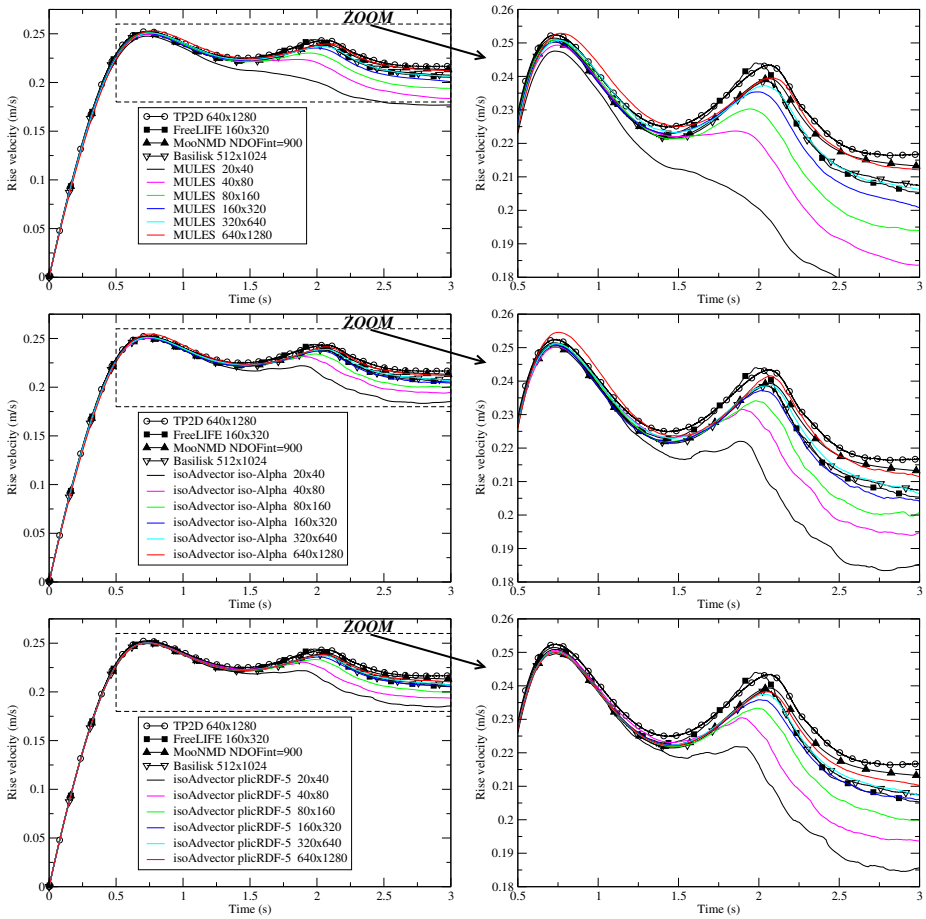

Figure 15: Time evolution of rise velocity on 2D triangular grids for solvers MULES (top), isoAdvector with isoAlpha (middle) and plicRDF5 (bottom) reconstruction schemes.

velocity results are the closest to the Basilisk reference. Literature data are not available with slip walls [19]. On tetrahedral grids, the velocities of all OpenFOAM® solvers are similar, but underestimate the rise velocity compared to their hexahedral equivalent.

4.4. Bubble shape

Figure 17 shows the bubble shape in 2D at time $t = 3$ obtained by reference solvers of the literature and by Basilisk. It can first be noted that each code gives a different solution, particularly in the chain of detached bubbles. As a

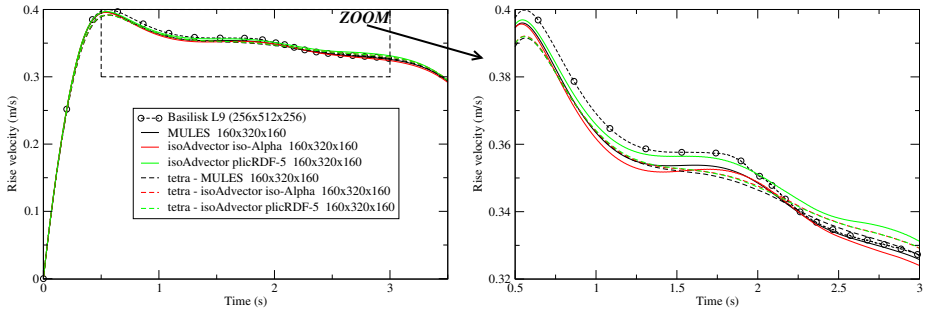

Figure 16: Time evolution of rise velocity on 3D hexahedral and tetrahedral grids of intermediate size. Comparison of Basilisk with MULES, isoAdvector with isoAlpha and plicRDF-5 reconstruction schemes.

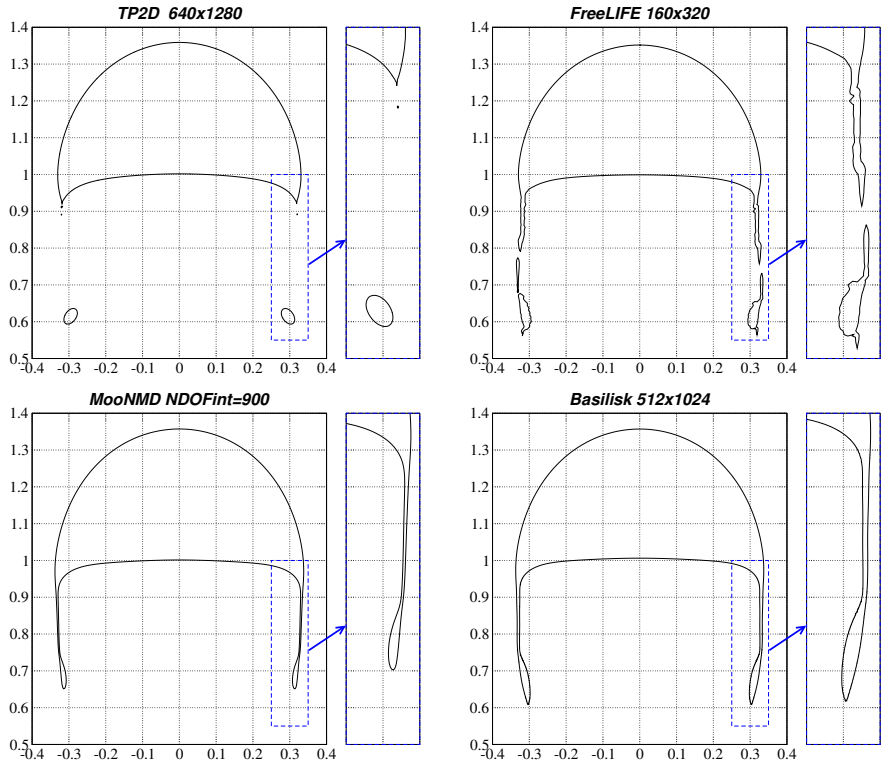

Figure 17: Single rising bubble shape in 2D at time t=3 for TP2D, FreeLIFE, MoonNMD and Basilisk solvers.

matter of fact, the test case number 2 of the benchmark of Hysing [11] is more challenging to simulate than test case 1. Figures 18, 19 and 20 compare the 2D bubble shapes for respectively MULES, isoAlpha and plicRDF-5, for the finest

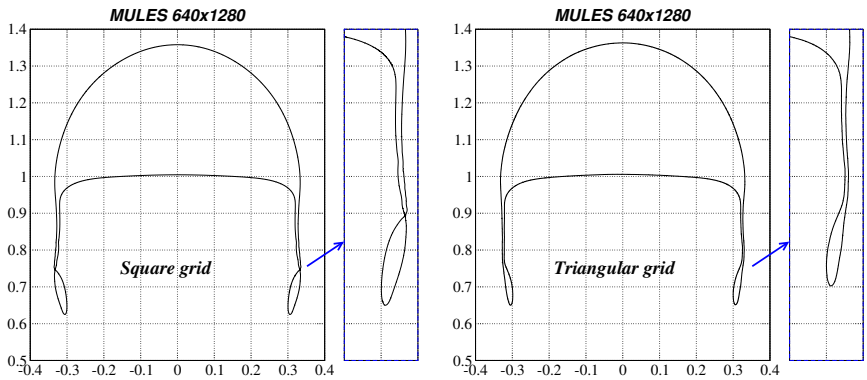

Figure 18: Single rising bubble shape in 2D at time t=3 for MULES. Left: square grids, Right: triangular.

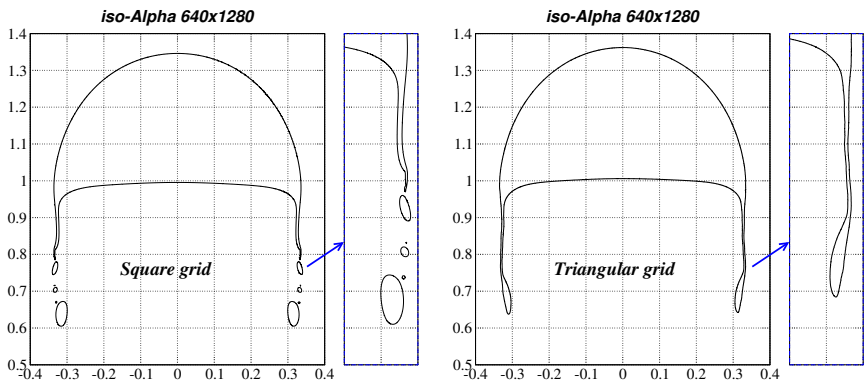

Figure 19: Single rising bubble shape in 2D at time t=3 for isoAdvector isoAlpha. Left: square grids, Right: triangular.

grid resolution, either on square or triangular meshes. All bubble shapes for all solvers are rather coherent in terms of global positions of the main leading and trailing fronts. For each solver of figures 18, 19 and 20, the trains of detached bubbles are different between square and triangular grids. The OpenFOAM® finest grids results are close to MooNMD and Basilik shapes, as plotted on figure 17.

For completeness about the bubble shape at time $t = 3$, we show on figures 21, 22 and 23 a comparison of the bubble shape in 2D on increasing size triangular grids with respect to the finest Cartesian grid shape made with the same solver. MULES and plicRDF-5 show a correct grid convergence. At the

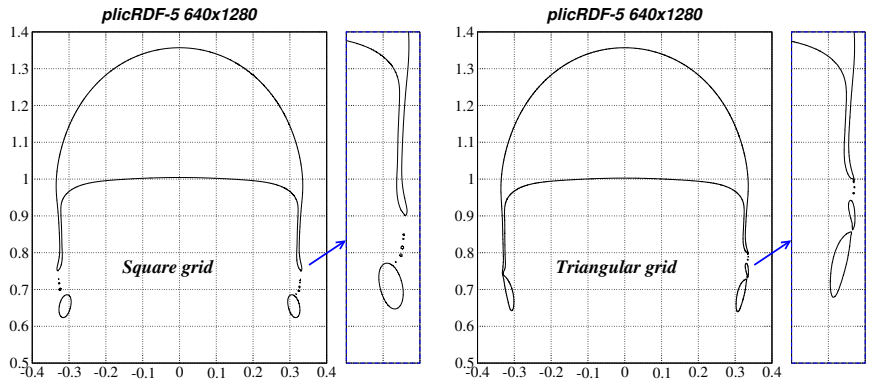

Figure 20: Single rising bubble shape in 2D at time t=3 for isoAdvector plicRDF-5. Left: square grids, Right: triangular.

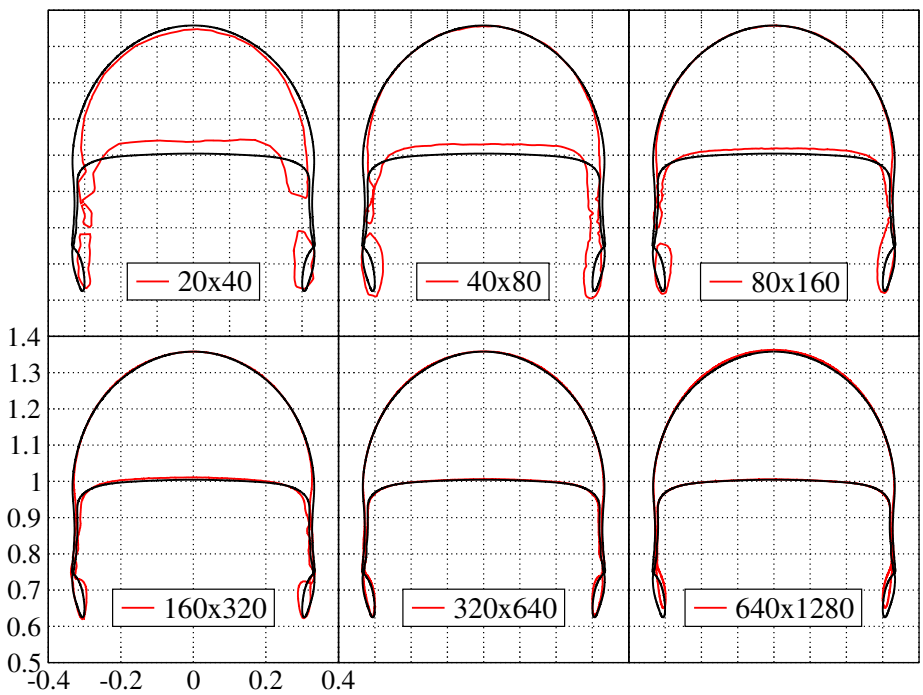

Figure 21: Comparison of 2D bubble shape obtained on different resolution triangular grids (red) against the finest Cartesian grid at 640×1280 (black). Plots for MULES at time $t = 3$.

finest levels (320×640 and 640×1280), square and triangular grids are almost identical, particularly in their main fronts. One should however notice a small discrepancy in the upper front for the MULES solver at 640×1280 (see bottom

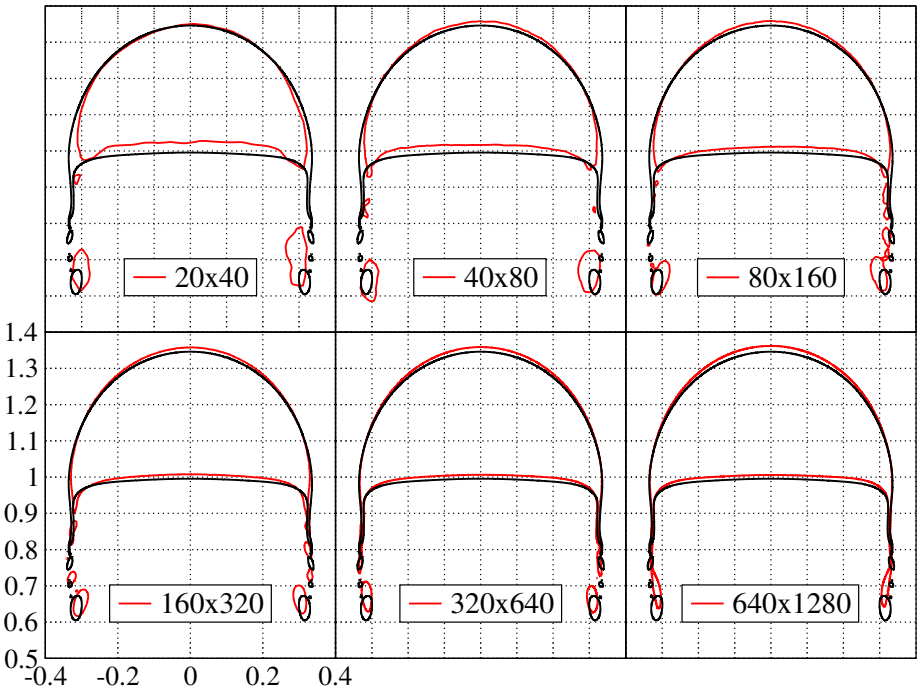

Figure 22: Comparison of 2D bubble shape obtained on different resolution triangular grids (red) against the finest Cartesian grid at 640×1280 (black). Plots for isoAdvector isoAlpha at time $t = 3$.

right of figure 21). For the isoAlpha solver, the difference between square and triangular grids remains for all levels, and even increases slightly at the highest resolution. This is correlated with the difference in bubble velocities observed on triangular grids as was noticed on figure 15.

In 3D, bubble shape results at the same grid resolutions as in figure 16 are shown on figure 24. On the hexahedral grid, plicRDF-5 bubble shape is the closest to the Basilisk reference. Due to a higher predicted bubble velocity, the Basilisk shape is slightly shifted upwards. All three OpenFOAM® solvers predict the same kind of bubble tail, shorter than the one obtained by Basilisk. MULES and isoAlpha predicted a small detachment on the hexahedral grid. On tetrahedral grids, the OpenFOAM® bubbles are shifted downwards at rather equivalent positions.

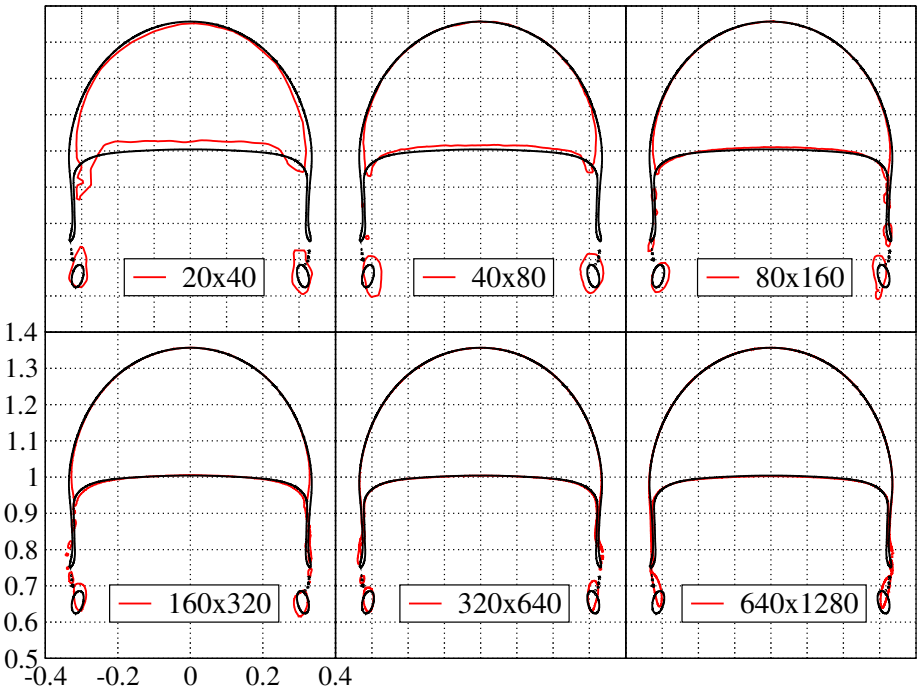

Figure 23: Comparison of 2D bubble shape obtained on different resolution triangular grids (red) against the finest Cartesian grid at 640×1280 (black). Plots for isoAdvector plicRDF-5 at time $t = 3$.

2D circularity is shown on figure 25 at grid resolution 160×320. OpenFOAM® solvers compare well to reference data, except with the TP2D solver which experiences a different bubble tail structure (see also top left plot in figure 17). The difference between square (continuous lines) and triangular (dashed lines) grids on MULES, isoAlpha or plicRDF-5 circularities is more visible for time $t > 2$ up to the end of the simulation. This is mainly due to the difference in the bubble tail. Basilisk data are not available for circularity or sphericity. In 3D, the figure 26 shows the equivalent results. Here again, more detached trailing small bubbles on Cartesian grids conduct to a lower sphericity.

It can be concluded that both in 2D and 3D, and both on hexahedral and triangular/tetrahedral grids, the plicRDF-5 has shown better ability to obtain bubble dynamics results that are more consistent and closer from the Basilisk

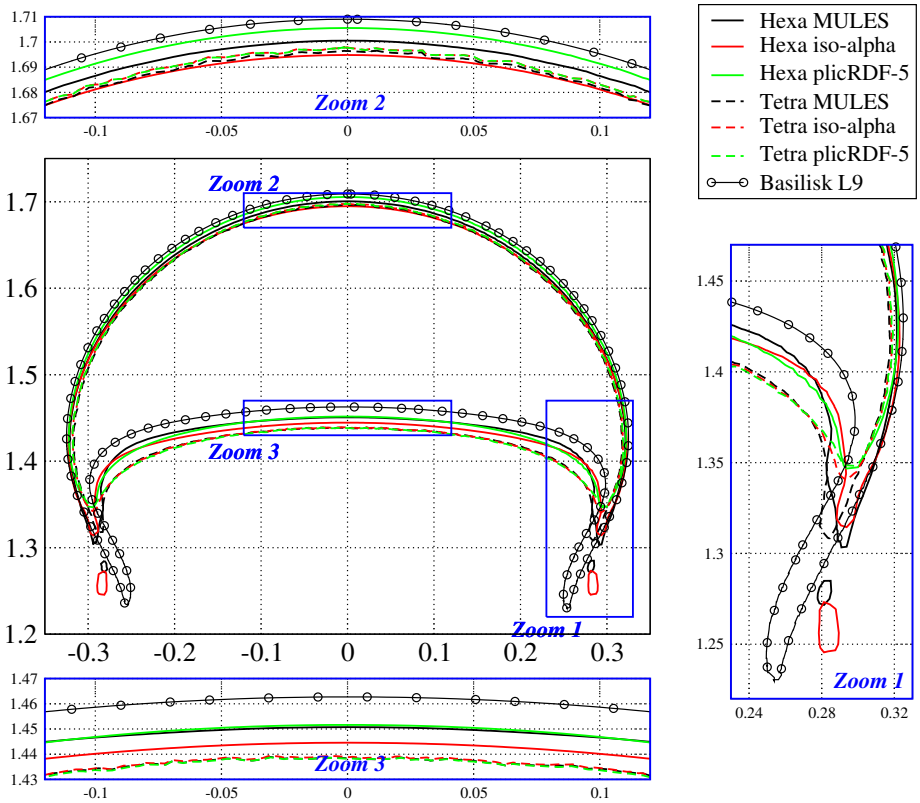

Figure 24: Comparison of 3D bubble shape obtained at time $t = 3$. Grid resolution is $160 \times 320 \times 160$ for OpenFOAM solvers and $256 \times 512 \times 256$ for Basilisk.

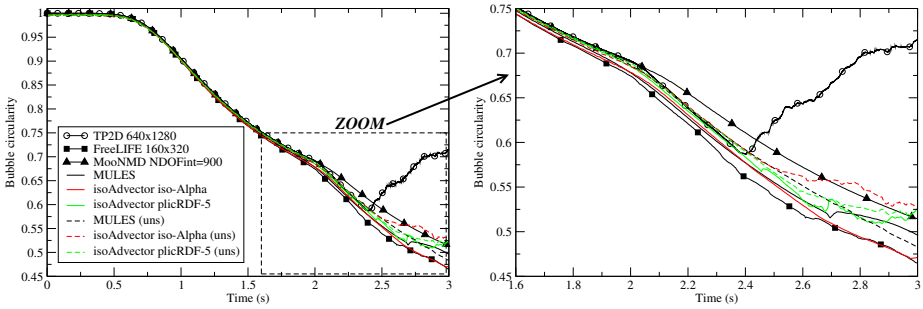

Figure 25: Bubble circularity at resolution 160×320. Comparison of MULES, isoAdvector isoAlpha and plicRDF-5 with reference data for both Cartesian and triangular grids.

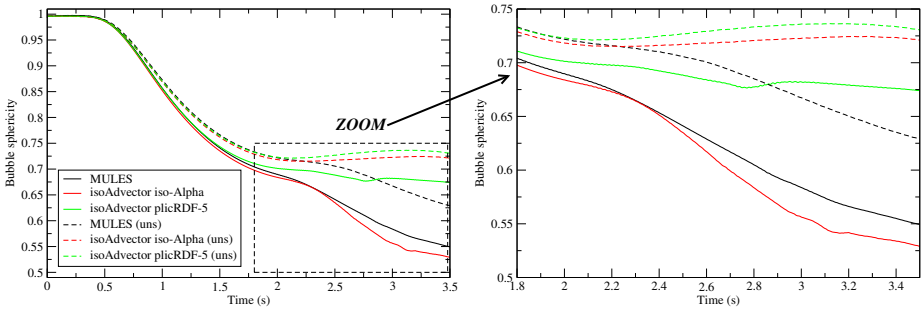

Figure 26: Bubble sphericity at resolution 160×320× 160. Comparison of MULES, isoAdvector isoAlpha and plicRDF-5 for both Cartesian and tetrahedral grids.

reference.

5. Bubble wake instability in spiraling regime

5.1. Definition of test case

In the last case, we consider a single rising bubble which undergoes a spiraling path. This case has been previously studied by Cano-Lozano [20]. The case is also available as an example from the Basilisk website (`http://basilisk.fr`) and is detailed in the article of Cano-Lozano [20].

The bubble rises along $+z$ direction and is initialized as a sphere at $z_0/D_0 = 3.5$, where D_0 is the bubble initial diameter. The density ratio ρ_1/ρ_2 between the fluids is 1000 and the dynamic viscosity ratio μ_1/μ_2 is 100. Index 1 refers to the continuous liquid phase while index 2 refers to the gas phase. The chosen Bond/Eötvös number $Bo = \rho_1\, g\, D_0^2/\sigma = 10$ and Galilei number $Ga = \rho_1\, g^{1/2}\, D_0^{3/2}/\mu_1 = 100.25$ classify the current bubble in the oscillatory dynamics regime, with dominant inertial forces [33]. In the simulations, the gravity g and first phase density ρ_1 are taken as unity, which gives a surface tension $\sigma = 0.1\,\mathrm{Nm}^{-1}$ and a rise velocity of the order of unity.

Basilisk simulations are realized inside a cubic tank of size $102.4^3 D_0$ and benefit from adaptive mesh refinement (AMR). Two different grids were used for Basilisk, at refinement levels 11 and 12. On the level 11 grid, the AMR method is limited to a minimum cell size of $102.4 D_0/2^{11} = 0.05 D_0$, which represents 20 cells per bubble diameter. The Basilisk level 12 grid thus corresponds to 40 cells per bubble diameter.

For OpenFOAM® simulations, we have chosen a fluid domain of size $32\times32\times128\ D_0$ in order to limit the computational cost of the simulations. OpenFOAM® computational grids were obtained by local refinements over a uniform background grid of $40\times40\times160$ cells. The background grid defines the refinement level 0, which thus corresponds to 1.25 cells per bubble diameter. A computational grid with refinement level up to 4 in regions where the bubble can be present was then created with the `snappyHexMesh` mesh generator. The level 4 corresponds to a division of cells by a factor 2^4, and so to 20 cells per bubble initial diameter in the refined regions. A finer grid with level 5 was also used, thus

corresponding to 40 cells per bubble diameter. This grid refinement is identical to Basilisk. In order to reduce the number of grid cells, the refinement at the maximum level has been limited to regions in the centerline of the fluid domain, along the bubble rising direction. A refinement cylindrical region of diameter $2D_0$ is imposed for $2 \leq z/D_0 \leq 32$. Then a cone of diameter varying between 2 and $4D_0$ is used above for $32 \leq z/D_0 \leq 64$. The top of the fluid domain is refined within a cylindrical region of diameter $4D_0$ for $64 \leq z/D_0 \leq 126$. The transition between levels is done through buffer layers of two cells (parameter `nCellsBetweenLevels` equal to 2). For example, the grid visible in figure 27 shows that the size jump between one fine level and its coarser neighbouring level is done with a transition layer of 2 cells. This method conducted to an overall grid size of 9.6 million cells at level 4 and of 72.9 million cells at level 5.

5.2. Results and discussion

For all OpenFOAM® solvers, the maximum CFL number was kept below 0.05 and the computations were run up to time 140 where the bubble reached the end of the domain. The very first result we obtained was that the MULES solver predicted bubble fragmentation. At time $t = 15\,\mathrm{s}$, the comparison of OpenFOAM® solvers bubble shapes on the level 4 grid, displayed on figure 27, clearly shows that the MULES bubble (in grey) is fragmented into one small trailing bubble and a main body bubble. This behavior is not physical and has not been observed with the other OpenFOAM® solvers, shown on figure 27 by the red and green bubble interfaces corresponding respectively to isoAlpha and plicRDF-5. Considered as a reference solution, Basilisk results did not show bubble breakup. Besides, more fragmentation with more trailing satellite bubbles were generated in time with MULES. In the following results, MULES data will thus not be discussed.

Bubble trajectories are displayed on figures 28, 29, 30 for both fine and coarse grids. Note that Basilisk transitioned earlier from a rectilinear path to a spiraling regime in comparison to the other VoF solvers at both grid resolution. The trajectory for iso-Alpha is 2D zigzag planar on the coarser grid, roughly

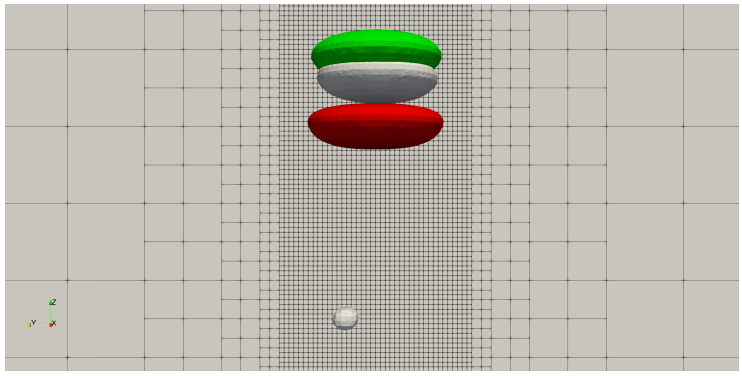

Figure 27: Bubble 26 interface at $t = 15\,\mathrm{s}$ for MULES (grey, showing fragmentation), isoAlpha (red) and plicRDF-5 (green) on the level 4 grid.

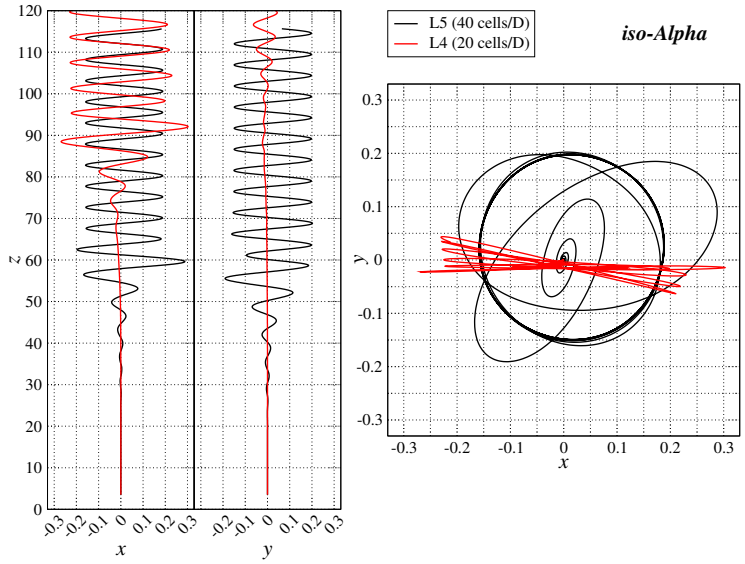

Figure 28: Bubble 26 trajectory for iso-Alpha.

in the plane $y = 0$. The trajectory for plicRDF-5 shows the development of a correct 3D spiraling path at the same refinement level. The amplitude of bubble's displacements is smaller with plicRDF-5 by a factor two. Basilisk shows an off-centered bubble path for the coarse grid resolution. On the finest grid level, the OpenFOAM® solvers predict the expected spiraling path, which leads to a circular trajectory in the xy plane. The diameter of the xy circle is larger for iso-Alpha, while plicRDF-5 conducts to a diameter equivalent to

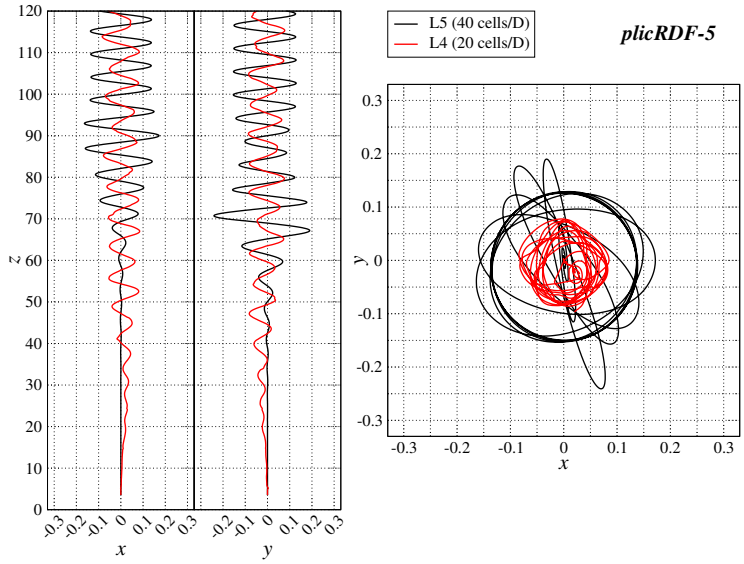

Figure 29: Bubble 26 trajectory for plicRDF-5.

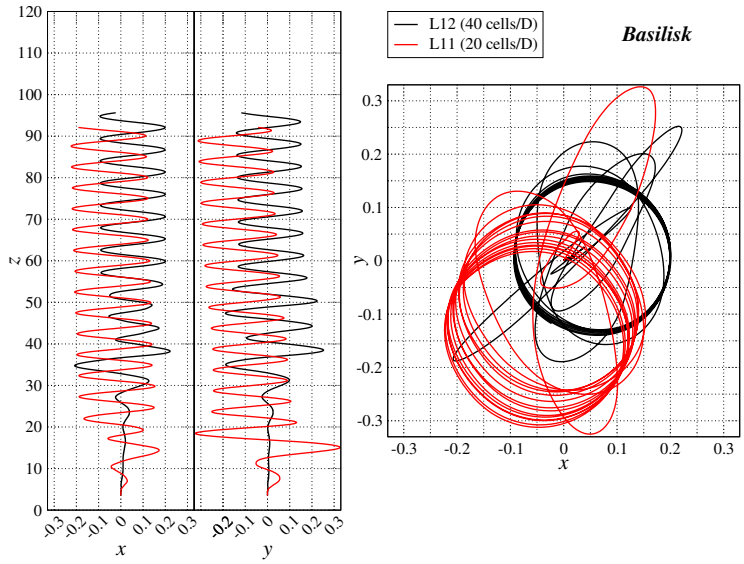

Figure 30: Bubble 26 trajectory for Basilisk.

Basilisk.

The frequency of the time signals of bubble centroid x and y coordinates were translated to Strouhal numbers $St = fD_0/U_0$ shown on table 2. The reference velocity U_0 was taken as unitary. OpenFOAM® solvers predict lower

frequencies than Basilisk on the coarse grid. On the finer grid, all solvers give identical results.

Table 2: Strouhal number for bubble 26 case.

VoF solver	**Strouhal**
isoAdvector isoAlpha	0.133 (20 cells/D_0); 0.150 (40 cells/D_0)
isoAdvector plicRDF-5	0.123 (20 cells/D_0); 0.150 (40 cells/D_0)
Basilisk	0.159 (20 cells/D_0); 0.150 (40 cells/D_0)

The bubble Reynolds number is shown in figure 31. The Reynolds number is based on the z velocity component of the bubble. Reynolds numbers are of the same order between the solvers, within a 12% range. On the coarse grid, isoAlpha predicts a largely oscillating Reynolds after the bubble starts its spiraling path. plicRDF-5 results present similar trends of small amplitude oscillating velocity as Basilisk. On the fine grid, we note that the bubble reaches a stationary rising velocity that is very similar between Basilisk and plicRDF-5, while isoAlpha predicts a smaller value.

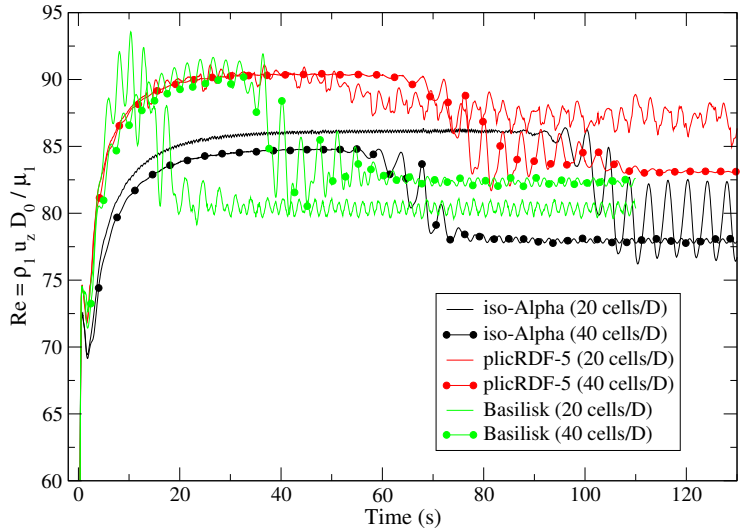

Figure 31: Bubble 26 Reynolds number as a function of time.

6. Conclusion

We have presented quantitative validations of the isoAdvector method (isoAlpha and plicRDF-5) against MULES and Basilisk, and against other solvers reference data of the literature. Three test cases have been used. The first test case aims at quantifying the spurious currents obtained in the three OpenFOAM® VoF variants tested here (MULES, isoAlpha and plicRDF-5). This configuration consists in a stagnant single bubble in a quiescent liquid, under zero gravity conditions. The new reconstruction method, plicRDF-5, significantly reduces the spurious currents due to its more accurate interface curvature calculation. Moreover, the plicRDF-5 reconstruction method demonstrates a better prediction of the pressure jump across the bubble. The second test case is the Hysing benchmark, as originally published by Hysing [11]. This benchmark simulates a single rising bubble in an initially quiescent liquid. The case has been extended to 3D using similar boundary conditions. isoAdvector has been verified to work for rising bubble simulation with similar or greater accuracy as MULES and with a sharper interface and slightly smaller calculation times. These results demonstrate that isoAdvector can be used for surface tension dominated flows. However, the sharper interface poses a challenge to the surface tension model in some simulations, for example at high resolutions or on unstructured triangular-prisms grids. The new reconstruction method, plicRDF-5, rectifies these problems. The last test case of this paper is a single bubble rising in a large tank taken from case number 26 of Cano-Lozano [20]. Two different grid resolutions are used and compared to Basilisk reference. On the coarsest grid, plicRDF-5 method was the only OpenFOAM® solver able to capture the expected spiraling trajectory. On the finest grid, both isoAdvector solvers demonstrated a better behavior, although the prediction of bubble trajectory and rising velocity goes in favour of plicRDF-5. Materials and reference results for benchmark cases used in this study can be downloaded from the OpenFOAM® wiki tutorial website at the first author URL `https://wiki.openfoam.com/Collection_by_authors#Lionel_Gamet`.

Acknowledgements

This work was granted access to the HPC resources of CINES under the allocation 2018-AP012B10362 made by GENCI.

L.G., M.S. and J.L.P. thank the IFP Energies Nouvelles research program FoReCaSt for funding this research and providing additional computing resources.

References

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

[2] S. Osher, J. A. Sethian, Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations, J. Comp. Phys. 79 (1) (1988) 12 – 49. doi:https://doi.org/10.1016/0021-9991(88)90002-2.

[3] X. Cai, H. Marschall, M. Wörner, O. Deutschmann, Numerical simulation of wetting phenomena with a phase-field method using OpenFOAM®, Chemical Engineering & Technology 38 (11) (2015) 1985–1992. doi:10.1002/ceat.201500089.

[4] F. Jamshidi, H. Heimel, M. Hasert, X. Cai, O. Deutschmann, H. Marschall, M. Wörner, On suitability of phase-field and algebraic volume-of-fluid OpenFOAM® solvers for gas-liquid microfluidic applications, Computer Physics Communications 236 (2019) 72–85.

[5] ESI, OpenFOAM, The Open Source CFD Toolbox, User Guide. Version v1912.
URL http://www.openfoam.com/documentation/user-guide

[6] J. Roenby, H. Bredmose, H. Jasak, A computational method for sharp interface advection, Royal Society Open Science 3 (11). doi:10.1098/rsos.160405.

[7] H. Scheufler, J. Roenby, Accurate and efficient surface reconstruction from volume fraction data on general meshes, J. Comp. Phys. 383 (2019) 1 – 23. doi:10.1016/j.jcp.2019.01.009.

[8] H. G. Weller, A new approach to vof-based interface capturing methods for incompressible and compressible flow, OpenCFD Ltd., Report TR/HGW/04.

[9] S. Popinet, An accurate adaptive solver for surface-tension-driven interfacial flows, Journal of Computational Physics 228 (16) (2009) 5838–5866.

[10] T. Abadie, J. Aubin, D. Legendre, On the combined effects of surface tension force calculation and interface advection on spurious currents within volume of fluid and level set frameworks, Journal of Computational Physics 297 (2015) 611–636. doi:10.1016/j.jcp.2015.04.054.

[11] S. Hysing, S. Turek, D. Kuzmin, N. Parolini, E. Burman, S. Ganesan, L. Tobiska, Quantitative benchmark computations of two-dimensional bubble dynamics, I.J.N.M.F. 60 (11) (2009) 1259–1288. doi:10.1002/fld.1934. URL http://www.featflow.de/en/benchmarks/cfdbenchmarking/bubble.html

[12] D. Zuzio, J. Estivalezes, An efficient block parallel amr method for two phase interfacial flow simulations, Computers and Fluids 44 (1) (2011) 339–357. doi:10.1016/j.compfluid.2011.01.035.

[13] N. Balcázar, O. Lehmkuhl, L. Jofre, J. Rigola, A. Oliva, A coupled volume-of-fluid/level-set method for simulation of two-phase flows on unstructured meshes, Computers and Fluids 124 (2016) 12–29. doi:10.1016/j.compfluid.2015.10.005.

[14] N. Kumar Singh, B. Premachandran, A coupled level set and volume of fluid method on unstructured grids for the direct numerical simulations of two-phase flows including phase change, International Journal of Heat and Mass Transfer 122 (2018) 182–203. doi:10.1016/j.ijheatmasstransfer.2018.01.091.

[15] J. Patel, G. Natarajan, A novel consistent and well-balanced algorithm for simulations of multiphase flows on unstructured grids, Journal of Computational Physics 350 (2017) 207–236. doi:10.1016/j.jcp.2017.08.047.

[16] J. Manik, A. Dalal, G. Natarajan, A generic algorithm for three-dimensional multiphase flows on unstructured meshes, International

Journal of Multiphase Flow 106 (2018) 228–242. doi:10.1016/j.ijmultiphaseflow.2018.04.010.

[17] M. Febres, D. Legendre, Enhancement of a 2D front-tracking algorithm with a non-uniform distribution of lagrangian markers, Journal of Computational Physics 358 (2018) 173–200. doi:10.1016/j.jcp.2017.12.021.

[18] R. C. Branco, B. de Barros Mendes Kassar, J. Carneiro, A.Nieckele, Accuracy evaluation of numerical predictions of a single rising bubble with VOF models, 2019, 25th ABCM International Congress of Mechanical Engineering, Uberlandia, MG, Brazil.
URL https://eventos.abcm.org.br/cobem2019/

[19] J. Adelsberger, P. Esser, M. Griebel, S. Groß, M. Klitz, A. Rüttgers, 3D incompressible two-phase flow benchmark computations for rising droplets, 2014, proceedings of the 11th World Congress on Computational Mechanics (WCCM XI), Barcelona, Spain, also available as INS Preprint No. 1401 and as IGPM Preprint No. 393.
URL http://wissrech.ins.uni-bonn.de/research/projects/risingbubblebenchmark/

[20] J. Cano-Lozano, C. Martínez-Bazán, J. Magnaudet, J. Tchoufag, Paths and wakes of deformable nearly spheroidal rising bubbles close to the transition to path instability, Physical Review Fluids 1 (5). doi:10.1103/PhysRevFluids.1.053604.

[21] S. Popinet, A quadtree-adaptive multigrid solver for the serre–green–naghdi equations, Journal of Computational Physics 302 (2015) 336–358.

[22] S. S. Deshpande, L. Anumolu, M. F. Trujillo, Evaluating the performance of the two-phase flow solver interfoam, Comp. Sc. & Discovery 5 (1) (2012) 014016.
URL http://stacks.iop.org/1749-4699/5/i=1/a=014016

[23] H. Marschall, K. Hinterberger, C. Schler, F. Habla, O. Hinrichsen, Numerical simulation of species transfer across fluid interfaces in free-surface flows using OpenFOAM, Ch. Eng. Sc. 78 (2012) 111 – 127. doi:https://doi.org/10.1016/j.ces.2012.02.034.

[24] A. Q. Raeini, M. J. Blunt, B. Bijeljic, Modelling two-phase flow in porous media at the pore scale using the volume-of-fluid method, J. Comp. Phys. 231 (17) (2012) 5653 – 5668. doi:https://doi.org/10.1016/j.jcp.2012.04.011.

[25] D. A. Hoang, V. van Steijn, L. M. Portela, M. T. Kreutzer, C. R. Kleijn, Benchmark numerical simulations of segmented two-phase flows in microchannels using the volume of fluid method, Computers & Fluids 86 (2013) 28 – 36. doi:https://doi.org/10.1016/j.compfluid.2013.06.024.

[26] C. Bilger, M. Aboukhedr, K. Vogiatzaki, R. Cant, Evaluation of two-phase flow solvers using level set and volume of fluid methods, J. Comp. Phys. 345 (2017) 665 – 686. doi:https://doi.org/10.1016/j.jcp.2017.05.044.

[27] J. Roenby, B. E. Larsen, H. Bredmose, H. Jasak, A new Volume-of-Fluid method in OpenFOAM, in: VII International Conference on Computational Methods in Marine Engineering, MARINE 2017, 2017.

[28] J. Roenby, H. Bredmose, H. Jasak, IsoAdvector: Geometric VOF on general meshes, in: OpenFOAM - Selected papers of the 11th Workshop, Springer, 2018.

[29] S. J. Cummins, M. M. Francois, D. B. Kothe, Estimating curvature from volume fractions, Computers & Structures 83 (6-7) (2005) 425–434. doi:10.1016/j.compstruc.2004.08.017.

[30] J. B. Bell, P. Colella, H. M. Glaz, A second-order projection method for the incompressible Navier-Stokes equations, Journal of Computational Physics 85 (1989) 257–283.

[31] S. Popinet, Numerical models of surface tension, Annual Review of Fluid Mechanics 50 (2018) 49–75. doi:10.1146/annurev-fluid-122316-045034.

[32] M. M. Francois, S. J. Cummins, E. D. Dendy, D. B. Kothe, J. M. Sicilian, M. W. Williams, A balanced-force algorithm for continuous and sharp interfacial surface tension models within a volume tracking framework, Journal of Computational Physics 213 (1) (2006) 141–173.

[33] M. K. Tripathi, K. C. Sahu, R. Govindarajan, Dynamics of an initially spherical bubble rising in quiescent liquid, Nature Communications 6.

[34] S. Turek, Efficient Solvers for Incompressible Flow Problems: An Algorithmic and Computational Approache, Vol. 6, Springer Science & Business Media, 1999.

[35] N. Parolini, E. Burman, A finite element level set method for viscous free-surface flows, in: Applied and Industrial Mathematics in Italy, World Scientific, 2005, pp. 416–427.

[36] V. John, G. Matthies, Moonmd - a program package based on mapped finite element methods, Computing and Visualization in Science 6 (2-3) (2004) 163–170.

[37] S. Ganesan, G. Matthies, L. Tobiska, On spurious velocities in incompressible flow problems with interfaces, Computer Methods in Applied Mechanics and Engineering 196 (7) (2007) 1193–1202.