



# A massively parallel accurate conservative level set algorithm for simulating turbulent atomization on adaptive unstructured grids

Romain Janodet, Carlos Guillaumon, Vincent Moureau, Renaud Mercier, Ghislain Lartigue, Pierre Bénard, Thibaut Ménard, Alain Berlemont

## ► To cite this version:

Romain Janodet, Carlos Guillaumon, Vincent Moureau, Renaud Mercier, Ghislain Lartigue, et al.. A massively parallel accurate conservative level set algorithm for simulating turbulent atomization on adaptive unstructured grids. Journal of Computational Physics, 2022, 458, 10.1016/j.jcp.2022.111075 . hal-03024186

**HAL Id: hal-03024186**

**<https://hal.science/hal-03024186>**

Submitted on 25 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.

# A massively parallel accurate conservative level set algorithm for simulating turbulent atomization on adaptive unstructured grids

ROMAIN JANODET<sup>1,2,\*</sup>, CARLOS GUILLAMÓN<sup>2</sup>, VINCENT MOUREAU<sup>1</sup>, RENAUD MERCIER<sup>2</sup>,  
GHISLAIN LARTIGUE<sup>1</sup>, PIERRE BÉNARD<sup>1</sup>, THIBAUT MÉNARD<sup>1</sup>, ALAIN BERLEMONT<sup>1</sup>

<sup>1</sup> CORIA, CNRS UMR6614, INSA and University of Rouen, Avenue de l'Université,  
76801 Saint-Etienne-du-Rouvray, France

<sup>2</sup> Safran Tech, Rue des Jeunes Bois, Chateaufort, 78114 Magny-Les-Hameaux, France

\* Corresponding author: romain.janodet@coria.fr

*November 10, 2020*

## Abstract

This article presents a massively parallel and robust strategy to perform the simulation of turbulent incompressible two-phase flows on unstructured grids in complex geometries. This strategy relies on a combination of a narrow-band accurate conservative level set/ghost-fluid framework with isotropic adaptive mesh refinement. This combination enables to accurately capture interface dynamics and topology, and the small physical scales at the liquid-gas interface are resolved at an affordable cost. In the accurate conservative level set framework, presented first in [O. Desjardins et al, J. Comput. Phys. 227 (2008) 8395-8416], the interface is defined as the iso-contour of a hyperbolic tangent function, which is advected by the fluid, and then reshaped using a reinitialization equation. Several forms of this reinitialization exist: the original ACLS form proposed by Desjardins et al. involves numerical estimation of the hyperbolic tangent gradient, which is difficult to compute accurately on unstructured meshes. It is thus susceptible to induce artificial deformation of the interface. A new form has been recently proposed in [R. Chiodi et al, J. Comput. Phys. 343 (2017) 186-200], which takes advantage of a mapping onto a classical distance level set while much better preserving the interface shape. Nevertheless, the implementation of this new form on unstructured grids requires special care. In this work, a robust implementation of this new form on unstructured meshes is proposed and implemented in the YALES2 low-Mach flow solver. In order to compute interface normals and curvature, the signed-distance function is reconstructed in parallel at nodes in the narrow band around the interface using a Geometric-Projection Marker Method (GPMM). This method relies on the triangulation of the level set iso-contour and exact geometric projection to the closest surface elements. Spatial convergence, robustness and efficiency of the overall procedure are firstly demonstrated through classical interface transport test cases and two-phase flow examples. Eventually, to emphasize the significant computational gain using adaptive mesh refinement and the ability to compute complex turbulent flows with large density ratios, two Large-Eddy Simulations (LES) of atomizing liquid jets in air are presented, each one at various resolutions. The first one is a low-pressure water jet in quiescent air from a compound nozzle with full computation of the internal injector flow, while the latter is a high-pressure kerosene jet in crossflow. Both simulations are validated against experiments, demonstrating the potential of the method to access a deep numerical insight into jet instabilities and internal flow dynamics with 3D unstructured meshes.

Keywords: Multiphase flows; Atomization; Unstructured grids; Incompressible flow LES; Conservative level set; Adaptive mesh refinement

## 1 Introduction

Multiphase flows are ubiquitous in nature and in industrial systems, and understanding the formation of sprays is crucial for designing complex injection technologies. More specifically, turbulent primary atomization is of first importance in aeronautical combustors, into which fuel is injected in liquid form, goes under atomization, evaporation, turbulent mixing and eventually combustion. All these physical processes are strongly dependent, and thus improving numerical predictions of pollutant formation requires deep numerical insights into the atomization process. This process designates the evolution of large liquid structures into small droplets and thus governs the liquid droplet distribution in the combustor and the subsequent evaporation. Realistic industrial

injector geometries for atomization simulations are very complex, including sharp edges and narrow corners, which hinders the meshing task with cartesian grids: the use of unstructured meshes is thus well suited for this type of simulations [1]. Although the literature remains limited about unstructured atomization computations, previous studies on static meshes can be found, such as [2, 3, 4, 5, 6, 7, 8, 9] for Volume Of Fluid (VOF) interface capturing techniques [10], or [11, 12, 13, 14, 15] for Level Set methods (LS) [16, 17].

Despite many years of study, high-performance atomization simulations remain a challenge: the wide range of involved time and space scales leads to important calculation costs. To tackle this issue, the use of Adaptive Mesh Refinement (AMR) is particularly helpful, allowing the allocation of computational resources to the interface dynamics. A popular technique for solving incompressible two-phase flows on adaptive cartesian grids is tree-based AMR (quad/octree), as in the work of Popinet [18, 19]. Regarding unstructured meshes, Dai and Schmidt [20] proposed a moving-mesh algorithm to simulate free-surface flows with large deformations on adaptive tetrahedral grids. Both studies are proposed in the context of VOF methods, and the meshes are typically refined based on the local curvature of the interface. For classical Level Set methods on adaptive tetrahedral grids, Morgan and Waltz [21] introduced a method to treat evolving fronts with specified velocity, while Rodriguez et al. presented a parallel two-phase flow solver using a finite-element formulation in [22], where the adaptive mesh is refined based on the distance to the interface.

In this work, the Accurate Conservative Level Set (ACLS) method is used to capture the interface on 2D and 3D unstructured grids: this method accurately predicts the interface dynamics while conserving liquid mass [23, 24, 25]. ACLS on unstructured grids has been proposed before in [12, 13] on static meshes, and by Antepara et al. [26] for adaptive tetrahedral grids. In this latter, non-conforming AMR with an octree data structure is used. In the present work, massively distributed adaptive conforming unstructured grids are employed, and adapted using both metric-based and skewness-based quality criteria. We extend these previous unstructured ACLS studies by including an adapted and improved reinitialization method, first introduced by Chiodi and Desjardins for cartesian grids [27] and that has never been implemented to date for unstructured grids. In particular, interface-merging properties of this reinitialization equation are discussed. Moreover, a narrow band around the liquid-gas interface is built to compute interface features only in the area of interest [28], allowing significant computational savings.

This article is organized as follows: the various parts of the procedure implemented in the YALES2 unstructured low-Mach number code [29] are first presented, namely the ACLS interface-capturing technique, the fractional-step integration of Navier-Stokes equations with explicit treatment of jump conditions, and the dynamic mesh adaptation strategy. The accuracy, spatial convergence, and computational efficiency of the unstructured ACLS method are assessed on classical interface transport test cases. The behavior of the overall algorithm in the case of canonical capillary-driven two-phase flows is then verified. Finally, two LES simulations in complex geometries are performed on adaptive tetrahedral meshes: the break-up of a turbulent liquid jet from a low-pressure compound nozzle [30, 31] and of a non-reactive high-pressure liquid jet in crossflow are shown at various resolutions, demonstrating the accessibility of jet instabilities with 3D adaptive unstructured meshes. Results are validated against experiments performed in the same conditions [32, 33].

## 2 Mathematical formulation

### 2.1 Accurate Conservative Level Set framework

In the Accurate Conservative Level Set (ACLS) framework [25], the liquid-gas interface  $\Gamma$  is represented using a hyperbolic tangent profile:

$$\psi(\mathbf{x}, t) = \frac{1}{2} \left( \tanh \left( \frac{\phi(\mathbf{x}, t)}{2\varepsilon} \right) + 1 \right) \quad (1)$$

where the parameter  $\varepsilon$  sets the thickness of the profile, and  $\phi(\mathbf{x}, t) = \pm|\mathbf{x}(t) - \mathbf{x}_\Gamma(t)|$  is the signed-distance function.

Using  $\psi$ , the interface is located at the iso-level 1/2:

$$\Gamma(t) = \{\mathbf{x} \in \mathbb{R}^3 | \psi(\mathbf{x}, t) = 1/2\} \quad (2)$$

as depicted in the one-dimensional example of Fig. 1. In this figure, a characteristic profile thickness  $\delta_\psi$  is used, computed based on the maximum gradient of  $\psi$ :

$$\delta_\psi = \frac{1}{\max |\nabla \psi|} = 4\varepsilon \quad (3)$$

Assuming the flow velocity field  $\mathbf{u}$  is divergence free, the scalar  $\psi$  is advected by the fluid:

$$\frac{\partial \psi}{\partial t} + \nabla \cdot (\psi \mathbf{u}) = 0 \quad (4)$$

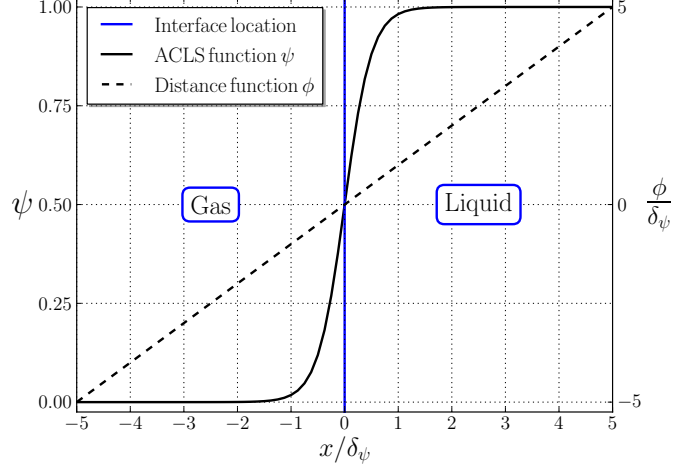


Figure 1: Interface capturing on a one-dimensional example using the ACLS method.  $x_\Gamma = 0$ ;  $\delta_\psi = 4\varepsilon$ .

and then reshaped using the reinitialization equation of [25]:

$$\frac{\partial \psi}{\partial \tau} = \nabla \cdot \left( \underbrace{\varepsilon(\nabla \psi \cdot \mathbf{n})\mathbf{n}}_{\text{Diffusion}} - \underbrace{\psi(1-\psi)\mathbf{n}}_{\text{Resharpener}} \right) \quad (5)$$

where  $\tau$  is a pseudo-time, and  $\mathbf{n}$  is the interface normal:

$$\mathbf{n} = \frac{\nabla \phi}{|\nabla \phi|} \quad (6)$$

A new form of the reinitialization equation was recently proposed by Chiodi et al. in [27]:

$$\frac{\partial \psi}{\partial \tau} = \nabla \cdot \left( \frac{1}{4 \cosh^2 \left( \frac{\phi_{\text{map}}}{2\varepsilon} \right)} (|\nabla \phi_{\text{map}} \cdot \mathbf{n}| - 1) \mathbf{n} \right) \quad (7)$$

where  $\phi_{\text{map}} = \varepsilon \ln(\psi/(1-\psi))$  is an analytical signed-distance function, mapped for  $\psi \in ]0; 1[$ . More details regarding the history of conservative level-set reinitialization equations can be found in [34].

The signed-distance function  $\phi$  is reconstructed at nodes in the narrow band around the interface using a Geometric-Projection Marker Method (GPMM), firstly introduced in [35], to estimate the smallest distance to the interface. The mean curvature  $\kappa$  is computed directly from  $\phi$  using Goldman's formula [36]:

$$\kappa = \frac{\text{Tr}(\mathcal{H}(\phi)) - \frac{\nabla \phi^T}{|\nabla \phi|} \cdot \mathcal{H}(\phi) \cdot \frac{\nabla \phi}{|\nabla \phi|}}{|\nabla \phi|} \quad (8)$$

where  $\mathcal{H}(\phi)$  is the Hessian matrix of the signed-distance function.

The ACLS method is not strictly conservative for two reasons. First, the approximate and exact enclosed volumes of liquid are respectively:

$$\tilde{V}_l = \int_V \psi \, dV \quad \text{and} \quad V_l = \int_V H(\psi - 1/2) \, dV \quad (9)$$

where  $H$  is the Heaviside function (0 in gas, 1 in liquid). In the ACLS method, the approximate volume  $\tilde{V}_l$  is conserved, but differs from the exact volume  $V_l$  due to interface curvature, i.e. in the ubiquitous case of non-straight interfaces.

A second reason for losing liquid mass on irregular meshes is the variation of the mesh resolution from fine to coarse, which leads to the smoothing of  $\psi$ , and may imply a loss of the  $\psi = 1/2$  iso-surface. In this case, the interface is not detected anymore and gas can be set instead of liquid.

Finally, the reinitialization process on non-homogeneous meshes may also induce errors in the transport speed: the profile thickness parameter  $\varepsilon$  is mesh dependant, typically set to  $\Delta x/2$ . Spatial variations of this parameter induce changes in the profile thickness, which in turn may induce errors on the transport velocity of the  $\psi = 1/2$  iso-surface.

Because of all these numerical errors on non-homogeneous meshes, special attention is needed when coupling the ACLS procedure with dynamic mesh adaptation, so that the interface never encounters strong cell-size gradients.



## 2.2 Navier-Stokes equations

The momentum equation for an incompressible Newtonian fluid is written in each of the phases as:

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot (\mathbf{u} \otimes \mathbf{u}) = -\frac{1}{\rho} \nabla p + \nabla \cdot \boldsymbol{\tau} + \mathbf{g} \quad (10)$$

with  $\nabla \cdot \mathbf{u} = 0$ . The flow velocity is assumed continuous in both normal and tangential directions across the liquid/gas interface  $\Gamma$ , and hence there is no velocity jump:  $[\mathbf{u}]_\Gamma = 0$ . However, pressure and density are discontinuous across  $\Gamma$  [37]. Surface tension  $\sigma$  is embedded in the pressure jump  $[p]_\Gamma$ , and, without Marangoni forces, the pressure jump is written as:

$$[p]_\Gamma = p_{l,\Gamma} - p_{g,\Gamma} = \sigma \kappa_\Gamma + 2[\mu]_\Gamma \mathbf{n}^t \cdot \nabla \mathbf{u} \cdot \mathbf{n} \quad (11)$$

It is numerically treated using the Ghost-Fluid Method (GFM) [38]. In Eq. (11),  $\mathbf{n}$  is the interface normal from Eq. (6),  $\kappa_\Gamma$  is the total interface curvature,  $p_{l,\Gamma}$  and  $p_{g,\Gamma}$  are the liquid and gas pressure at the interface, respectively. The material properties are reconstructed as:

$$\rho(\mathbf{x}, t) = \rho_g + (\rho_l - \rho_g)H(\psi(\mathbf{x}, t) - 1/2) \quad \text{and} \quad \mu(\mathbf{x}, t) = \mu_g + (\mu_l - \mu_g)\psi(\mathbf{x}, t) \quad (12)$$

for the density and the dynamic viscosity, respectively.

The linear solver used to solve the Poisson equation for pressure is the Deflated Preconditioned Conjugate Gradient of [39].

## 3 Numerical formulation for unstructured grids

### 3.1 Accurate Conservative Level Set algorithm

#### 3.1.1 Narrow-band ACLS

In order to reduce computational costs, interface features such as signed distance, normals, curvature and reinitialization fluxes are computed in a restricted region around the interface, called "narrow band" [28]. Each node  $i$  belonging to the narrow band is identified with a signed integer  $b_i$  ( $b > 0$  in liquid and  $b < 0$  in gas), depending on the node layer it is part of. The flagging of the narrow band is illustrated in Fig. 2 (a). Using this

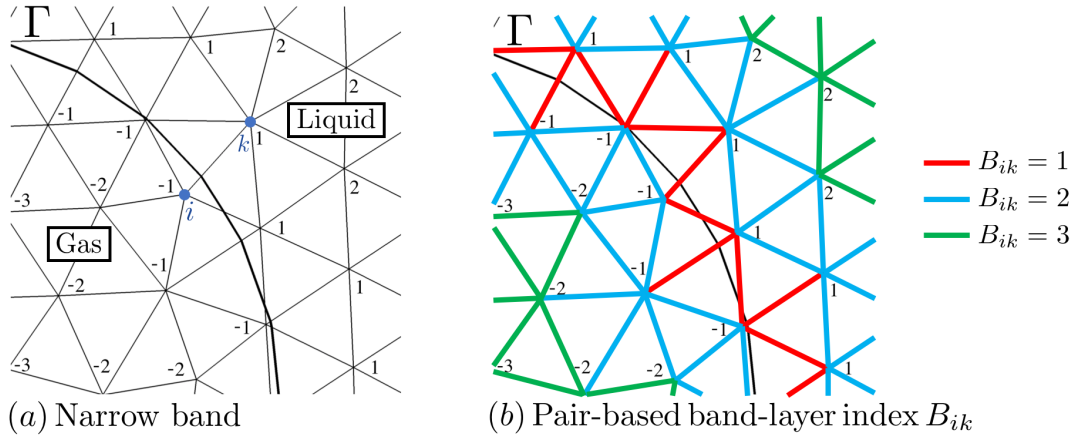


Figure 2: (a) Flagging of the narrow band around the interface  $\Gamma$ , here  $ik$  is a pair of nodes crossing the interface; (b) Pair-based band-layer index  $B_{ik}$  for data propagation in the Geometric-Projection Marker Method (GPMM).

methodology, an edge  $ik$  crossing the interface verifies  $b_i b_k = -1$ . In practice, the band width is set between 8 and 12 cells on each side of the interface:  $8 \leq |b|_{\max} \leq 12$ .

#### 3.1.2 Signed-distance function reconstruction: the GPMM algorithm

To compute the interface normals (Eq. (6)) and curvature (Eq. (8)), a reconstructed signed-distance function  $\phi$  is used. As detailed in [25], using this smooth function instead of  $\psi$  for the computation of normal avoids spurious orientation of this normal vector, which is thus appropriate for the reinitialization equation. This distance reconstruction is challenging on unstructured meshes, and very few massively parallel algorithms are able to do it. In the work of Dapogny and Frey [40], the signed-distance function is reconstructed on unstructured

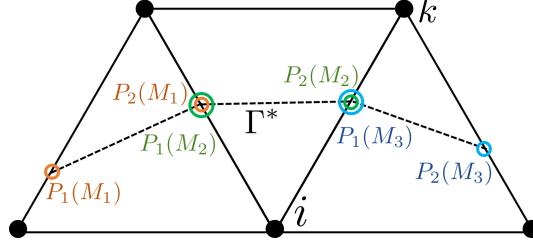


Figure 3: Segmentation of the interface in 2D.  $P_K(M_I)$  is the  $K$ -th intersection point associated with the  $I$ -th marker  $M_I$ .

meshes from a triangulation of the interface using the unsteady Eikonal equation. In particular, the reduction of triangulation-induced numerical errors when refining the mesh close to the interface is discussed and quantified. In this work, the interface is also approximated by a triangulation before reconstructing the distance, but it has been chosen to proceed geometrically using a pair-based Fast-Marching Method (FMM) for its computation from the explicit surface. The methodology is the following: (i) a preliminary sorting of the node pairs in the narrow band is performed to be able to sweep from the close-interface band layers to the ones further; (ii) a segmentation (2D) or triangulation (3D) of the interface within grid cells is done; (iii) a Fast-Marching Method (FMM) is then performed along the edges to build a collection of the closest segments (2D) or triangles (3D) at each node of the band; (iv) the signed-distance function is computed from an exact geometric projection at each node of the band. All these steps constitute the Geometric-Projection Marker Method (GPMM), first introduced in [35] and presented here in more details.

**Sorting of the node pairs** In order to propagate data efficiently from the first band level  $|b| = 1$  to the furthest  $|b| = |b|_{\max}$ , sorting the pairs of nodes layer by layer in the narrow band is necessary. To do so, a pair-based band-layer index  $B_{ik}$  is defined as follows:

$$B_{ik} = \begin{cases} 1 & \text{if } b_i b_k = -1 \\ \min(|b_i|, |b_k|) + 1 & \text{if } b_i b_k \neq -1 \end{cases} \quad (13)$$

with  $1 \leq B_{ik} \leq |b|_{\max} + 1$ . This index is depicted in Fig. 2 (b).

**Determination of intersection points** It is then needed to compute the position  $\mathbf{x}_P$  of the intersection points  $P$  between the interface  $\Gamma$  and the edges  $ik$  crossing the interface ( $B_{ik} = 1$ ). Given the conservative level-set functions  $\psi_i$  and  $\psi_k$  (Eq. (1)) at the nodes  $i$  and  $k$  belonging to a crossing pair  $ik$ , one can invert these functions to build two distance functions to the interface  $\phi_{\text{inv},i}$  and  $\phi_{\text{inv},k}$ :

$$\begin{cases} \phi_{\text{inv},i} = 2\varepsilon_i \tanh(2\psi_i - 1) \\ \phi_{\text{inv},k} = 2\varepsilon_k \tanh(2\psi_k - 1) \end{cases} \quad (14)$$

The level-set position  $\theta$  on a pair  $ik$  crossing the interface is then calculated as:

$$\theta = \frac{\phi_{\text{inv},i}}{\phi_{\text{inv},i} - \phi_{\text{inv},k}} \quad (15)$$

which means that  $\theta = 0$  and  $\theta = 1$  correspond to  $\Gamma$  at nodes  $i$  and  $k$ , respectively. Thus, if the pair of nodes  $ik$  verifies the condition  $(\psi_i - 0.5)(\psi_k - 0.5) < 0$ , the position of the intersection point is eventually computed by interpolating the nodal coordinates as:

$$\mathbf{x}_P = \mathbf{x}_i + \theta(\mathbf{x}_k - \mathbf{x}_i) = \mathbf{x}_i + \theta \Delta \mathbf{x}_{ik} \quad (16)$$

**Segmentation/Triangulation of the interface** In order to compute the distance to an explicit interface, noted  $\Gamma^*$  in the following, a segmentation in 2D or a triangulation in 3D of the implicit interface  $\Gamma$  is first performed. This step is illustrated in 2D in Fig. 3. The concept of marker stems from this procedure. A marker  $M$  is an array containing the coordinates  $\mathbf{x}_K(M) = (x_K(M), y_K(M), z_K(M))$  of its associated intersection points  $P_K(M)$  seen in Fig. 3. Hence, for 2D triangular grids, each interface segment is defined by a set of two intersection points ( $K = 1, 2$ ) stored in a marker. For 3D tetrahedral meshes, there can be three or four intersection points depending on how the interface intersects a tetrahedron. For a three-point intersection, the interface within the cell is defined by a single triangle. In the case of a four-point intersection, the resulting

quadrilateral is divided into four triangles to ease the storing of the surface elements, leading in both cases to three points stored in one marker in 3D ( $K = 1, \dots, 3$ ).

Each node of the narrow band knows a maximum fixed number of markers  $N_M$  defined by the user ( $I = 1, \dots, N_M$ ), which is usually set to 3 in 2D and 10 in 3D. Indeed, as studied in [35], propagating several markers significantly improves the accuracy and spatial convergence of the method. A set of markers known by a node  $i$  of the narrow band is a marker list. Each marker  $M$ , which is a node-based array  $M_i$ , also contains the absolute distance  $d(M_i)$  from the node  $i$  to the associated surface element, and two identifiers  $id_{1,2}(M_i)$ .  $id_1$  is the processor number and  $id_2$  is the mesh element number in which the intersection points are. A 2D nodal marker list is presented in Table 1.

Table 1: A 2D marker list for a node  $i$  of the narrow band: 2 identifiers  $id_{1,2}$ , 4 coordinates ( $K = 1, 2$ ), 1 distance  $d$ .

$M_{1,i}$	...	$M_{I,i}$	...	$M_{N_M,i}$
$id_1(M_{1,i})$	...	$id_1(M_{I,i})$	...	$id_1(M_{N_M,i})$
$id_2(M_{1,i})$	...	$id_2(M_{I,i})$	...	$id_2(M_{N_M,i})$
$x_1(M_{1,i})$	...	$x_1(M_{I,i})$	...	$x_1(M_{N_M,i})$
$y_1(M_{1,i})$	...	$y_1(M_{I,i})$	...	$y_1(M_{N_M,i})$
$x_2(M_{1,i})$	...	$x_2(M_{I,i})$	...	$x_2(M_{N_M,i})$
$y_2(M_{1,i})$	...	$y_2(M_{I,i})$	...	$y_2(M_{N_M,i})$
$d(M_{1,i})$	...	$d(M_{I,i})$	...	$d(M_{N_M,i})$

The markers will then be sorted by distance for each node  $i$ , within the marker list so that:

$$d(M_{1,i}) < \dots < d(M_{I,i}) < \dots < d(M_{N_M,i}) \quad (17)$$

**Sorting of markers and distance computation** When a marker is evaluated to update and fill marker lists, a check is first performed based on the two marker identifiers to identify whether the marker is already present in the marker list or not. If not, the marker distance needs to be estimated to place the new marker in the list. Two cases are then considered based on the nodal band-layer index  $b_i$ , and a threshold  $|b|_{\text{exact}} = 2$ : (i) in the first two band levels, i.e. if  $|b_i| \leq |b|_{\text{exact}}$ , the distance  $d$  is computed by performing an exact geometric projection; (ii) for the furthest band levels, i.e. if  $|b|_{\text{exact}} < |b_i| \leq |b|_{\text{max}}$ , the distance is initially approximated in order to speed-up the marker-sorting procedure. We focus on 2D triangular grids in the following for sake of simplicity.

In case (i), the geometric projection algorithm is used, illustrated in Fig. 4. If the projection is inside

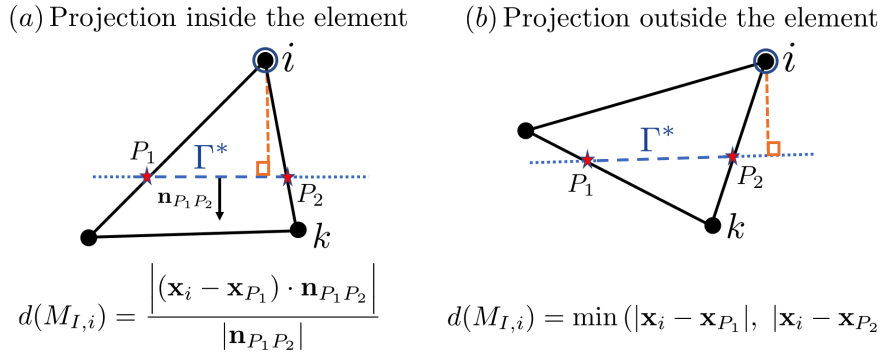


Figure 4: Projection algorithm to compute the exact geometric absolute distance  $d$  to the triangulated interface  $\Gamma^*$ .

the element (Fig. 4 (a)), the distance to the segment  $[P_1 P_2]$  stored in marker  $M_{I,i}$  at node  $i$  is computed geometrically as:

$$d(M_{I,i}) = \frac{|(\mathbf{x}_i - \mathbf{x}_{P_1}) \cdot \mathbf{n}_{P_1 P_2}|}{|\mathbf{n}_{P_1 P_2}|} \quad (18)$$

where  $\mathbf{n}_{P_1 P_2}$  is the normal vector to the interface segment  $[P_1 P_2]$ . Otherwise, if the projection is outside the element (Fig. 4 (b)), the distance  $d$  is the distance between the node  $i$  and the closest intersection point  $P_K$ :

$$d(M_{I,i}) = \min(|\mathbf{x}_i - \mathbf{x}_{P_1}|, |\mathbf{x}_i - \mathbf{x}_{P_2}|) \quad (19)$$

For case (ii), the distance is simply approximated for the sorting as:

$$d(M_{I,i}) = \min \left( \left| \mathbf{x}_i - \mathbf{x}_{P_1} \right|, \left| \mathbf{x}_i - \mathbf{x}_{P_2} \right|, \frac{\left| (\mathbf{x}_i - \mathbf{x}_{P_1}) + (\mathbf{x}_i - \mathbf{x}_{P_2}) \right|}{2} \right) \quad (20)$$

Once the marker distance is estimated, it is then needed to sort the marker by distance and thus include the new one in the nodal marker list. This procedure is presented in Algorithm 1.

---

**Algorithm 1** Update of a marker list at a node  $i$  of the narrow band. The new marker to be included in the list is  $M_{*,i}$ .

---

```

1: if  $d(M_{*,i}) \leq d(M_{1,i})$  then
2:   for  $I = N_M - 1, 1, -1$  do
3:      $M_{I+1,i} = M_{I,i}$ 
4:   end for
5:    $M_{1,i} = M_{*,i}$ 
6: else if  $d(M_{*,i}) \geq d(M_{N_M,i})$  then
7:   return
8: else
9:   for  $I = 1, N_M - 1$  do
10:    if  $d(M_{*,i}) < d(M_{I+1,i})$  then
11:      for  $I' = N_M - 1, 1 + I, -1$  do
12:         $M_{I'+1,i} = M_{I',i}$ 
13:      end for
14:       $M_{I+1,i} = M_{*,i}$ 
15:    return
16:  end if
17: end for
18: end if

```

---

**Optimization and parallelization** The present algorithm is iterative for parallelism purposes. YALES2 parallelization uses a double-domain decomposition: within a processor, grid elements are assembled in groups. To optimize the algorithm, it is necessary to check local changes, within a cell group or a processor.

In addition to the marker list array, another array which contains marker flags  $\mathcal{F}(M)$  (0 or 1) is created before propagating data (i.e. after treating the first band level) in the narrow band.  $\mathcal{F}_i(M) = 1$  means that the marker  $M$  is new in the list at node  $i$ : e.g. this flag would be turned to 1 right after lines 5 and 14 of Algorithm 1. This operation allows to have updated information on the status of markers during data propagation, and thus avoids unnecessary repetition of computational operations (e.g. redundant updates and sorting of markers).

Another optimization is to store the squared distance  $d^2(M_{I,i})$  in marker lists instead of the distance  $d(M_{I,i})$ , to avoid many calculations of square roots, which have seen to be very expensive due to the large number of distance computations in the present algorithm. Indeed, this minor change does not affect the comparison and sorting of markers.

**Final nodal signed-distance computation in the narrow band** Once all markers are sorted,  $M_{1,i}$  contains the smallest squared distance  $d^2(M_{1,i})$ . The signed-distance is thus simply computed at node  $i$  as  $\phi_i = \pm \sqrt{d^2(M_{1,i})}$  if  $|b_i| \leq |b|_{\text{exact}}$ , otherwise for the nodes where the distance  $d$  in markers has been only approximated for the sorting ( $|b|_{\text{exact}} < |b_i| \leq |b|_{\text{max}}$ ), the distance  $\phi_i$  is eventually computed using the projection algorithm presented above (Fig. 4) onto the closest surface element, known thanks to the marker sorting.

### 3.1.3 Finite-volume framework for unstructured grids: spatial discretization and notations

YALES2 is a pair-based finite-volume low-Mach solver with node-centered control volumes [29]. These control volumes  $V_i$  are based on the dual mesh, as shown in Fig. 5. The spatial discretization is ensured by 4th-order central finite-volume schemes, written on the dual mesh constructed from edge and element centers (dual cells connect centroids of cells to centroids of faces).

In this framework, second and fourth-order gradients of a quantity  $\zeta$  are computed at a given node  $i$  using the gradient theorem as:

$$\nabla \zeta|_i = \frac{1}{V_i} \sum_{k \in \mathcal{N}_i} \zeta_{ik} \mathbf{dA}_{ik} \quad (21)$$

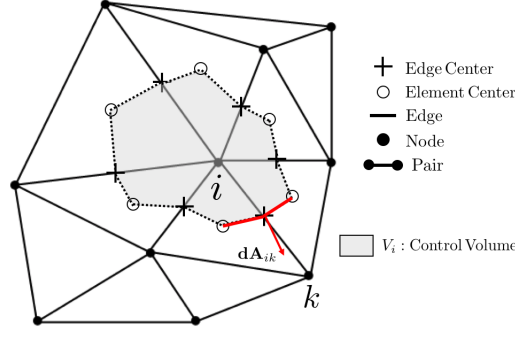


Figure 5: Finite-volume pair-based discretization in YALES2, using node-centered control volumes  $V_i$ .

$$\nabla \zeta|_i^{2\text{nd}} = \frac{1}{V_i} \sum_{k \in \mathcal{N}_i} \left( \frac{\zeta_i + \zeta_k}{2} \right) \mathbf{dA}_{ik} \quad (22)$$

$$\nabla \zeta|_i^{4\text{th}} = \frac{1}{V_i} \sum_{k \in \mathcal{N}_i} \left( \frac{\zeta_i + \zeta_k}{2} + \frac{\nabla \zeta|_i^{2\text{nd}} - \nabla \zeta|_k^{2\text{nd}}}{6} \cdot \Delta \mathbf{x}_{ik} \right) \mathbf{dA}_{ik} \quad (23)$$

with  $\mathcal{N}_i$  the set of neighbors of node  $i$ , and  $\Delta \mathbf{x}_{ik}$  the direction of the pair  $ik$ . It is useful to define the unit pair direction  $\mathbf{e}_{ik} = \Delta \mathbf{x}_{ik} / \Delta x_{ik}$ , which will be used in the following sections. More details about the high-order finite-volume framework used can be found in [41].

Using the divergence theorem, the divergence of a flux  $\mathbf{F}$  writes:

$$\nabla \cdot \mathbf{F}|_i = \frac{1}{V_i} \sum_{k \in \mathcal{N}_i} \mathbf{F}_{ik} \cdot \mathbf{dA}_{ik} \quad (24)$$

with  $\mathbf{F}_{ik}$  the pair-based flux computed as in the gradient to either second or fourth order.

### 3.1.4 Interface normals and curvature computation

The interface normals and curvature are computed in the narrow band around the interface, using the reconstructed signed-distance function  $\phi$  from the GPMM algorithm presented above.

The interface normals are pair-based, and computed as:

$$\mathbf{n}_{ik} = \nabla \phi_{\text{GPMM}}|_{ik}^{4\text{th}} = (\nabla \phi_{\text{GPMM}}|_i^{4\text{th}} + \nabla \phi_{\text{GPMM}}|_k^{4\text{th}}) / 2 \quad (25)$$

where  $\nabla \phi_{\text{GPMM}}|_{i \text{ or } k}^{4\text{th}}$  are fourth-order nodal gradients, computed using Eq. (23).

The interface curvature  $\kappa_\Gamma$  is computed in two steps: (i) the curvature is first estimated at nodes ( $\kappa_i$ ) in the narrow band using Eq. (8); (ii) for the node pairs crossing the interface ( $b_i b_k = -1$ ), a linear interpolation of the nodal curvature radii  $R_i = 1/\kappa_i$  is performed to obtain  $\kappa_\Gamma$ .

**Computation of curvature at nodes** The local mean curvature at a node  $i$  is given by discretizing Eq. (8):

$$\kappa_i = \frac{\text{Tr}(\mathcal{H}_i(\phi_{\text{GPMM}})) - \frac{\nabla \phi_{\text{GPMM}}^T|_i^{2\text{nd}}}{|\nabla \phi_{\text{GPMM}}|_i^{2\text{nd}}} \cdot \mathcal{H}_i(\phi_{\text{GPMM}}) \cdot \frac{\nabla \phi_{\text{GPMM}}|_i^{2\text{nd}}}{|\nabla \phi_{\text{GPMM}}|_i^{2\text{nd}}}}{|\nabla \phi_{\text{GPMM}}|_i^{2\text{nd}}} \quad (26)$$

where  $\mathcal{H}_i(\phi_{\text{GPMM}})$  is the discretized Hessian operator at node  $i$  applied to the reconstructed signed-distance function  $\phi_{\text{GPMM}}$ . Two discretizations of the Hessian operator are studied in this work: compact and non-compact. The compact Hessian computation necessitates only neighboring nodes (i.e. a compact stencil), and is written as:

$$\mathcal{H}_i^{\text{C}}(\phi_{\text{GPMM}}) = \frac{1}{V_i} \sum_{k \in \mathcal{N}_i} \left[ \underbrace{\frac{\phi_{\text{GPMM},k} - \phi_{\text{GPMM},i}}{\Delta x_{ik}} \mathbf{e}_{ik}}_{\text{Aligned with } ik} + \underbrace{\nabla \phi_{\text{GPMM}}|_{ik}^{2\text{nd}} - (\nabla \phi_{\text{GPMM}}|_{ik}^{2\text{nd}} \cdot \mathbf{e}_{ik}) \mathbf{e}_{ik}}_{\text{Orthogonal to } ik} \right] \otimes \mathbf{dA}_{ik} \quad (27)$$

whereas the non-compact formulation is simply obtained by applying two consecutive second-order gradients, leading to a non-compact stencil:

$$\mathcal{H}_i^{\text{NC}}(\phi_{\text{GPMM}}) = \nabla \left( \nabla \phi_{\text{GPMM}}|_i^{2\text{nd}} \right) \Big|_i^{2\text{nd}} \quad (28)$$

**Interpolation for crossing pairs** Once the nodal curvatures are computed, a linear interpolation of the curvature radii is performed for the pairs  $ik$  crossing the interface  $\Gamma$ , resulting in a pair-based interface curvature:

$$\kappa_{\Gamma,ik} = \frac{1}{\theta R_i + (1-\theta)R_k} = \frac{\kappa_i \kappa_k}{\theta \kappa_k + (1-\theta)\kappa_i} \quad (29)$$

with  $\theta$  the interpolation weight ( $0 < \theta < 1$ ), defined in Eq. (15), representing the position of the interface on the crossing pair  $ik$ .

### 3.1.5 Discretization of the reinitialization equation

The reformulated reinitialization (Eq. (7)) is discretized using Eq. (24):

$$\frac{\psi_i^{n^*+1} - \psi_i^{n^*}}{\Delta\tau} = \frac{1}{V_i} \sum_{k \in \mathcal{N}_i} \left( \frac{1}{4 \cosh^2 \left( \frac{\phi_{\text{map},ik}^{n^*}}{2\varepsilon_{ik}} \right)} \left( \nabla \phi_{\text{map}}|_{ik}^{n^*} \cdot \mathbf{n}_{ik} - \mathbf{n}_{ik} \cdot \mathbf{n}_{ik} \right) \right) \mathbf{n}_{ik} \cdot d\mathbf{A}_{ik} \quad (30)$$

where  $\phi_{\text{map},ik} = (\phi_{\text{map},i} + \phi_{\text{map},k})/2$  and  $\varepsilon_{ik} = (\varepsilon_i + \varepsilon_k)/2$  are linearly interpolated interface distance and profile thickness, respectively.  $\nabla \phi_{\text{map}}|_{ik}$  is computed using a compact term in the direction of the node pair as in Eq. (27):

$$\nabla \phi_{\text{map}}|_{ik} = \frac{\phi_{\text{map},k} - \phi_{\text{map},i}}{\Delta x_{ik}} \mathbf{e}_{ik} + \frac{\nabla \phi_{\text{map}}|_i^{2\text{nd}} + \nabla \phi_{\text{map}}|_k^{2\text{nd}}}{2} - \left( \frac{\nabla \phi_{\text{map}}|_i^{2\text{nd}} + \nabla \phi_{\text{map}}|_k^{2\text{nd}}}{2} \cdot \mathbf{e}_{ik} \right) \mathbf{e}_{ik} \quad (31)$$

where  $\nabla \phi_{\text{map}}|_i^{2\text{nd}}$  are second-order nodal gradients computed using Eq. (22).

The interface normals  $\mathbf{n}_{ik}$  are computed with Eq. (25). It has been chosen to use non-unit normals, to avoid spurious gradient orientations when several interfaces are close to each other, and thus improve the front-merging properties of the implemented reinitialization.

Special attention has to be paid in regions of the narrow band where  $\psi$  is close to 0 and 1. In these regions,  $\phi_{\text{map}}$  is difficult to estimate and two improvements in the reinitialization algorithm are introduced: (i) in order to ensure a correct computation of  $\nabla \phi_{\text{map}}|_{ik}$ , which requires several nodal values of  $\phi_{\text{map}}$ , an under/overshoot tag is attributed to all pairs with at least one node concerned by  $\psi$  falling outside  $]0, 1[$ ; (ii) the resharpener term is removed so that the reinitialization becomes purely diffusive. Using the original isotropic diffusion term of Olsson and Kreiss [23] in this case, the reinitialization then takes the form:

$$\frac{\psi_i^{n^*+1} - \psi_i^{n^*}}{\Delta\tau} = \frac{1}{V_i} \sum_{k \in \mathcal{N}_i} \varepsilon_{ik} \nabla \psi|_{ik}^{n^*} \cdot d\mathbf{A}_{ik} \quad (32)$$

with  $\nabla \psi|_{ik}^{n^*}$  discretized as in Eq. (31). Once the overshoots or undershoots of  $\psi$  disappear, the reinitialization turns back to its main form given by Eq. (30).

Usually, 3 steps of reinitialization are performed and the pseudo-timestep of each reinitialization step is simply determined by a Fourier condition on the diffusive part of the reinitialization. This stability condition leads to:

$$\Delta\tau = \min_{\text{pairs } ik} \left( \frac{\text{Fo} \cdot \Delta x_{ik}^2}{\varepsilon_{ik}} \right) \quad (33)$$

where  $\Delta x_{ik}$  stands for the length of the pair  $ik$ , and Fo is the Fourier number set to 0.5.

### 3.1.6 Transport of the ACLS function - Time integration

The ACLS function  $\psi$  is advected using Eq. (4). In YALES2, a 4th-order discretization is performed for convective terms, leading to the following semi-discrete equation for a node  $i$ :

$$\frac{\partial \psi_i}{\partial t} + \nabla \cdot (\psi \mathbf{u})|_i = 0 \quad (34)$$

Then by applying Eq. (24):

$$\frac{\partial \psi_i}{\partial t} + \frac{1}{V_i} \sum_{k \in \mathcal{N}_i} \psi_{ik} \mathbf{u}_{ik} \cdot d\mathbf{A}_{ik} = 0 \quad (35)$$

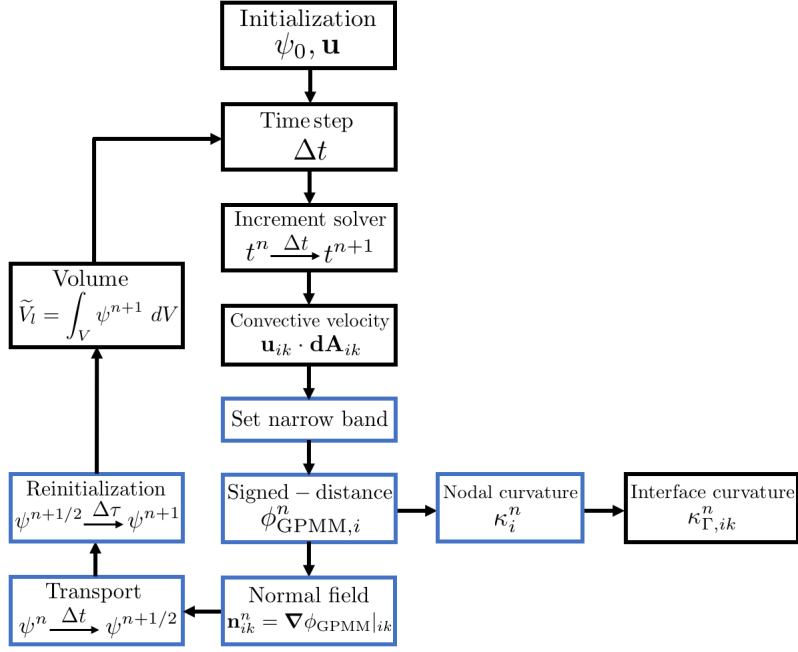


Figure 6: Full ACLS procedure: the blue boxes represent computations in the narrow band.

Using a 4th-order spatial discretization as in Section 3.1.3, Eq. (35) yields:

$$\frac{\partial \psi_i}{\partial t} + \frac{1}{V_i} \sum_{k \in \mathcal{N}_i} \left( \frac{\psi_i + \psi_k}{2} + \frac{\nabla \psi_i^{\text{2nd}} - \nabla \psi_k^{\text{2nd}}}{6} \cdot \Delta \mathbf{x}_{ik} \right) \left( \frac{\mathbf{u}_i + \mathbf{u}_k}{2} + \frac{\nabla \mathbf{u}_i^{\text{2nd}} - \nabla \mathbf{u}_k^{\text{2nd}}}{6} \cdot \Delta \mathbf{x}_{ik} \right) \cdot d\mathbf{A}_{ik} = 0 \quad (36)$$

Time integration is performed using the TFV4A scheme of [42], which is a 4th-order, Runge-Kutta-like, time-marching scheme. The timestep  $\Delta t$  is computed from the classical explicit CFL stability criterion:

$$\Delta t = \min_{\text{pairs } ik} \left( \frac{\text{CFL} |\mathbf{dA}_{ik}| \Delta x_{ik}}{|\mathbf{u}_{ik} \cdot \mathbf{dA}_{ik}|} \right) \quad (37)$$

where  $\Delta x_{ik}$  is the length of the pair  $ik$ . The full ACLS procedure is depicted in Fig. 6.

## 3.2 Low-Mach flow solver

### 3.2.1 Ghost-Fluid Method

The Ghost-Fluid Method [25, 38] is a robust and accurate way to explicitly deal with jump conditions. Let us consider the jump at a liquid-gas interface in one dimension of a physical quantity  $\zeta$ , written  $[\zeta]_\Gamma = [\zeta](x_\Gamma) = \zeta_l(x_\Gamma) - \zeta_g(x_\Gamma) = \zeta_{l,\Gamma} - \zeta_{g,\Gamma}$ . The liquid and gas quantities in each phase can be extended in the other phase by continuity:

$$\zeta_l(x) = \zeta_{l,\Gamma} + (x - x_\Gamma) \frac{\partial \zeta}{\partial x} \Big|_{l,\Gamma} + \mathcal{O}(x - x_\Gamma)^2 \quad \text{and} \quad \zeta_g(x) = \zeta_{g,\Gamma} + (x - x_\Gamma) \frac{\partial \zeta}{\partial x} \Big|_{g,\Gamma} + \mathcal{O}(x - x_\Gamma)^2 \quad (38)$$

which leads to the expression of the jump everywhere in the fluid domain:

$$[\zeta](x) = [\zeta]_\Gamma + (x - x_\Gamma) \left[ \frac{\partial \zeta}{\partial x} \right]_\Gamma + \mathcal{O}(x - x_\Gamma)^2 \quad (39)$$

The extension to 3D is straightforward:

$$[\zeta](\mathbf{x}) = [\zeta]_\Gamma + (\mathbf{x} - \mathbf{x}_\Gamma) \cdot [\nabla \zeta]_\Gamma + \mathcal{O}(\mathbf{x} - \mathbf{x}_\Gamma)^2 \quad (40)$$

Fig. 7 illustrates the principle of the Ghost-Fluid Method. In the following, this methodology is used to discretize the pressure gradient in the numerical framework of this study: pair-based finite-volume schemes.

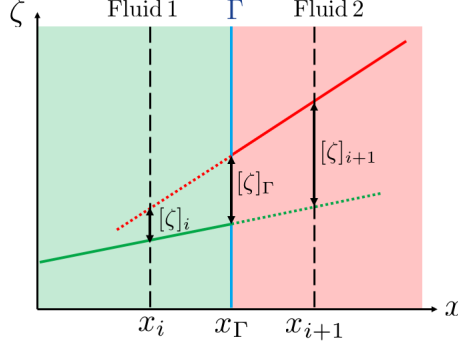


Figure 7: Ghost-Fluid Method: extension of the jump of a physical quantity  $\zeta$ .

**Discretization of the pressure gradient** A first-order finite-difference approximation of the pressure term gives:

$$\left. \frac{1}{\rho} \frac{\partial p}{\partial x} \right|_{l,i+\frac{1}{2}} = \frac{1}{\rho_l} \frac{p_{l,i+1} - p_{l,i}}{\Delta x} + \mathcal{O}(\Delta x) \quad (41)$$

Using the pressure jump at  $x_i$ , the liquid pressure is  $p_{l,i} = [p]_i + p_{g,i}$  and can be replaced as follows:

$$\left. \frac{1}{\rho} \frac{\partial p}{\partial x} \right|_{l,i+\frac{1}{2}} = \frac{1}{\rho_l} \frac{p_{l,i+1} - [p]_i - p_{g,i}}{\Delta x} + \mathcal{O}(\Delta x) \quad (42)$$

The expression for the pressure jump  $[p]_i$  is given by Eq. (39):

$$[p]_i = [p]_\Gamma + (x_i - x_\Gamma) \left[ \frac{\partial p}{\partial x} \right]_\Gamma + \mathcal{O}(x_i - x_\Gamma)^2 \quad (43)$$

$[p]_\Gamma$  is a known quantity (Eq. (11)) but  $[\partial p / \partial x]_\Gamma$  is unknown a priori. With the assumption that the velocity is continuous across the interface, the pressure term in the momentum equation Eq. (10) should also be continuous:

$$\left[ \frac{1}{\rho} \frac{\partial p}{\partial x} \right]_\Gamma = 0 \iff \frac{1}{\rho_l} \frac{\partial p}{\partial x} \Big|_{l,\Gamma} = \frac{1}{\rho_g} \frac{\partial p}{\partial x} \Big|_{g,\Gamma} \quad (44)$$

After many developments, which can be found in [25], the final form of the discretized pressure gradient for finite differences is obtained, for  $\Gamma \in [x_i; x_{i+1}]$ :

$$\left. \frac{1}{\rho} \frac{\partial p}{\partial x} \right|_{l,i+\frac{1}{2}} = \left. \frac{1}{\rho} \frac{\partial p}{\partial x} \right|_{g,i+\frac{1}{2}} = \frac{1}{\rho^*} \frac{p_{l,i+1} - p_{g,i}}{\Delta x} - \frac{1}{\rho^*} \frac{[p]_\Gamma}{\Delta x} + \mathcal{O}(\Delta x) \quad (45)$$

with  $\rho^* = \theta \rho_g + (1 - \theta) \rho_l$  a modified density, computed using the interpolation weight defined in Eq. (15).

**Implementation in the Poisson solver** YALES2 is a pair-based finite-volume solver, as a result implementation of finite-difference schemes can be performed on the pairs. The expression of the discretized pressure term of the momentum equation written in Eq. (45) is integrated in the Poisson solver of YALES2 to account for interfacial jump conditions.

To solve the incompressible Navier-Stokes equations, a projection method is employed [43]. The Poisson equation, used to enforce the solenoidal nature of the velocity field  $\mathbf{u}^{n+1}$ , is:

$$\nabla \cdot \left( \frac{1}{\rho} \nabla p \right) = \frac{1}{\Delta t} \nabla \cdot \mathbf{u}^* \quad (46)$$

with  $\mathbf{u}^*$  the predicted velocity, estimated from  $\mathbf{u}^n$  (see next section 3.2.2). Eq. (46) can be discretized as presented in Section 3.1.3:

$$\sum_{k \in \mathcal{N}_i} \frac{1}{\rho} \nabla p \Big|_{ik} \cdot \mathbf{dA}_{ik} = \frac{1}{\Delta t} \sum_{k \in \mathcal{N}_i} \mathbf{u}_{ik}^* \cdot \mathbf{dA}_{ik} \quad (47)$$

in which the pressure gradient is computed at pairs. A linear system of the form  $Ap = b$  that includes the jump conditions needs to be solved, and is written as:

$$\underbrace{\sum_{k \in \mathcal{N}_i} \frac{1}{\rho^*} \frac{p_k - p_i}{\Delta x_{ik}} \mathbf{e}_{ik} \cdot \mathbf{dA}_{ik}}_{Ap} = \underbrace{\frac{1}{\Delta t} \sum_{k \in \mathcal{N}_i} \mathbf{u}_{ik}^* \cdot \mathbf{dA}_{ik} + \sum_{k \in \mathcal{N}_i} \frac{1}{\rho^*} \frac{[p]_\Gamma}{\Delta x_{ik}} \mathbf{e}_{ik} \cdot \mathbf{dA}_{ik}}_b \quad (48)$$



The second part of the RHS  $b$ , which contains the pressure jump, is added for all pairs  $ik$  crossing the interface. The linear system is solved using the Deflated Preconditioned Conjugate Gradient of [39].

The correction step is written as:

$$\frac{\mathbf{u}^{n+1} - \mathbf{u}^*}{\Delta t} = -\frac{1}{\rho} \nabla p \Big|_i \quad (49)$$

in which the pressure gradient is calculated at nodes. For a pair of nodes which does not cross the interface (i.e. within one incompressible phase, liquid or gas), the pressure gradient is simply discretized as in Section 3.1.3, with a constant density  $\rho$  corresponding to the phase. For a crossing pair  $ik$ , the jump conditions need to be accounted for:

$$\frac{1}{\rho} \nabla p \Big|_i^{2\text{nd}} = \frac{1}{\rho_i V_i} \sum_{k \in \mathcal{N}_i} \frac{p_i + p_k + [p]_k}{2} \mathbf{dA}_{ik} \quad (50)$$

where the pressure jump is expressed from the developments above using the Ghost-Fluid Method:

$$[p]_k = \frac{\rho_i}{\rho^*} [p]_\Gamma + \left(1 - \frac{\rho_i}{\rho^*}\right) (p_i - p_k) \quad (51)$$

### 3.2.2 Velocity prediction with momentum correction

In order to avoid velocity mixing between liquid and gas, and thus limit spurious kinetic energy transfers at the interface, Desjardins and Moureau introduced a momentum correction at the interface in [44]. In this section, we only consider the convection term in Eq. (10) for convenience. The prediction step writes:

$$\frac{\mathbf{u}^* - \mathbf{u}^n}{\Delta t} = -\nabla \cdot (\mathbf{u}^{n+1/2} \otimes \mathbf{u}^{n+1/2}) \quad (52)$$

which can be rewritten in terms of momentum:

$$\frac{\rho^{**} \mathbf{u}^* - \rho^n \mathbf{u}^n}{\Delta t} = -\nabla \cdot (\rho^{n+1/2} \mathbf{u}^{n+1/2} \otimes \mathbf{u}^{n+1/2}) \quad (53)$$

where  $\rho^{**}$  is a predicted density computed from the following step:

$$\frac{\rho^{**} - \rho^n}{\Delta t} = -\nabla \cdot (\rho^{n+1/2} \mathbf{u}^{n+1/2}) \quad (54)$$

Eq. (54) can be discretized in space as:

$$\frac{\rho_i^{**} - \rho_i^n}{\Delta t} = -\frac{1}{V_i} \sum_{k \in \mathcal{N}_i} \rho_{ik}^{n+1/2} \mathbf{u}_{ik} \cdot \mathbf{dA}_{ik} \quad (55)$$

with  $\rho_{ik}^{n+1/2}$  computed using a simple upwinded scheme to ensure the boundedness of the density flux at the interface:

$$\rho_{ik}^{n+1/2} = \begin{cases} (1 - \mathcal{U}_{ik}) \frac{\rho_i^n + \rho_k^n}{2} + \mathcal{U}_{ik} \rho_i^n & \text{if } \mathbf{u}_{ik} \cdot \mathbf{dA}_{ik} \geq 0 \\ (1 - \mathcal{U}_{ik}) \frac{\rho_i^n + \rho_k^n}{2} + \mathcal{U}_{ik} \rho_k^n & \text{if } \mathbf{u}_{ik} \cdot \mathbf{dA}_{ik} < 0 \end{cases} \quad (56)$$

where the pair-based limiter  $\mathcal{U}$  is computed as:

$$\mathcal{U}_{ik} = \max \left( \frac{|b|_{\text{upw}} - \min(|b_i|, |b_k|) + 1}{|b|_{\text{upw}}}, 0 \right) \quad (57)$$

with  $|b|_{\text{upw}} = 5$ , meaning that the first five narrow-band levels from the interface are concerned by the upwinding. A similar numerical strategy is applied for the conservative flux of Eq. (53):

$$\frac{\rho_i^{**} \mathbf{u}_i^* - \rho_i^n \mathbf{u}_i^n}{\Delta t} = -\frac{1}{V_i} \sum_{k \in \mathcal{N}_i} \rho_{ik}^{n+1/2} \mathbf{u}_{ik} \cdot \mathbf{dA}_{ik} \quad (58)$$

with  $\rho_{ik}^{n+1/2}$  calculated to 4th-order as:

$$\rho_{ik}^{n+1/2} = \begin{cases} (1 - \mathcal{U}_{ik}) \frac{\rho_i^n \mathbf{u}_i^n + \rho_k^n \mathbf{u}_k^n}{2} + \mathcal{U}_{ik} \rho_i^n \mathbf{u}_i^n + \rho_{ik}^{n+1/2} \frac{\nabla \mathbf{u}_i^{2\text{nd}} - \nabla \mathbf{u}_k^{2\text{nd}}}{6} \cdot \Delta \mathbf{x}_{ik} & \text{if } \mathbf{u}_{ik} \cdot \mathbf{dA}_{ik} \geq 0 \\ (1 - \mathcal{U}_{ik}) \frac{\rho_i^n \mathbf{u}_i^n + \rho_k^n \mathbf{u}_k^n}{2} + \mathcal{U}_{ik} \rho_k^n \mathbf{u}_k^n + \rho_{ik}^{n+1/2} \frac{\nabla \mathbf{u}_i^{2\text{nd}} - \nabla \mathbf{u}_k^{2\text{nd}}}{6} \cdot \Delta \mathbf{x}_{ik} & \text{if } \mathbf{u}_{ik} \cdot \mathbf{dA}_{ik} < 0 \end{cases} \quad (59)$$

### 3.2.3 Time integration - Full solution procedure

As in Section 3.1.6, time integration is performed using the TFV4A scheme of [42], which is a 4th-order, Runge-Kutta-like, time-marching scheme. The timestep  $\Delta t$  is computed from the explicit capillary constraint:

$$\Delta t = \min_{\text{crossing pairs } ik} \sqrt{\frac{\text{STN}^2 \Delta x_{ik}^2 \min(\rho_g, \rho_l)}{\sigma |\kappa_{\Gamma, ik}|}} \quad (60)$$

where STN is the surface tension number, usually set to 0.5.

The full solution procedure is here summarized:

- (i) Using the unstructured ACLS algorithm, advance the interface implicitly through the  $\psi$  field with Eq. (36) using  $\mathbf{u}^n$ , then reinitialize  $\psi$  to reform the hyperbolic tangent profile with Eq. (30).
- (ii) Set the new narrow band around the interface (Section 3.1.1).
- (iii) Compute interface features in the narrow band: geometric signed distance  $\phi_{\text{GPMM}}$  (Section 3.1.2), normals  $\mathbf{n}$  (Eq. (25)), curvature  $\kappa_{\Gamma}$  (Eq. (26) then Eq. (29)).
- (iv) Predict the velocity field  $\mathbf{u}^*$  from  $\mathbf{u}^n$ , using the momentum correction at the interface of [44] adapted to unstructured meshes (Section 3.2.2).
- (v) Project the velocity field by solving the Poisson equation, in which interface jump conditions are treated with the Ghost-Fluid Method (Eq. (48)).
- (vi) Correct the velocity, with the pressure gradient that includes the jump conditions using the Ghost-Fluid Method (Eq. (49)), eventually leading to  $\mathbf{u}^{n+1}$  so that  $\nabla \cdot \mathbf{u}^{n+1} = 0$ .
- (vii) Grid adaptation (see next section 3.3).

### 3.3 Isotropic dynamic mesh adaptation strategy

To resolve the small physical scales at the interface on tetrahedral grids at a moderate cost, an isotropic Adaptive Mesh Refinement (AMR) technique is used [45], employing the MMG library [46]. Special attention is needed when coupling AMR with ACLS to avoid liquid mass loss, so that the interface never encounters cell-size gradients. Thus, the mesh is refined around the interface based on the distance  $\mathcal{D} = |\phi|$  to it, so that  $\Gamma$  always stays in a protected region of constant cell size  $\Delta x_{\min}$  (Fig. 8). It is important to stress that this metric evolution ensures that the remeshing process always occurs upwind and downwind of the interface but never at the interface itself. This strategy avoids any interpolation error at the interface location.

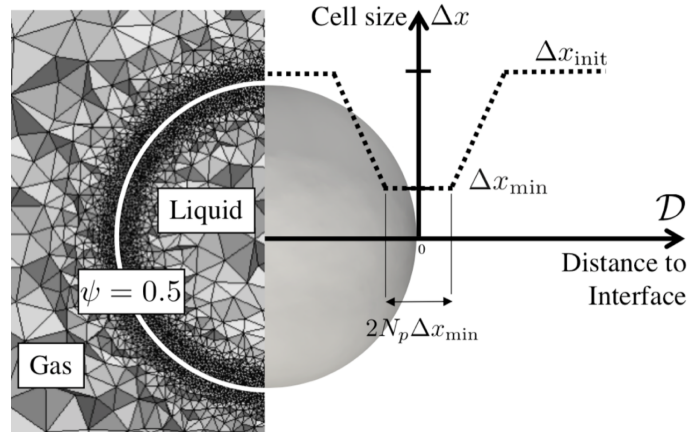


Figure 8: Adaptive mesh refinement strategy: refinement based on the distance  $\mathcal{D}$  to the interface.

The width of this region is  $2N_p\Delta x_{\min}$ , with  $N_p$  a user-defined parameter, usually set between 6 and 12. Away from this area, the metric evolution follows a linear law until it reaches the maximum cell size  $\Delta x_{\text{init}}$ , allowing only smooth and controlled cell-size gradients in the domain. The adaptation process is triggered automatically when the interface approaches the edge of the protected region, leading to considerable computational savings in remeshing.

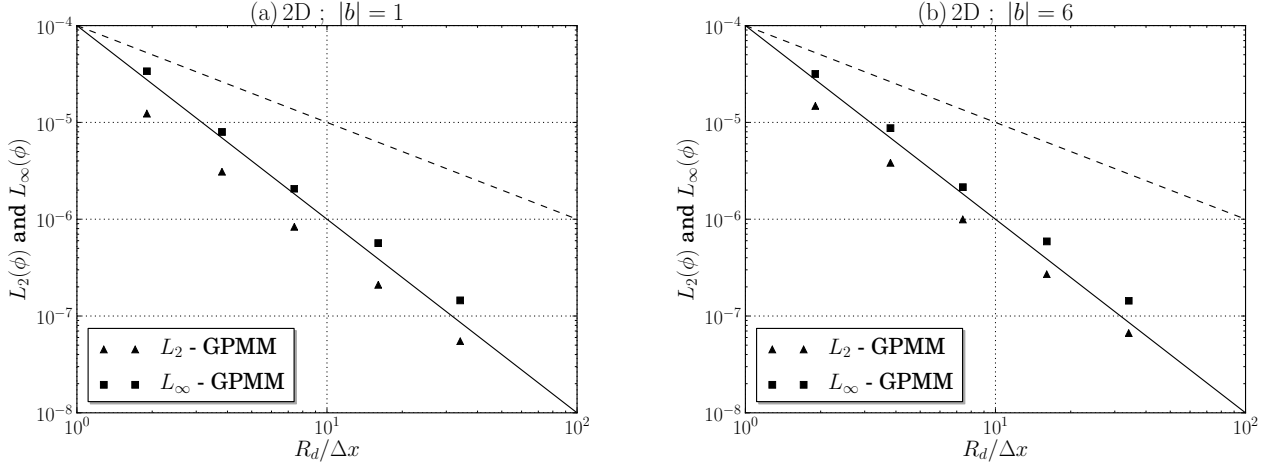


Figure 9:  $L_2$  and  $L_\infty$  of the reconstructed signed-distance function  $\phi$  with the 2D GPMM algorithm (Geometric-Projection Marker Method), in first (a) and sixth (b) narrow-band levels  $|b|$ . The dashed and solid lines are first and second-order convergence, respectively.

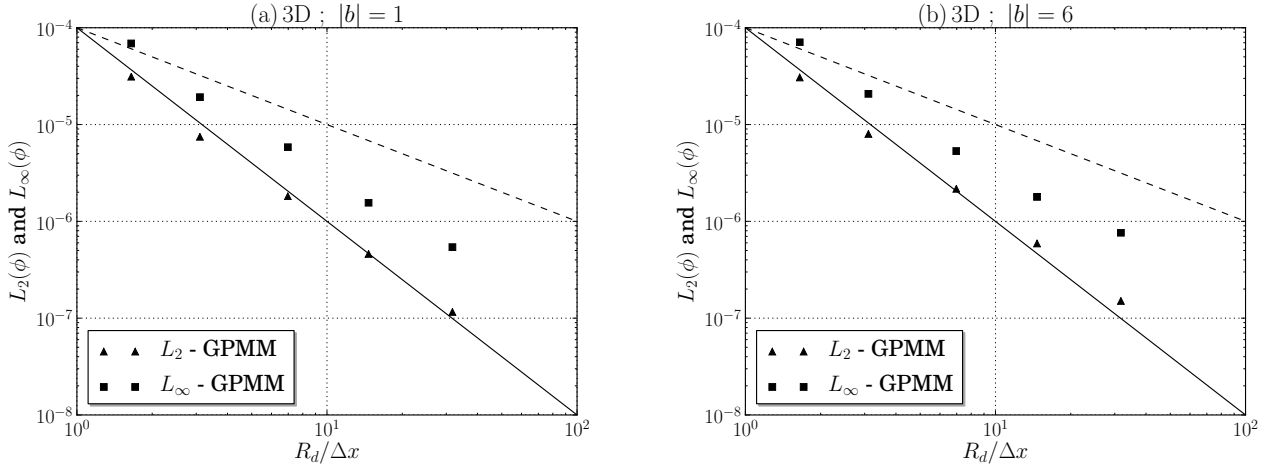


Figure 10:  $L_2$  and  $L_\infty$  of the reconstructed signed-distance function  $\phi$  with the 3D GPMM algorithm (Geometric-Projection Marker Method), in first (a) and sixth (b) narrow-band levels  $|b|$ .

## 4 Validation on unstructured grids

### 4.1 Accuracy of signed-distance and curvature computation

A 2D/3D static droplet of radius  $R_d$  is initialized on homogeneous triangular/tetrahedral grids to assess the accuracy of the GPMM procedure, both in 2D and 3D, for the signed-distance reconstruction on irregular meshes. Errors are calculated as follows:

$$L_2(\phi) = \sqrt{\frac{1}{N_{i \in |b|}} \sum_{i=1}^{N_{i \in |b|}} (\phi_{\text{th},i} - \phi_{\text{GPMM},i})^2} \quad \text{and} \quad L_\infty(\phi) = \max_{i \in |b|} |\phi_{\text{th},i} - \phi_{\text{GPMM},i}| \quad (61)$$

where  $|b|$  is the absolute narrow-band level,  $\phi_{\text{th},i}$  is the theoretical distance at node  $i$  to the circular/spherical interface, and  $N_{i \in |b|}$  is the total number of nodes in the corresponding absolute narrow-band level. The calculations of  $L_2$  and  $L_\infty$  are performed for nodes belonging to edges crossed by the interface ( $|b| = 1$ ) and in the sixth band level ( $|b| = 6$ , far from the interface). As homogeneous triangles/tetrahedra are used to tessellate the domain, the mesh cell size is computed as  $\Delta x = \sqrt{V_i}$  in 2D, and as  $\Delta x = \sqrt[3]{V_i}$  in 3D, where  $V_i$  is the control volume based on the dual mesh around a node  $i$ . Results are shown in Fig. 9 and Fig. 10 for the 2D and 3D cases, respectively.

It can be seen in Fig. 9 (a) that in the first band level ( $|b| = 1$ ), i.e. close to the interface, the GPMM procedure converges with second-order accuracy ( $L_2$  and  $L_\infty$ ) with low error values, which is satisfactory for

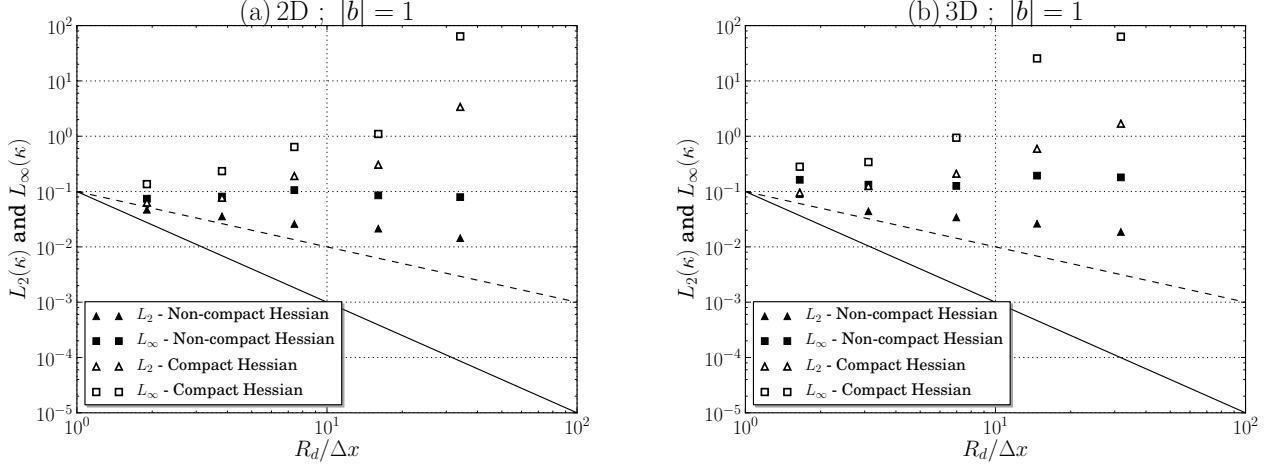


Figure 11:  $L_2$  and  $L_\infty$  norms of the 2D (a) and 3D (b) interface curvature  $\kappa_\Gamma$ , considering two spatial discretizations of the Hessian operator: with compact and non-compact stencils.

triangular grids. Fig. 9 (b) shows the results for the sixth band level ( $|b| = 6$ ), i.e. far from the interface. Again, as the resolution of the droplet increases, both  $L_2$  and  $L_\infty$  errors exhibit second-order convergence. A similar behavior is displayed for 3D tetrahedral meshes in Fig. 10. As several markers are propagated in the narrow band ( $N_M = 3$  in 2D and  $N_M = 10$  in 3D, see Section 3.1.2), a sufficient amount of data is available at a node far from the interface to compute the projected distance to the closest surface element. The advantage of the GPMM algorithm, which is an exact geometric projection, is that the numerical errors are only induced by segmentation (2D) or triangulation (3D): a perfect line (2D) or plane (3D) on an unstructured mesh lead to a distance computed within machine accuracy.

On the same test case, the accuracy of the interface curvature  $\kappa_\Gamma$  (Eq. (29)) is evaluated in 2D and 3D using the following error norms:

$$L_2(\kappa) = \frac{1}{\kappa_{\Gamma, \text{th}}} \sqrt{\frac{1}{N_{\text{cp}}} \sum_{ik=1}^{N_{\text{cp}}} (\kappa_{\Gamma, \text{th}} - \kappa_{\Gamma, ik})^2} \quad \text{and} \quad L_\infty(\kappa) = \frac{1}{\kappa_{\Gamma, \text{th}}} \max_{\text{cp}} |\kappa_{\Gamma, \text{th}} - \kappa_{\Gamma, ik}| \quad (62)$$

where  $N_{\text{cp}}$  stands for the total number of node pairs crossing the interface  $\Gamma$ ,  $\kappa_{\Gamma, \text{th}}$  is the constant theoretical curvature of the circle/sphere, and  $\kappa_{\Gamma, ik}$  is the pair-based interface curvature, for a node pair  $ik$  among the crossing pairs, obtained with Eq. (29). Results are shown in Fig. 11, for the two different discretizations of the Hessian operator  $\mathcal{H}(\phi_{\text{GPMM}})$  in Eq. (26). The non-compact discretization consists in applying two consecutive second-order gradients to the reconstructed signed-distance function  $\phi_{\text{GPMM}}$  (Eq. (28)), leading to a non-compact stencil, whereas a smaller stencil is used in the compact formulation: only direct-neighbor points are used to compute the Hessian (Eq. (27)). Slight convergence of the  $L_2$ -norm is observed for the 2D and 3D interface curvatures computed using the non-compact Hessian matrix. However, using a compact stencil leads to divergence of the curvature for both  $L_2$  and  $L_\infty$  norms, in 2D and 3D. As a consequence, the non-compact formulation will be used in the following.

## 4.2 Interface transport tests

In order to assess the accuracy, spatial convergence and robustness of the ACLS procedure, classical interface transport tests are performed.

### 4.2.1 Static sphere on static tetrahedral grids

A 3D static sphere is initialized on a homogeneous tetrahedral mesh. The two reinitialization equations Eq. (5) [25] and Eq. (7) [27] are compared at two grid resolutions: 6 and 12 points in the sphere radius, and each of them is performed 50,000 times throughout the runs. As homogeneous tetrahedra are used to tessellate the domain, the mesh cell size is computed as  $\Delta x = \sqrt[3]{V_i}$ , where  $V_i$  is the control volume based on the dual mesh around a node  $i$ . Results are presented in Fig. 12. As presented in [27] for cartesian meshes, it is observed that the reformulated reinitialization Eq. (7) induces much less artificial deformation of the interface at high levels of reinitialization, here on tetrahedral meshes. Thus, the conclusion of [27] can be extended to unstructured meshes.

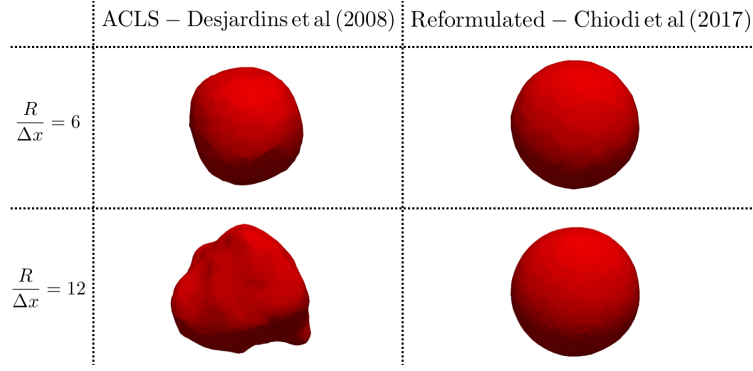


Figure 12: 3D static sphere transport test on two tetrahedral grids: comparison between Eq. (5) (original ACLS reinitialization) and Eq. (7) (reformulated reinitialization) after 50,000 iterations of reinitialization.

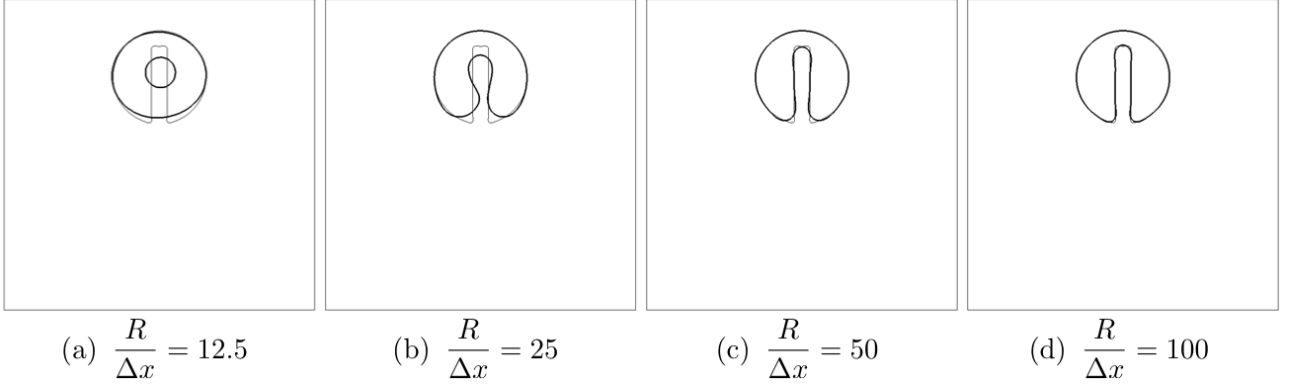


Figure 13: Interface contours after 10 rotations for various mesh resolutions. The thin line represents a simulation performed with 400 points in the disk radius.

#### 4.2.2 Rotation of a notched disk on static triangular grids

In order to assess the robustness of the methods presented above for the transport of interfaces with topology that includes sharp edges and corners, the classical notched disk rotation case, introduced by S.T. Zalesak in [47], is first run on homogeneous triangular grids. A two-dimensional disk of radius  $R = 0.15$ , with a notch of width  $w = 0.05$ , is initialized at  $(x, y) = (0, 0.25)$  in a  $[-0.5; 0.5]^2$  square. Given the angular velocity  $\Omega = 1\text{rad/s}$  and the origin of the domain as the center of rotation  $(x_0, y_0) = (0, 0)$ , the velocity field is:

$$\mathbf{u}(x, y) = \begin{pmatrix} -\Omega(y - y_0) \\ \Omega(x - x_0) \end{pmatrix} = \begin{pmatrix} -y \\ x \end{pmatrix} \quad (63)$$

Thus, the disk performs one rotation about the origin in  $T = 2\pi/\Omega = 2\pi\text{s}$ . In the present paragraph, 10 rotations are done on triangular meshes with a CFL of 0.48, which in theory should not change the shape of the slotted disk. Three reinitialization steps are done at each temporal iteration and  $\varepsilon = \Delta x/2$ . The narrow band is of width  $|b|_{\max} = 12$  to ensure that  $|\nabla\psi| = 0$  at the edge of the band and thus better preserve mass.

Final interface contours after 10 rotations are presented in Fig. 13 for various mesh resolutions. The results are superimposed over a highly-resolved simulation (400 points in the disk radius). Good mesh convergence properties for interface dynamics are demonstrated: at the highest resolution shown (100 points in the disk radius), the initial shape is well retrieved. Fig. 14 shows two scalar fields, namely the signed narrow-band levels  $b$  and the reconstructed signed-distance function using the GPMM algorithm  $\phi_{\text{GPMM}}$ , after 10 rotations for  $R/\Delta x = 50$ . These fields are seen to be smooth and well computed. Fig. 15 depicts the node normal vector field around the interface in the same computational conditions: as detailed in Section 3.1.2, computing the normals from a signed-distance function avoids spurious normal orientation.

To quantitatively assess the accuracy of the method, we use a methodology inspired by [48] to compute boundedness, shape, and mass errors at the end of the runs  $t = T$ :

$$E_{\text{bound}}(T) = \max \left( -\min_{i=1 \dots N_{\text{cv}}} (\psi_i(T) V_i), \max_{i=1 \dots N_{\text{cv}}} ((\psi_i(T) - 1) V_i) \right) \quad (64)$$

$$E_{\text{shape}}(T) = \sum_{i=1}^{N_{\text{cv}}} |\psi_i(T) - \psi_i(0)| V_i \quad (65)$$

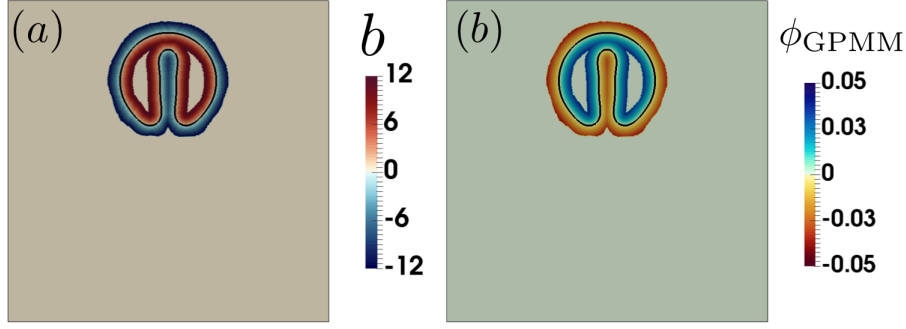


Figure 14: Scalar fields after 10 rotations with  $R/\Delta x = 50$ . (a) Signed narrow-band levels  $b$ ; (b) Reconstructed signed-distance function  $\phi_{\text{GPMM}}$ . Black contours represent the interface.

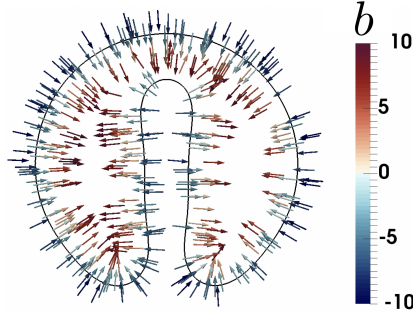


Figure 15: Node normal vector field  $\mathbf{n}_i = \nabla \phi_{\text{GPMM}}|_i^{4\text{th}}$  around the interface after 10 rotations with  $R/\Delta x = 50$ , colored by the signed narrow-band levels  $b$ .

$$E_{\text{mass}}(T) = \left| \sum_{i=1}^{N_{\text{cv}}} \psi_i(T) V_i - \sum_{i=1}^{N_{\text{cv}}} \psi_i(0) V_i \right| \quad \text{and} \quad \text{mass loss [\%]} = \frac{100 \left| \sum_{i=1}^{N_{\text{cv}}} \psi_i(T) V_i - \sum_{i=1}^{N_{\text{cv}}} \psi_i(0) V_i \right|}{\sum_{i=1}^{N_{\text{cv}}} \psi_i(0) V_i} \quad (66)$$

These errors allow the estimation of the largest overshoot/undershoot in the domain, the discrepancies in the final shape compared to the initial shape, and the mass loss throughout the run, respectively. The computational efficiency is measured through a reduced computational time RCT:

$$\text{RCT} = \frac{\text{WCT} \times N_{\text{cores}}}{N_{\text{iter}} \times N_{\text{cv}}} \quad (67)$$

with WCT the Wall Clock Time. The results are presented in Table 2.

Although machine accuracy is not reached, the level of errors in mass remains very low for all resolutions. Moreover, neither undershoot nor overshoot are present at the end of the simulation (10 rotations) for half of the resolutions: some can appear during these runs with an extremely small amplitude but are naturally corrected over time, as the interface topology becomes smoother. The shape error is plotted in Fig. 16: the method exhibits between first and second-order convergence on two-dimensional triangular meshes for this test case. The parallel efficiency of the conservative level set solver is demonstrated through the RCT values, which remain low for all calculations.

Table 2: Error norms for each mesh resolution at the end of the notched disk test, and computational cost of the runs.

$R/\Delta x$	$E_{\text{bound}}(T)$	$E_{\text{shape}}(T)$	$E_{\text{mass}}(T)$	mass loss [%]	$N_{\text{cores}}$	RCT [ $\mu\text{s}$ ]
12.5	$4.562 \times 10^{-10}$	$1.360 \times 10^{-2}$	$1.410 \times 10^{-7}$	$2.406 \times 10^{-4}$	1	3.3
25	0	$7.544 \times 10^{-3}$	$1.158 \times 10^{-7}$	$1.986 \times 10^{-4}$	1	2.9
50	$1.119 \times 10^{-11}$	$2.226 \times 10^{-3}$	$1.736 \times 10^{-9}$	$2.980 \times 10^{-6}$	2	3.5
100	0	$8.427 \times 10^{-4}$	$1.874 \times 10^{-8}$	$3.219 \times 10^{-5}$	8	4.9
200	$1.274 \times 10^{-11}$	$3.434 \times 10^{-4}$	$3.473 \times 10^{-9}$	$5.965 \times 10^{-6}$	28	5.8
400	0	$1.224 \times 10^{-4}$	$3.076 \times 10^{-9}$	$5.284 \times 10^{-6}$	112	5.8

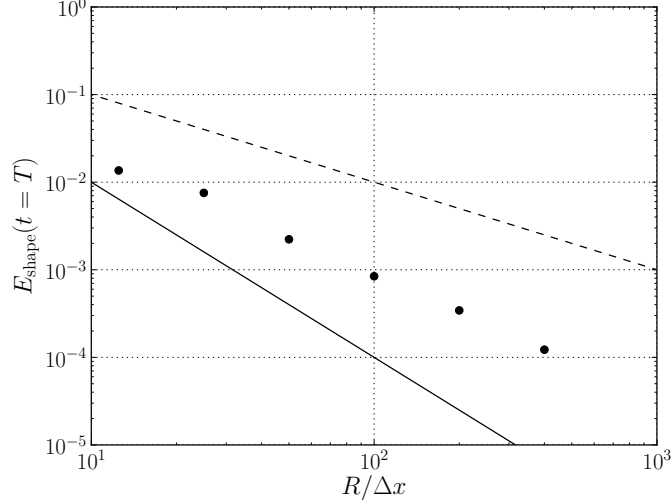


Figure 16: Shape error  $E_{\text{shape}}$  after 10 rotations of the 2D notched disk ( $t = T$ ), for various mesh resolutions: from 12.5 to 400 points in the disk radius. The dashed and solid lines are first and second-order convergence, respectively.

#### 4.2.3 Circle deformation in a vortex on static triangular grids

A two-dimensional deformation of a disk in a vortex is then run [49]. This test is used to evaluate the behaviour of the algorithms at resolution limit, which is useful for two-phase flow simulations, as thin ligaments are present. A disk of radius  $R = 0.15$  is initially centered at  $(x, y) = (0, 0.25)$  in a 2D box of dimensions  $[-0.5; 0.5]^2$ . The disk is deformed by a swirling velocity field:

$$\mathbf{u}(x, y, t) = \begin{pmatrix} -2 \sin^2(\pi x) \sin(\pi y) \cos(\pi y) \cos(\pi t/T) \\ 2 \sin^2(\pi y) \sin(\pi x) \cos(\pi x) \cos(\pi t/T) \end{pmatrix} \quad (68)$$

The disk is first stretched for  $0 \leq t \leq T/2$ , and then unstretched for  $T/2 \leq t \leq T$ . At  $t = T$ , the disk is theoretically back to its original circular shape. The total simulation time is chosen as  $T = 8\text{s}$ . The initial CFL number is 0.64 for all performed simulations of this paragraph. Three reinitialization steps are done at each temporal iteration and  $\varepsilon = \Delta x/2$ . The narrow band is of width  $|b|_{\text{max}} = 12$ .

Maximum and final interface deformation, at  $t = T/2$  and  $t = T$  respectively, are presented in Fig. 17 for various mesh resolutions. The results are superimposed over a highly-resolved simulation (400 points in the disk radius). Good mesh convergence properties are demonstrated: at the highest resolution shown (100 points in the disk radius), the original circular shape is perfectly retrieved. It should also be observed that at resolution limit, no locking phenomenon on the mesh nor artificial surface tension occur: the tail breaks up into small circles.

Using the same methodology than in the previous section, boundedness, shape, and mass errors (Eq. (64) to (66)) are computed at the end of the runs ( $t = T$ ). The results are presented in Table 3, with the reduced computational times of the runs (Eq. (67)). Again, the levels of error in bound and mass remain very low. Good mass conservation is ensured even at coarse grid resolutions. Convergence of the geometric error is displayed in Fig. 18: the method exhibits between first and second-order convergence on two-dimensional triangular meshes.

Table 3: Error norms at the end of the circle deformation test, and computational cost of the runs.

$R/\Delta x$	$E_{\text{bound}}(T)$	$E_{\text{shape}}(T)$	$E_{\text{mass}}(T)$	mass loss [%]	$N_{\text{cores}}$	RCT [ $\mu\text{s}$ ]
12.5	$2.128 \times 10^{-7}$	$3.642 \times 10^{-2}$	$1.790 \times 10^{-9}$	$2.518 \times 10^{-6}$	1	4.7
25	$1.212 \times 10^{-7}$	$8.452 \times 10^{-3}$	$4.637 \times 10^{-8}$	$6.551 \times 10^{-5}$	1	3.8
50	$5.266 \times 10^{-8}$	$1.970 \times 10^{-3}$	$3.223 \times 10^{-8}$	$4.558 \times 10^{-5}$	2	3.7
100	$2.012 \times 10^{-8}$	$6.016 \times 10^{-4}$	$1.053 \times 10^{-8}$	$1.490 \times 10^{-5}$	8	5.2
200	$1.917 \times 10^{-9}$	$2.719 \times 10^{-4}$	$3.155 \times 10^{-9}$	$4.464 \times 10^{-6}$	28	6.1
400	$8.623 \times 10^{-10}$	$1.289 \times 10^{-4}$	$1.165 \times 10^{-9}$	$1.649 \times 10^{-6}$	112	6.4

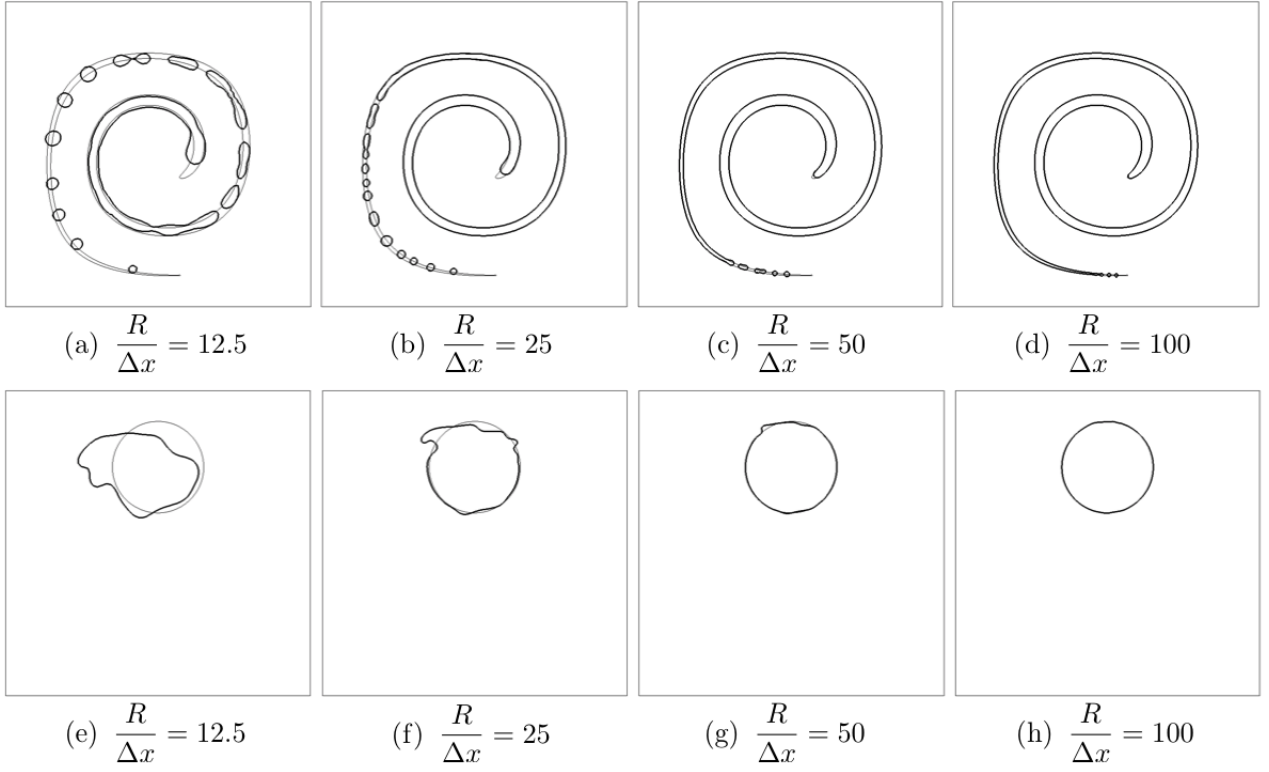


Figure 17: Interface contours at maximum deformation ( $t = T/2$ ; top) and final shape ( $t = T$ ; bottom) for various mesh resolutions. The thin line represents a simulation performed with 400 points in the disk radius.

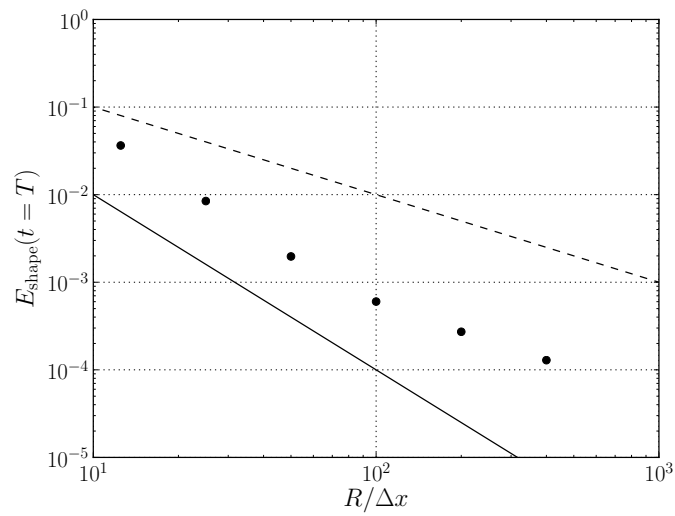


Figure 18: Shape error  $E_{\text{shape}}$  at the end of the 2D circle deformation in a vortex test ( $t = T$ ), for various mesh resolutions: from 12.5 to 400 points in the disk radius. The dashed and solid lines are first and second-order convergence, respectively.



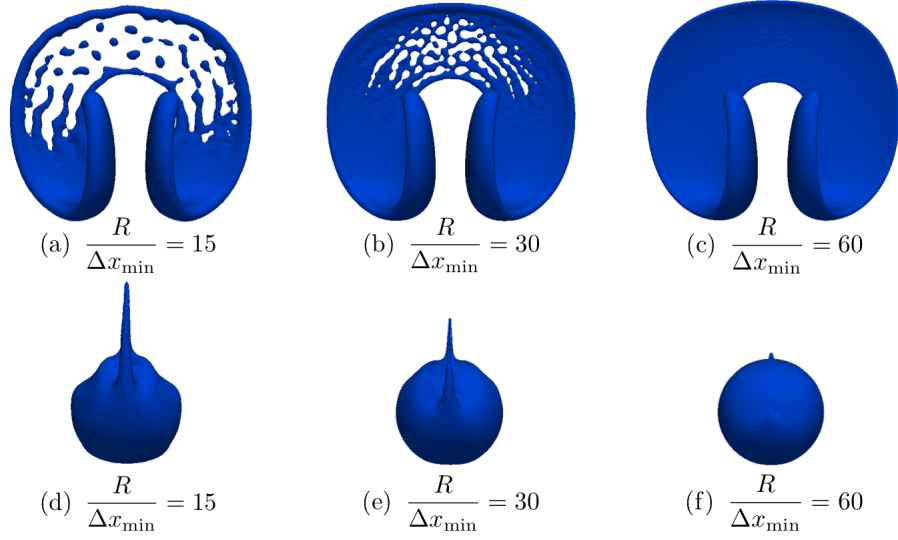


Figure 19: Interface contours at maximum stretch ( $t = T/2$ ; top) and final shape ( $t = T$ ; bottom) for various mesh resolutions.

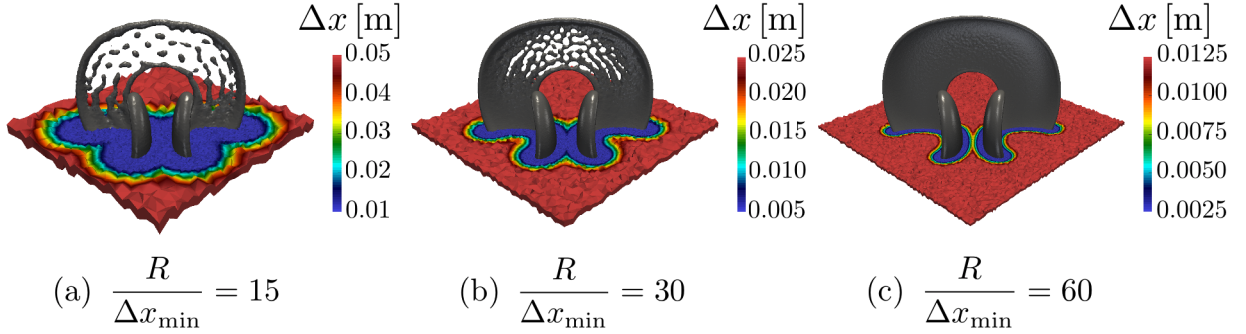


Figure 20: Tetrahedral cell size distribution at maximum stretch ( $t = T/2$ ) for various mesh resolutions.

#### 4.2.4 Sphere deformation in a vortex on adaptive tetrahedral grids

The previous test case can be extended to 3D [49]. A sphere of radius  $R = 0.15$  is initialized at  $(x, y, z) = (-0.15, -0.15, -0.15)$  in a  $[-0.5, 0.5]^3$  three-dimensional box. The advection velocity is specified as:

$$\mathbf{u}(x, y, z, t) = \begin{pmatrix} 2 \sin^2(\pi x) \sin(2\pi y) \sin(2\pi z) \cos(\pi t/T) \\ -\sin(2\pi x) \sin^2(\pi y) \sin(2\pi z) \cos(\pi t/T) \\ -\sin(2\pi x) \sin(2\pi y) \sin^2(\pi z) \cos(\pi t/T) \end{pmatrix} \quad (69)$$

The simulation time is chosen as  $T = 3s$ . At  $t = T$ , i.e. after one full stretch/unstretch cycle, the initial sphere is supposed to be recovered. At maximum deformation ( $t = T/2$ ), a thin sheet is formed and breaks up numerically into small spheres if the mesh is not fine enough. Adaptive tetrahedral meshes are used for this test case, dynamically and locally refined during the runs around the interface location, with a metric  $\Delta x_{\min}$ . Here, it has been chosen to change the background mesh from one simulation to another so that  $\Delta x_{\max}/\Delta x_{\min} = 5$  for all runs. Larger ratios will be considered in the next sections of this article. For all calculations of this section, the refined region is of half-width  $10\Delta x_{\min}$  ( $N_p = 10$ , see Fig. 8), and AMR is triggered each time the interface is closer than 5 cells to the border of this fine region. The narrow-band width  $|b|_{\max}$  is set to 10, and the initial CFL number is 0.64.

Interface contours are seen in Fig. 19 and contours along with the tetrahedral cell-size distribution in Fig. 20: good convergence is observed, and the final shape is almost retrieved for  $R/\Delta x = 60$ . The results are in accordance with the ones obtained for the two-dimensional case.

The computational efficiency, measured through the reduced computational time RCT (Eq. (67)), is presented in Table 4. The increase in computational cost remains low when increasing the resolution, and is mainly due to adaptive mesh refinement. The proposed ACLS procedure is scalable.

Table 4: Computational costs of the sphere deformation simulations

$R/\Delta x_{\min}$	$N_{\text{cores}}$	$N_{\text{cells,ini}}/N_{\text{cores}}$	RCT [ $\mu\text{s}$ ]	RCT <sub>ACLS</sub> [ $\mu\text{s}$ ]	RCT <sub>AMR</sub> [ $\mu\text{s}$ ]
15	10	93,361	153.6	77.2	73.6
30	28	122,645	210.6	86.3	120.3
60	224	71,239	372.4	99.4	267.2

### 4.3 Capillary-driven flows

#### 4.3.1 Damping of a capillary wave on static triangular grids

The damping of a two-dimensional surface wave is then studied. Two superposed immiscible viscous fluids at rest are separated by a flat interface, initially perturbed by a small-amplitude sinusoidal capillary wave. The initial signed-distance function is written as:

$$\phi_0(x, y) = -y - A_0 \cos\left(\frac{2\pi x}{\lambda}\right) \quad (70)$$

in a two-dimensional  $[-\lambda/2; \lambda/2]^2$  domain which is periodic in the x-direction, and with slip walls for top and bottom boundaries. The initial wave amplitude is chosen as  $A_0 = \lambda/100$ , with  $\lambda = 2\pi$  the wavelength of the perturbation. The initial perturbation is damped over time due to viscous dissipation.

The results are compared to analytical solutions from Prosperetti's theory [50], which quantifies the viscous damping for vanishingly small amplitudes and infinite domains as an initial-value problem. The theoretical solution for two fluids with equal kinematic viscosity  $\nu$  and  $\lambda = 2\pi$  is:

$$A_{\text{th}}(t) = \frac{4(1-4\beta)\nu^2}{8(1-4\beta)\nu^2 + \omega_0^2} A_0 \text{erfc}\sqrt{\nu t} + \sum_{i=1}^4 \frac{z_i}{Z_i} \left( \frac{A_0 \omega_0^2}{z_i^2 - \nu} \right) \exp((z_i^2 - \nu)t) \text{erfc}(z_i \sqrt{t}) \quad (71)$$

where  $z_i$  are the four roots - two pairs of conjugated complex numbers - of the following fourth-order polynomial equation:

$$z^4 - 4\beta\sqrt{\nu}z^3 + 2(1-6\beta)\nu z^2 + 4(1-3\beta)\nu^{3/2}z + (1-4\beta)\nu^2 + \omega_0^2 = 0 \quad (72)$$

and  $Z_i = \prod_{j=1, j \neq i}^4 (z_j - z_i)$ . In the case of equal density  $\rho$ ,  $\beta = 1/4$  and the normal-mode oscillation frequency  $\omega_0$  is computed from the dispersion relation as follows:

$$\omega_0^2 = \frac{\sigma}{2\rho} \quad (73)$$

The non-dimensional time is  $t^* = \omega_0 t$ . The calculations are performed on three homogeneous triangular grids ( $\lambda/\Delta x = 30; 100; 200$ ) up to  $T^* = \omega_0 T = 25$ , which corresponds to approximately 4 oscillations. The unity density ratio case is investigated, gravity is neglected, and 3 steps of reinitialization are performed at each timestep. The physical and numerical parameters of the runs are summarized in Table 5.

Table 5: Physical and numerical parameters for the viscous damping of a capillary wave case.

$\rho_l/\rho_g$	$\mu_l/\mu_g$	$\text{La} = \rho\lambda\sigma/\mu^2$	$\lambda/\Delta x$	$\Delta t^*$
1	1	3000	30; 100; 200	0.001

The temporal evolution of the wave amplitude over time is depicted in Fig. 21 (a). The damping is correctly reproduced from 100 points in the wavelength  $\lambda$  in the case of equal densities. Fig. 21 (b) shows the instantaneous relative amplitude error  $|A(t) - A_{\text{th}}(t)|/A_0$  and confirm the convergence properties. To quantify further the differences between theory and numerical simulations, the following  $L_2$  norm is computed, as in [19]:

$$L_2 = \frac{1}{\lambda} \sqrt{\frac{1}{T^*} \int_{t^*=0}^{T^*} (A(t^*) - A_{\text{th}}(t^*))^2 dt^*} \quad (74)$$

where  $A$  and  $A_{\text{th}}$  are the computed and theoretical wave amplitudes at a given instant, respectively. The  $L_2$ -norm values and the computational costs of the runs are presented in Table 6.

Again, the results show fair convergence and accordance with theoretical data in the case of unity density ratio. It validates the overall algorithm robustness and physical meaning in the case of surface tension/viscosity interactions on unstructured grids.

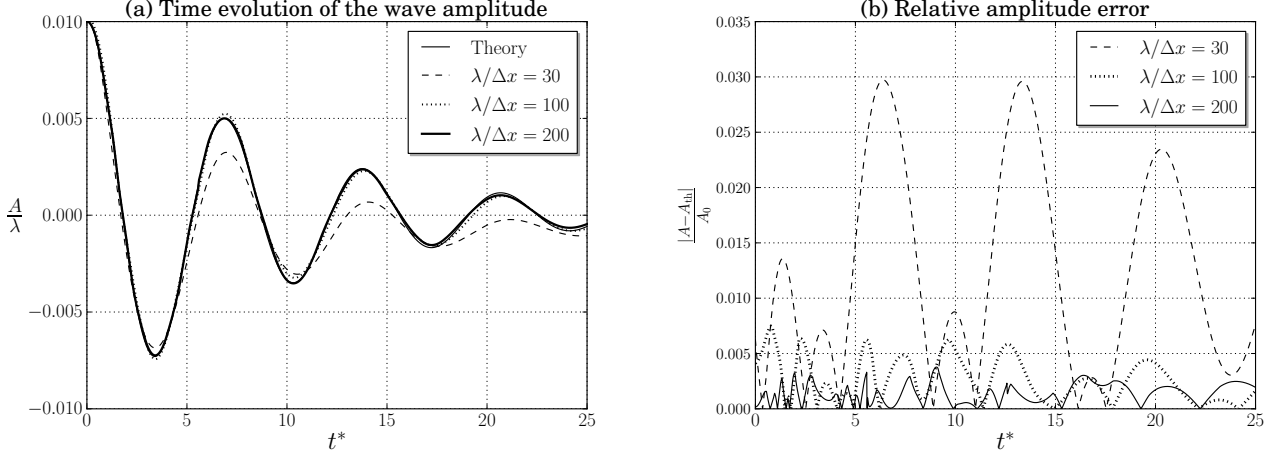


Figure 21: (a) Temporal evolution of the dimensionless wave amplitude  $A/\lambda$  over non-dimensional time  $t^* = \omega_0 t$ ; (b) Instantaneous relative amplitude error  $|A - A_{th}|/A_0$ .

Table 6: Error levels of the  $L_2$  norm (Eq. (74)) and reduced computational time (RCT; Eq. (67)) for each resolution.

$\lambda/\Delta x$	$L_2$	$N_{\text{cores}}$	$N_{\text{cells}}/N_{\text{cores}}$	RCT [ $\mu\text{s}$ ]
30	$1.528 \times 10^{-4}$	1	2028	23.2
100	$3.250 \times 10^{-5}$	1	22,476	13.3
200	$1.647 \times 10^{-5}$	4	22,520	20.2

#### 4.3.2 Spurious currents on static tetrahedral grids

This elementary test is used to assess the numerical errors induced by the surface tension computation procedure. For a static droplet, it aims to show the capability of a method to predict Laplace’s law, i.e. the balance between surface tension and pressure forces. For a droplet in equilibrium, the theoretical velocity is 0 and thus non-physical velocities indicate the presence of spurious (or parasitic) currents. Exact balance (to within machine accuracy) has been shown first on cartesian grids in [19].

Most past studies have been conducted with low Laplace numbers. In this work, we study spurious currents to check the robustness and assess the numerical errors of the algorithms in physical conditions close to the ones found in the range of the targeted practical applications, i.e. realistic liquid injection systems. To this mean, a high Laplace number is investigated:  $\text{La} = 2,000,000$ .

To study their magnitude over time, the maximum Capillary number is used:  $\text{Ca}_{\text{max}} = \mu_l u_{\text{max}}/\sigma$ . Time is normalized using a visco-capillary timescale:  $t^* = t/\tau$ , with  $\tau = \mu_l D/\sigma$ . A three-dimensional droplet of diameter  $D = 2\text{mm}$  is initialized in a unit centimetric square box. The simulations are performed on three static tetrahedral grids of resolutions  $\Delta x = 200\mu\text{m}, 100\mu\text{m}, 50\mu\text{m}$ , up to  $t^* = 10,000$ . Three reinitialization steps are performed at each iteration, and the explicit capillary time-step constraint STN in Eq. (60) is set to 0.71 for all resolutions. The physical and numerical parameters are presented in Table 7.

Table 7: Physical and numerical parameters for the measurements of spurious currents.

$\rho_l/\rho_g$	$\mu_l/\mu_g$	$\text{La} = \rho_l \sigma D/\mu_l^2$	$D/\Delta x$
1000	50	$2 \times 10^6$	10; 20; 40

Results are presented in Fig. 22. Reasonably low errors ( $\text{Ca}_{\text{max}} \sim 10^{-3}$ ) are seen for all resolutions, and a significant reduction of the spurious current magnitude is observed for  $D/\Delta x = 40$ . The spatial convergence of the spurious currents, which dominate the solution in such physical conditions, is displayed, and the presented procedure is demonstrated to be highly robust on unstructured grids.

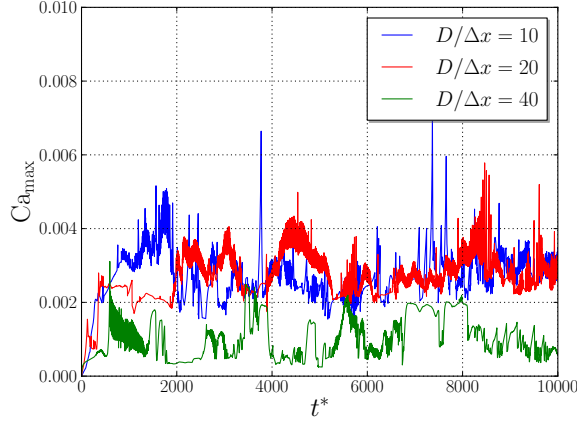


Figure 22: Spurious currents around a static droplet on tetrahedral meshes.  $La = 2,000,000$ .  $\rho_l/\rho_g = 1000$ .  $\mu_l/\mu_g = 50$ .

### 4.3.3 Rayleigh-Plateau instability

In this section, following the studies of [19, 20, 25, 51], numerical predictions of the capillary instability are performed on 3D static unstructured meshes and validated against Weber’s linear theory [52]. A static column of water of density  $\rho_l$ , dynamic viscosity  $\mu_l$ , and surface tension  $\sigma$  in quiescent air is considered. This column is infinite in the x-direction and its radius is  $r_0$ . A small axisymmetric perturbation wave is imposed at the column surface. The solution is of the form:

$$r(x, t) = r_0 + \epsilon r_0 \exp(\omega t + i k x) \quad (75)$$

where  $\epsilon$  is the perturbation, assumed to be small,  $k = 2\pi/\lambda$  is the wavenumber, and  $\omega$  is the growth rate. Using the non-dimensional wavenumber  $\xi = k r_0$  and growth rate  $\omega_0 = \sqrt{\sigma/(2\rho_l r_0^3)}$ , the equation for the growth rate in Weber’s theory applied to static jets, for  $\xi < 1$ , is given by:

$$\omega^2 + \frac{3\mu_l \xi^2}{\rho_l r_0^2} \omega = \omega_0^2 \xi^2 (1 - \xi^2) \quad (76)$$

The mathematical developments leading to this result can be found in [53].

The simulations are performed in a 3D, periodic in the x-direction, cylindrical domain of length  $[0; \lambda]$  and radius  $[0; 3r_0]$ . The signed-distance function is initialized as:

$$\phi_0(x, y, z) = r_0 - \epsilon r_0 \cos(kx) - \sqrt{y^2 + z^2} \quad (77)$$

where  $r_0 = (1/3) \times 10^{-3} \text{m}$ , and the initial velocity is set to 0. In order to keep the interface free from spurious signals, the initial perturbation is set to be of the same order of magnitude than the homogeneous tetrahedral-cell size:  $\epsilon r_0 \sim \Delta x$ , here  $\epsilon = 0.06$ . The runs are performed at various  $\xi$  values, ranging between 0 and 1 (unstable oscillatory solutions), and using an effective resolution of  $r_0/\Delta x = 6$ . The definition of the cases can be found in Table 8. For all cases, the narrow-band width is  $|b|_{\max} = 12$  and 3 steps of reinitialization are

Table 8: Cases definition for the Rayleigh-Plateau instability simulations.

Case	$\lambda$ [mm]	$\lambda/r_0$	$\xi = k r_0$	$N_{\text{cells}}$
1	16	48	0.13	2.5M
2	8	24	0.26	1.3M
3	4	12	0.52	637K
4	3.3	10	0.63	525K
5	2.8	8.5	0.74	453K
6	2.3	7	0.90	372K

performed at each timestep. Classical water/air parameters are used, and are summarized in Table 9. The Laplace number is  $La = \rho_l r_0 \sigma / \mu_l^2 = 18,530$ .

The interface contours for  $\xi = 0.52$  (Case 3) can be seen in Fig. 23: implicit interface topology changes are correctly predicted. The relative deformation based on the maximum radius is defined at  $x = \lambda/2$  as:

Table 9: Physical parameters for the Rayleigh-Plateau instability simulations.

$\rho_l/\rho_g$	$\mu_l/\mu_g$	La
829	62	18,530

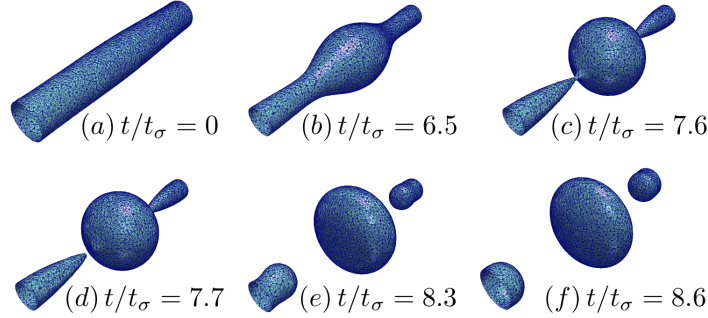


Figure 23: Rayleigh-Plateau instability on a homogeneous tetrahedral mesh: interface contours at various non-dimensional times  $t/t_\sigma$ .  $\text{La} = 18,530$ .  $\xi = kr_0 = 0.52$ .  $r_0/\Delta x = 6$ .  $t_\sigma = 1/\omega_0$ .

$$\frac{r_{\max} - r_0}{\epsilon r_0} = \exp(\omega t) \quad (78)$$

The characteristic time is  $t_\sigma = 1/\omega_0 = 1.013\text{ms}$ . Using  $t_\sigma$ , Eq. (78) becomes:

$$\ln\left(\frac{r_{\max} - r_0}{\epsilon r_0}\right) = \frac{\omega}{\omega_0} \frac{t}{t_\sigma} \quad (79)$$

This term is linear in time and the derivative is the non-dimensional growth rate. However, due to the use of fully 3D tetrahedral meshes, several causes of inaccuracy might deteriorate the quality of the computed growth rate: (i) the jet can be slightly off-centered in the radial direction, (ii) the maximum amplitude can be strongly shifted from the middle of the jet in the longitudinal direction. While problem (ii) is solved by choosing an initial perturbation of the same order of magnitude than the cell size, problem (i) requires a dedicated methodology. To post-process the results, the jet is discretized into rings of chosen thickness  $\delta = 80\mu\text{m}$ , and area  $A$ . For each ring  $i$  of each temporal solution  $n$ , the mean radius is computed from the mean center:

$$\bar{r}_i^n = \frac{1}{A} \int_A \sqrt{(y - \bar{y})^2 + (z - \bar{z})^2} dA \quad (80)$$

where  $\bar{y} = (1/A) \int_A y dA$  and  $\bar{z} = (1/A) \int_A z dA$ . The maximum mean radius  $\bar{r}_{\max}^n$  is then computed for each temporal solution  $n$ . The temporal evolution of the relative deformation for  $\xi = 0.52$  (Case 3) is plotted in Fig. 24 (a). Despite a non-linear phase at the beginning of the calculation, remaining until the disturbance has grown quite large, the non-dimensional growth rate  $\omega/\omega_0$  is correctly predicted. To quantify the growth rate only in the linear part, a temporal mean is performed from  $t/t_\sigma = 4$  to the breakup. The standard deviation SD of the growth-rate signal is also computed within the range considered:

$$\text{SD} = \sqrt{\frac{1}{N-1} \sum_{n=1}^{N-1} \left( \left. \frac{\omega}{\omega_0} \right|_n - \left. \frac{\omega}{\omega_0} \right|_{\text{mean}} \right)^2} \quad (81)$$

where  $N$  is the total number of temporal solutions.

The theoretical and numerical growth rates for all computed non-dimensional wavenumbers  $\xi$  are presented in Table 10 and depicted in the dispersion diagram of Fig. 24 (b). Larger standard deviations are obtained in the second and sixth cases ( $\xi = 0.26$  and  $\xi = 0.90$  respectively): the linear regime is more oscillatory, less clear to identify, in these two conducted simulations.

#### 4.3.4 3D droplet collision using adaptive mesh refinement

This section aims to validate the unstructured discretization of the reformulated reinitialization equation in the case of a interface merging scenario. This is a non-trivial case: a bad prediction of the resharpening term leads to the locking of the interface on the grid, and as a consequence keeps the two interfaces of the two droplets from

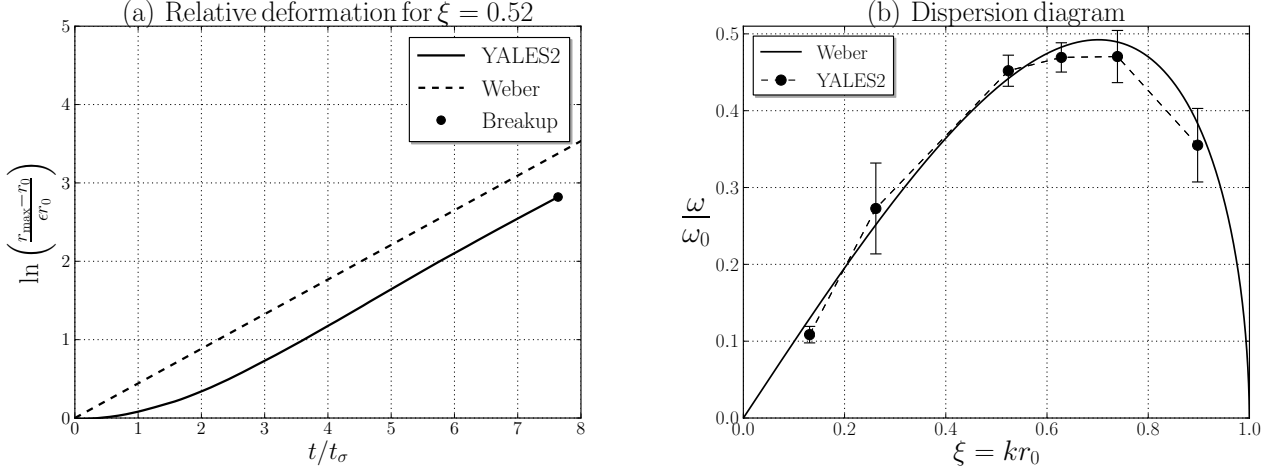


Figure 24: Rayleigh-Plateau instability.  $La = 18,530$ .  $r_0/\Delta x = 6$ . (a) Temporal evolution of the relative deformation based on the maximum radius for  $\xi = kr_0 = 0.52$ ,  $t_\sigma = 1/\omega_0$ . (b) Dispersion diagram; the error bars represent the standard deviation SD of the growth-rate signal  $\pm SD$  (see Eq. (81)).

Table 10: Numerically predicted mean growth rates versus theoretical growth rates from Weber’s theory, with  $r_0/\Delta x = 6$ .

Case	$\xi$	$\omega/\omega_0 _{\text{mean}}$	$\omega/\omega_0 _{\text{th}}$	Relative error	Standard deviation SD
1	0.13	0.109	0.130	-16.2%	0.011
2	0.26	0.273	0.252	+8.3%	0.059
3	0.52	0.452	0.442	+2.3%	0.020
4	0.63	0.469	0.483	-2.9%	0.019
5	0.74	0.470	0.489	-3.9%	0.034
6	0.90	0.355	0.383	-7.3%	0.048

merging. In order to demonstrate that the presented procedure allows implicit topology changes, a head-on collision of two equal-size droplets of water in air is computed.

The physical parameters involved in this process are the Weber number  $We = \rho_l u_r^2 D / \sigma$ , with  $u_r$  the relative velocity of the two droplets; the Ohnesorge number  $Oh = \mu_l / \sqrt{\sigma \rho_l D}$ ; the non-dimensional impact parameter  $X$ , which is 0 for a head-on collision; and the droplet diameter ratio  $\Delta$ . The values are presented in Table 11 and correspond to the experiment of [54], in the reflexive separation regime. The narrow-band width is  $|b|_{\text{max}} = 10$ , and the minimum cell size  $\Delta x_{\text{min}} = 6\mu\text{m}$  is imposed in a refined region of half-width  $10\Delta x_{\text{min}}$  ( $N_p = 10$ , see Fig. 8). The mesh adaptation process is triggered automatically when the interface is 5 cells away from the border of the refined region, which leads to massive computational savings. Three steps of reinitialization are performed at each iteration.

Instantaneous interface positions and meshes are presented in Fig. 25, and show good interface quality and accuracy, even after a very large number of iterations ( $\Delta t \sim 0.01\mu\text{s}$ ). Despite the presence of an entrapped bubble due to the lack of perfect symmetry with the unstructured grid, implicit topology changes are shown, and fair overall dynamics is displayed compared to the experimental results of [54]. The computational efficiency is presented in Table 12.

Table 11: Physical parameters for the water/air head-on droplet collision simulation.

We	Oh	$\Delta$	X
23	0.0047	1	0

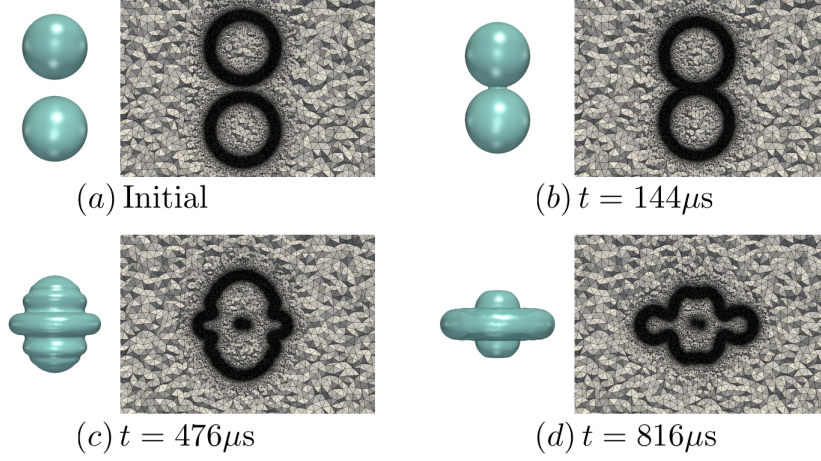


Figure 25: Water/Air droplet collision.  $We = 23$ ;  $Oh = 0.0047$ ;  $X=0$ ;  $\Delta = 1$ ;  $D/\Delta x_{\min} = 133$ .

Table 12: Computational performances of the water/air head-on droplet collision simulation.

$D/\Delta x_{\min}$	$\Delta x_{\max}/\Delta x_{\min}$	$N_{\text{cells,ini}}$	$N_{\text{cores}}$	$RCT_{\text{AMR}} [\mu s]$	$RCT [\mu s]$
133	17	36M	448	23.8	239.4

## 5 Application to injection systems

### 5.1 LES of a low-pressure water jet from a compound nozzle

This study follows the experiments of [30] and the simulations of [31]. A single low-pressure compound nozzle is considered: three shifted cylinders are superimposed, as depicted in Fig. 26, from which water is injected in quiescent air at Reynolds number  $Re = 3653$ , based on the discharge orifice diameter  $d_{\text{inj}} = 180 \mu\text{m}$ , and liquid Weber number  $We_l = 1061$ . The shift of the disks induces non-axial velocity components, and a double-vortex flow is formed, as illustrated in Fig. 27.

In these physical conditions, primary atomization is mainly driven by capillary effects such as ligament breakup and jet forming by bubble burst, and enhanced by liquid turbulence. The issued water jet is slow, and hence aerodynamic forces are negligible. As the injector geometry strongly conditions the primary breakup process, the unsteady resolution of the internal flow is important, as mentioned in [31].

In this work, the whole flow is computed by the mean of one Large-Eddy Simulation (LES), from the internal flow to the primary atomization process. The LES turbulence model is the dynamic Smagorinsky model [55, 56]. Adaptive tetrahedral meshes are used and two resolutions are investigated:  $\Delta x_{\min} = 5 \mu\text{m}$  and  $\Delta x_{\min} = 2.5 \mu\text{m}$ . For the two simulations, the refined region is of half-width  $10\Delta x_{\min}$  ( $N_p = 10$ , see Fig. 8), and AMR is triggered each time the interface is 5 cells away from the edge of this region. The jet morphology and angle are studied, and compared to the ones of [31].

#### 5.1.1 Internal flow

As presented above, the adaptive mesh refinement strategy is based on interface displacement, and thus the mesh size inside the injector, where only liquid is present, is the same for the two conducted calculations: the  $y^+$  field represented at the walls is shown in Fig. 28. Converged temporal statistics of the flow velocity projected in the nozzle exit plane are shown in Fig. 29: the double-vortex secondary flow of the discharged disk (Disk 3 of Fig. 26) is clearly displayed.



Figure 26: Compound nozzle geometry: triple-disk injector.



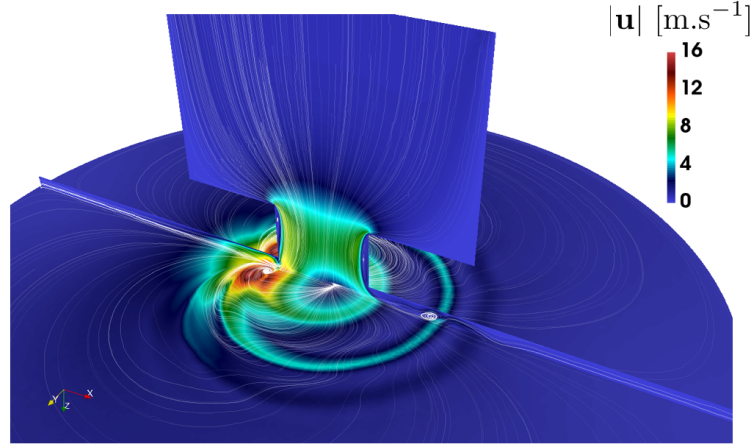


Figure 27: Illustration of the double-vortex internal flow in the cavity disk (Disk 2 in Fig. 26): velocity magnitude along with streamlines.

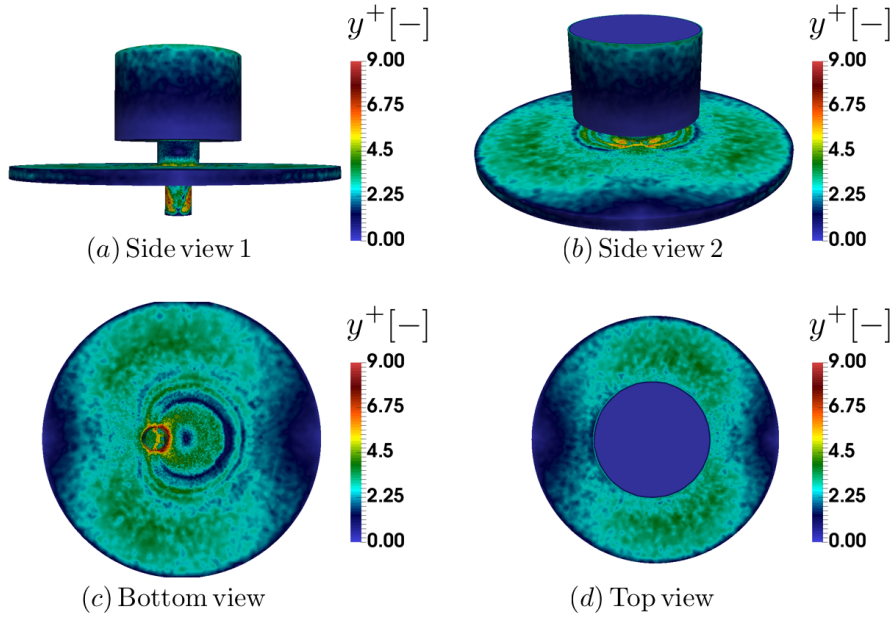


Figure 28:  $y^+$  distribution represented at boundaries.  $d_{inj} = 180\mu\text{m}$ .  $\text{Re} = 3653$ .  $\text{We}_1 = 1061$ .

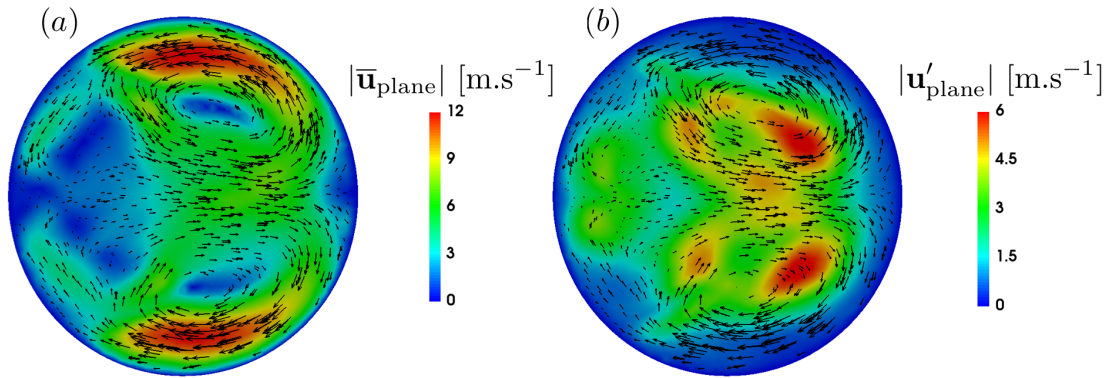


Figure 29: Velocity statistics: (a) Mean and (b) RMS velocity projected in the exit section.  $d_{inj} = 180\mu\text{m}$ .  $\text{Re} = 3653$ .



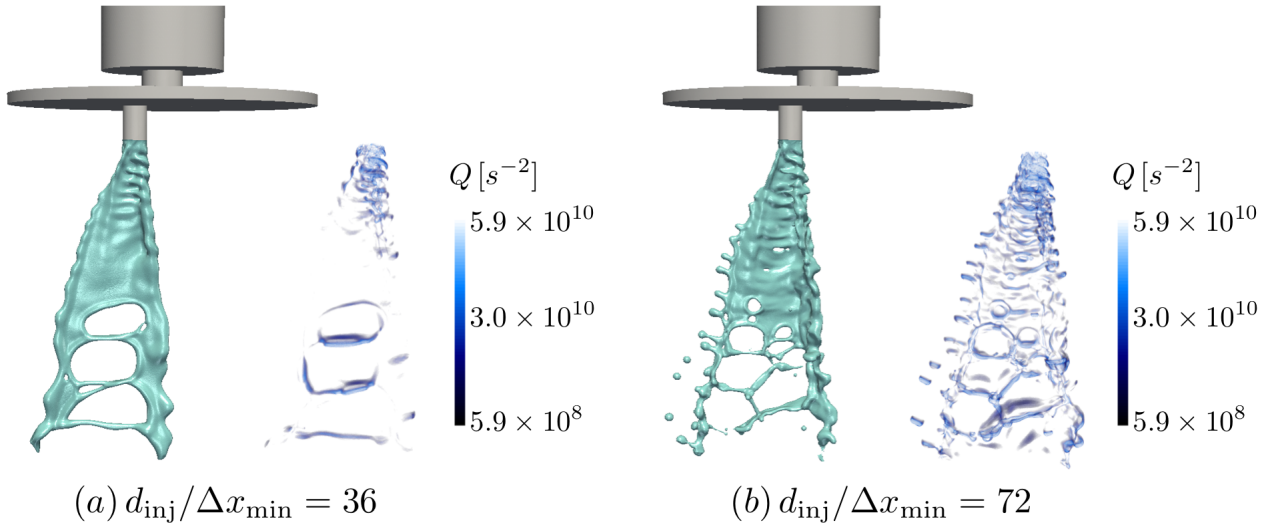


Figure 30: Interface contours and Q-criterion of the water jet atomizing from a low-pressure compound nozzle in quiescent air at the two investigated resolutions: (a)  $\Delta x_{\min} = 5\mu\text{m}$  and (b)  $\Delta x_{\min} = 2.5\mu\text{m}$ .

### 5.1.2 External flow

A comparison of interface topologies and vorticity-dominated regions at the two simulated resolutions are shown in Fig. 30. A clear difference is seen in the appearance of jet instabilities: a closeup on some capillary effects, especially the Rayleigh-Plateau instability is shown for the  $2.5\mu\text{m}$  case in Fig. 31. It depicts the temporal evolution of a ligament, circled in red in Fig. 31 (a), which is pinched off in Fig. 31 (b) until breakup (c) to produce two droplets. These outcome drops later collide and merge. These observations are simply not accessible with  $\Delta x_{\min} = 5\mu\text{m}$ . The jet topology in the  $2.5\mu\text{m}$  case is close to the one obtained by a DNS simulation at  $1.44\mu\text{m}$ , performed with a symmetry plane in [31]. This similarity is encouraging to target an even higher resolution. Fig. 32 shows the mesh cell-size distribution for both computed resolutions. The background mesh is kept identical for both, and it does not affect the quality of the solution: only smooth cell-size gradients are present in the domain, even with a higher cell-size ratio  $\Delta x_{\max}/\Delta x_{\min} = 22.8$  in the  $2.5\mu\text{m}$  case.

Eventually, the spray angle  $\alpha$  is estimated at various distances  $z/d_{\text{inj}}$  away from the nozzle exit for  $d_{\text{inj}}/\Delta x_{\min} = 72$ , in order to compare with the simulations and experiments of [31]. An image processing technique has been employed on 250 frames, corresponding to  $250\mu\text{s}$  of physical time, of the projected shadow of the spray to obtain a single mean image: Fig. 33 (a). The angle values are presented in Fig. 33 (b): the angles are rather overestimated, especially close to the nozzle, compared to the experimental values presented in [31], which lie between  $20^\circ$  and  $25^\circ$ . This could be explained by the lack of atomization close to the nozzle in our numerical simulations. Nevertheless the overall behavior is correct, despite a more significant decrease of the angle with the distance from the nozzle exit.

### 5.1.3 Computational performances

The computational cost and performances of the two conducted simulations on adaptive tetrahedral grids are shown in Table 13. Good scalability of the unstructured ACLS/AMR procedure is displayed. The cost of the simulations is moderate: in comparison, the same run at  $d_{\text{inj}}/\Delta x_{\min} = 72$  on a static grid requires 8192 cores and 1.62B cells. Hence, the present method allows huge computational savings for simulations on tetrahedral grids.

Table 13: Computational costs and performances of the low-pressure compound injector simulations on adaptive grids. The simulations have been conducted on 2nd Gen AMD EPYC processors.

$d_{\text{inj}}/\Delta x_{\min}$	$N_{\text{cores}}$	$N_{\text{cells}}$	RCT [ $\mu\text{s}$ ]	RCT <sub>ACLS</sub> [ $\mu\text{s}$ ]	RCT <sub>AMR</sub> [ $\mu\text{s}$ ]	RCT <sub>Poisson</sub> [ $\mu\text{s}$ ]	CPU time [h]/Phys. time [ms]
36	600	48M	342.3	99.7	137.2	50.7	52,174
72	1400	141M	610.0	142.0	208.6	199.2	915,531

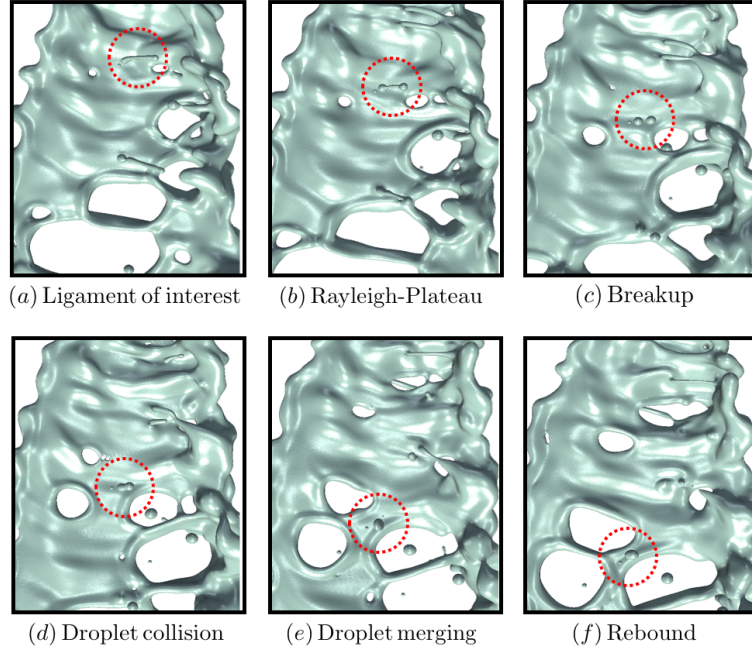


Figure 31: Closeup on some capillary effects during the atomization process of a water jet in quiescent air from a low-pressure compound nozzle.  $d_{\text{inj}} = 180\mu\text{m}$ .  $\text{Re} = 3653$ .  $\text{We}_l = 1061$ .  $\Delta x_{\text{min}} = 2.5\mu\text{m}$ .

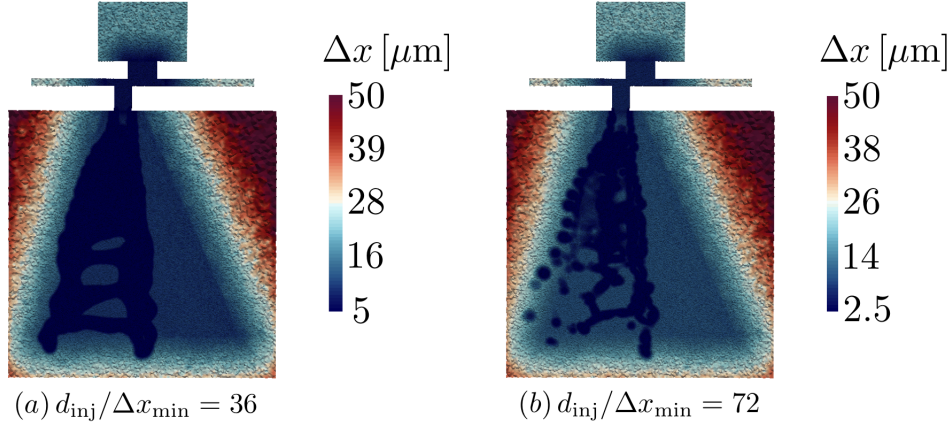


Figure 32: Mid-plane colored by cell size.  $d_{\text{inj}} = 180\mu\text{m}$ . (a)  $\Delta x_{\text{min}} = 5\mu\text{m}$ ; (b)  $\Delta x_{\text{min}} = 2.5\mu\text{m}$ .

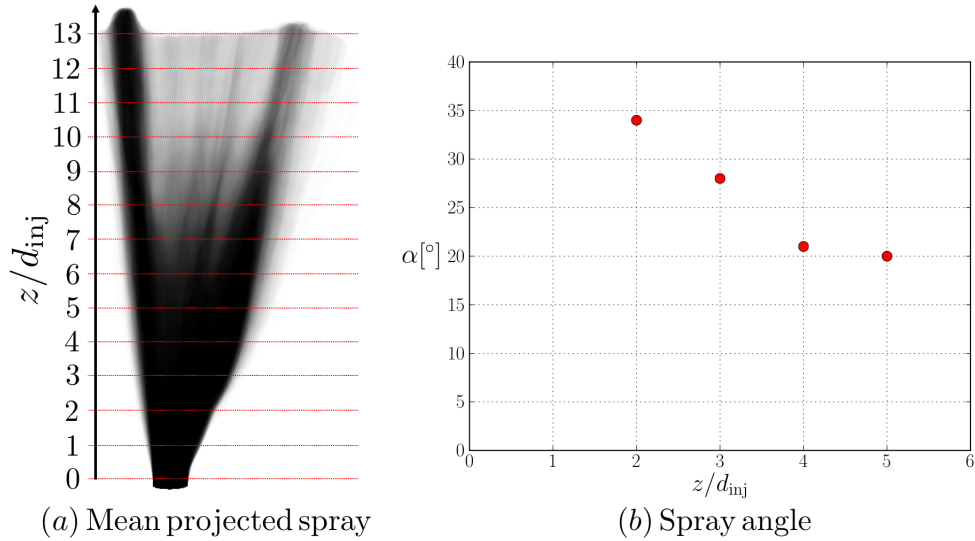


Figure 33: Estimation of the spray angle  $\alpha$  at different distances  $z/d_{\text{inj}}$  from the nozzle exit.  $d_{\text{inj}}/\Delta x_{\text{min}} = 72$ .

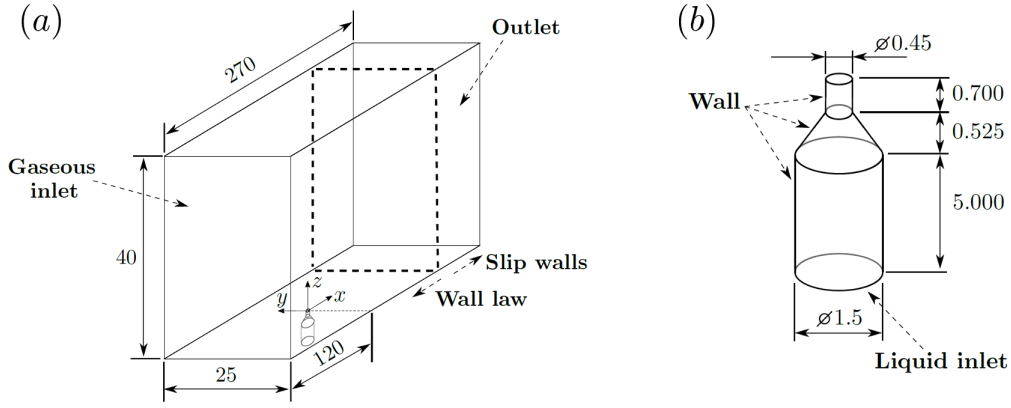


Figure 34: Numerical domain and boundary conditions of the experimental test bench of Becker and Hassa [32]: (a) complete domain, (b) detailed view of the injection nozzle. All dimensions are in mm.

## 5.2 LES of a high-pressure kerosene jet in crossflow

In order to test the capabilities of the algorithm for resolving atomization in aeronautical injection systems, a non-reactive liquid kerosene jet in crossflow (JICF) injection has been simulated. Liquid is injected through a nozzle into a plenum where a stream of air flows in the perpendicular direction to the liquid (the crossflow). Then, the jet bends towards the air direction and atomizes due to the aerodynamic interaction. The chosen configuration is the experimental test-bench by Becker and Hassa [32], which consists of a kerosene JICF injected into a high-pressure environment representative of the ambient conditions found within aeronautical gas turbines. The physics of the JICF is governed by the two following parameters: the kinetic energy ratio between the liquid and gaseous phases  $q$  and the gaseous Weber number  $We_g$ . They are defined as follows:

$$q = \frac{\rho_l u_l^2}{\rho_g u_g^2} \quad \text{and} \quad We_g = \frac{\rho_g u_g^2 d_{inj}}{\sigma} \quad (82)$$

where  $\rho$  is the density,  $u$  is the bulk velocity of the flow,  $d_{inj}$  is the liquid nozzle diameter (0.45mm in Fig. 34) and  $\sigma$  is the surface tension coefficient. The subindexes  $l$  and  $g$  denote liquid and gaseous phases respectively.

### 5.2.1 Numerical setup

Fig. 34 (a) shows the numerical setup replicating the experiment from [32]. It consists of a plenum of dimensions  $(L_x, L_y, L_z) = (270\text{mm}, 40\text{mm}, 25\text{mm})$  and a tapered nozzle for liquid injection. A close-up view of the nozzle is shown in Fig. 34 (b). The injected liquid is JET-A kerosene with density  $\rho_l = 795\text{kg.m}^{-3}$ , dynamic viscosity  $\mu_l = 1.5 \times 10^{-3}\text{kg.m}^{-1}\text{s}^{-1}$  and surface tension  $\sigma = 22 \times 10^{-3}\text{kg.s}^{-2}$ . The gaseous crossflow is pressurized air at 6bar with density  $\rho_g = 7.21\text{kg.m}^{-3}$  and dynamic viscosity  $\mu_g = 1.82 \times 10^{-5}\text{kg.m}^{-1}\text{s}^{-1}$ . The density and dynamic viscosity ratios are respectively  $\rho_l/\rho_g = 110$  and  $\mu_l/\mu_g = 83$ . The Ohnesorge number is  $Oh = \mu_l/\sqrt{\rho_l \sigma d_{inj}} = 0.017$ . Two operating points studied experimentally by [32] are simulated. They are given by the dimensionless numbers  $q$  and  $We_g$  shown in Table 14. Two interface resolutions are considered for both operating points:  $\Delta x_{min} = 20\mu\text{m}$  and  $10\mu\text{m}$ , that can be expressed relative to the injection diameter as  $d_{inj}/\Delta x_{min} = 22.5$  and 45 respectively. Therefore, a total of four JICF cases are computed. These simulations are performed with LES where a dynamic Smagorinsky closure is used for modeling the unresolved turbulence scales [55, 56].

Table 14: JICF operating points

Operating point	$u_l[\text{m.s}^{-1}]$	$u_g[\text{m.s}^{-1}]$	$q$	$We_g$	$d_{inj}/\Delta x_{min}$
1	17.5	75	6	830	22.5; 45
2	23.33	100	6	1470	22.5; 45

For kerosene injection, a Poiseuille profile is prescribed at the liquid inlet. The mean velocity of this profile is equal to the bulk velocity  $u_l$  given in Table 14. Regarding the gaseous phase, the experiments from [32] report a gaseous boundary layer thickness of 5mm right upstream the injector. As the numerical domain is smaller than the experimental test bench, a velocity profile with near-wall and outer regions is specified at the gaseous inlet. The near-wall region is modeled considering that the gaseous phase evolves in the streamwise direction as

a turbulent boundary layer along a flat plate. Its thickness is calculated following a  $1/7$ th power law. The outer part of the velocity profile follows a parabolic law. The injected mass flow rate is the same one as reported in the experiments.

The initial mesh is the same for all simulations: it consists of 32M tetrahedral elements ( $N_{\text{cells}} = 32\text{M}$ ). The number of cores  $N_{\text{cores}}$  allocated to each simulation ensures a constant ratio  $N_{\text{cells}}/N_{\text{cores}}$  between 100,000 and 150,000. For all simulations, the refined region is of half-width  $10\Delta x_{\text{min}}$  ( $N_p = 10$ , see Fig. 8), and AMR is triggered each time the interface is 4 cells away from the edge of this region. Each case will yield different meshes as function of the operating condition and mesh resolution, see Table 16. Therefore, the simulations will have different costs, see Table 17. Physical running times simulated depend on the computational resources available.

### 5.2.2 Vertical trajectory and experimental validation

Numerical simulations are validated with the experimental correlation obtained by [32] for the vertical jet penetration. It has been obtained by testing experimentally several operating conditions, and has a standard deviation of value 0.81. The correlation corresponds to the trajectory of the jet's windward side, and is given by:

$$\frac{z}{d_{\text{inj}}} = 1.57q^{0.36} \ln \left( 1 + 3.81 \frac{x}{d_{\text{inj}}} \right) \quad (83)$$

This trajectory is valid up to the streamwise position  $x/d_{\text{inj}} = 22$  and depends solely on the  $q$  factor, as the authors state that the influence of the Weber number on the trajectory is negligible. Other experimental studies, such as the one by Ragucci [33], provide experimental correlations with a dependence on  $We$ . While the role of  $We$  on the trajectory is still an open question, the results of our simulations are only compared to the ones of [32] (Eq. (83)).

The resulting mean trajectories are shown in Fig. 35. The experimental correlation is given by the black solid line, and the shadowed area denotes its confidence region. The red and blue lines are the numerical mean trajectories for the coarse and fine mesh resolutions,  $d_{\text{inj}}/\Delta x_{\text{min}} = 22.5$  and  $d_{\text{inj}}/\Delta x_{\text{min}} = 45$ , respectively. They are obtained by averaging the instantaneous trajectories of the windward side. Fig. 35 (a) shows the first operating point from Table 14. It can be seen that the numerical penetration in the near-nozzle region is slightly underestimated for both mesh resolutions. However, as the jet moves downstream its penetration fits into the experimental limits. The effect of mesh resolution on the trajectory is not very significant for this operating point. The second operating point, Fig. 35 (b), shows a larger difference in trajectory with the minimum cell size. For the coarsest resolution  $d_{\text{inj}}/\Delta x_{\text{min}} = 22.5$ , the trajectory is underestimated and does not fall within the confidence interval of the correlation, although its tendency is similar to the experimental results. For the finest resolution  $d_{\text{inj}}/\Delta x_{\text{min}} = 45$ , the trajectory follows very closely the experimental correlation. This discrepancy with respect to the other trajectories is due to liquid separation at the nozzle's convergent exit section in the fine cases. The effective area of the jet is reduced when injected to the plenum and, due to mass conservation, its velocity increases. Consequently, the jet penetrates further than the one of the coarse mesh, where liquid separation is not observed. Fig. 36 shows clearly this phenomenon. This difference is attributed to the resolution of the detached boundary layer within the nozzle by the fine mesh, which is not resolved by the coarse one. For the operating point at lower  $We_g$ , liquid separation is obtained for both mesh resolutions. This can be explained by a lower liquid Reynolds number for this operating point, therefore the boundary layer is thicker and can be captured by coarse mesh. These observations, where solutions with liquid detachment match their trajectories to the experimental one, suggest that this phenomenon was actually occurring in the experiments of [32].

The accuracy of the mean numerical trajectories can be quantitatively assessed by defining a  $L_2$  error as in Eq. (84):

$$L_2 = \sqrt{\frac{1}{N} \sum_{i=1}^N \left( \left. \frac{z}{d_{\text{inj}}} \right|_{\text{exp},i} - \left. \frac{z}{d_{\text{inj}}} \right|_{\text{num},i} \right)^2} \quad (84)$$

where  $N$  is the total number of sample points along the trajectory coordinate  $x/d_{\text{inj}}$ . The values for each case are shown in Table 15. As also observed in Fig. 35, the two resolutions for the operating point at low  $We_g$  have very similar errors. For the high  $We_g$  simulations, the coarse mesh shows a larger error, while the fine mesh yields the lowest error of all the simulations.

### 5.2.3 Jet topology and breakup

The effect of mesh resolution (see Table 16) is shown in the instantaneous snapshots of Fig. 36 for the operating point  $We_g = 1470$ . The liquid-gas interface and the mesh at the plane  $y = 0$  are displayed. Both pictures show

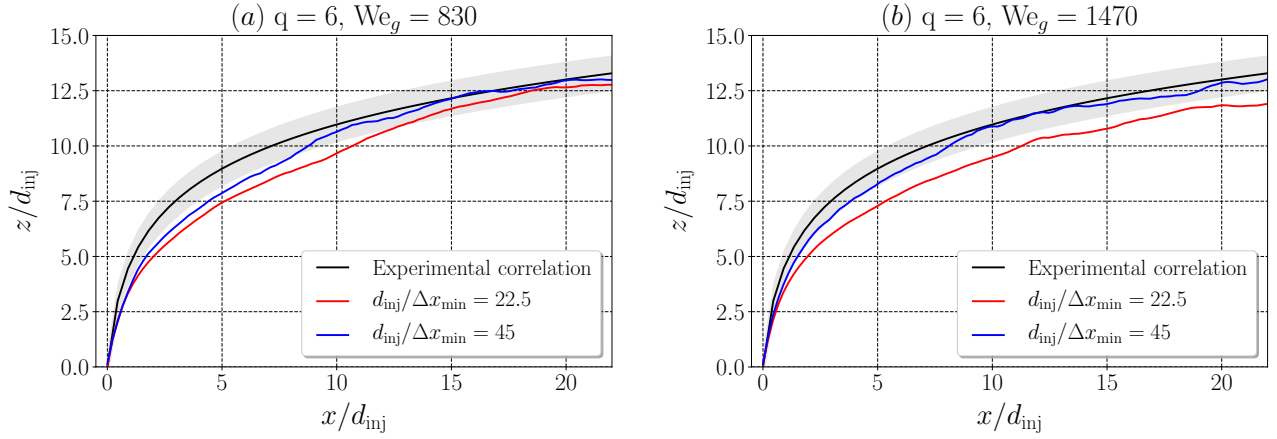


Figure 35: Numerical mean jet trajectories for (a)  $We_g = 830$  and (b)  $We_g = 1470$  compared with the experimental correlation from Becker [32]. The grey area represents the standard deviation.

Table 15: Error levels of the  $L_2$  norm (Eq. (84)) for jet in crossflow simulations

$We_g$	$d_{inj}/\Delta x_{min}$	$L_2$
830	22.5	1.079
830	45	0.661
1470	22.5	1.376
1470	45	0.442

the column and surface breakup features characteristic of jet in crossflow atomization [32]. The interface cell size has a direct effect onto breakup and atomization. Both meshes capture different column breakup modes: the coarse mesh shows Kelvin-Helmholtz instabilities developing in the windward side of the jet, while the fine mesh displays Rayleigh-Taylor instability waves. This directly affects primary atomization, as it can be seen from the different liquid ligaments and structures that result from the breakup of the liquid dense core. The detachment of the liquid jet inside the nozzle observed for the finest resolution also affects the development of surface instabilities that will eventually contribute to breakup. Subsequent secondary atomization is also strongly affected by the resolution, as the finer mesh can capture smaller droplets than the coarse mesh (see Table 16).

Regarding the operating point, Fig. 37 shows instantaneous snapshots of the liquid-gas interface for the finest case ( $d_{inj}/\Delta x_{min} = 45$ ) for both values of  $We_g$ . The same time instant than in Fig. 36 is displayed. By comparing the operating points, it is seen that both cases present similar column breakup features following Rayleigh-Taylor instabilities. The simulation with  $We_g = 830$  shows that ligaments formed during primary atomization stretch and form lobes more pronounced than for  $We_g = 1470$ . Furthermore, the number of wavelengths until breakup is lower for  $We_g = 1470$  than for  $We_g = 830$ . This causes an earlier rupture of the liquid ligaments.

#### 5.2.4 Computational performances

Table 16 shows the number of cores, tetrahedral elements and droplets generated at  $t = 0.3\text{ms}$  for all simulations. By comparing the mesh resolutions for each operating point, it is observed that six times more droplets are generated when the minimum cell size is reduced twice. Thus, as more droplets are resolved, more cores are needed to keep the ratio  $N_{cells}/N_{cores}$  constant for optimal performances: when the resolution increases twice, the number of elements rises near tenfold. When the operating points are compared at identical mesh resolutions, 50% more droplets are present in the highest gaseous-Weber case. This is due to a higher crossflow velocity, which increases the number of droplets generated by surface breakup.

The computational performances and costs are shown in Table 17. The Reduced Computational Times (RCT) are calculated according to Eq. (67). In all cases, the time spent in the ACLS procedure is almost constant and moderate despite the large liquid-gas surface to be resolved, demonstrating the high scalability of the interface-capturing algorithm. Most of the total computational time is spent in the AMR routine: indeed, the jet velocity is high and hence the AMR is automatically triggered much more frequently than in the low-pressure jet simulations of Section 5.1. In general, jet in crossflow simulations are more expensive than the



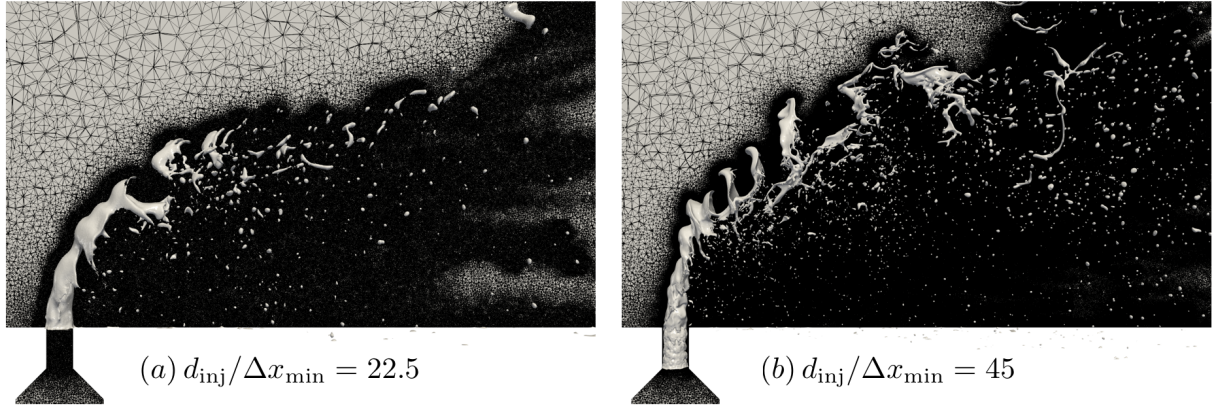


Figure 36: Lateral view of meshes and interface contours near the injector at time instant  $t = 0.3\text{ms}$ .  $q = 6$ .  $We_g = 1470$ .

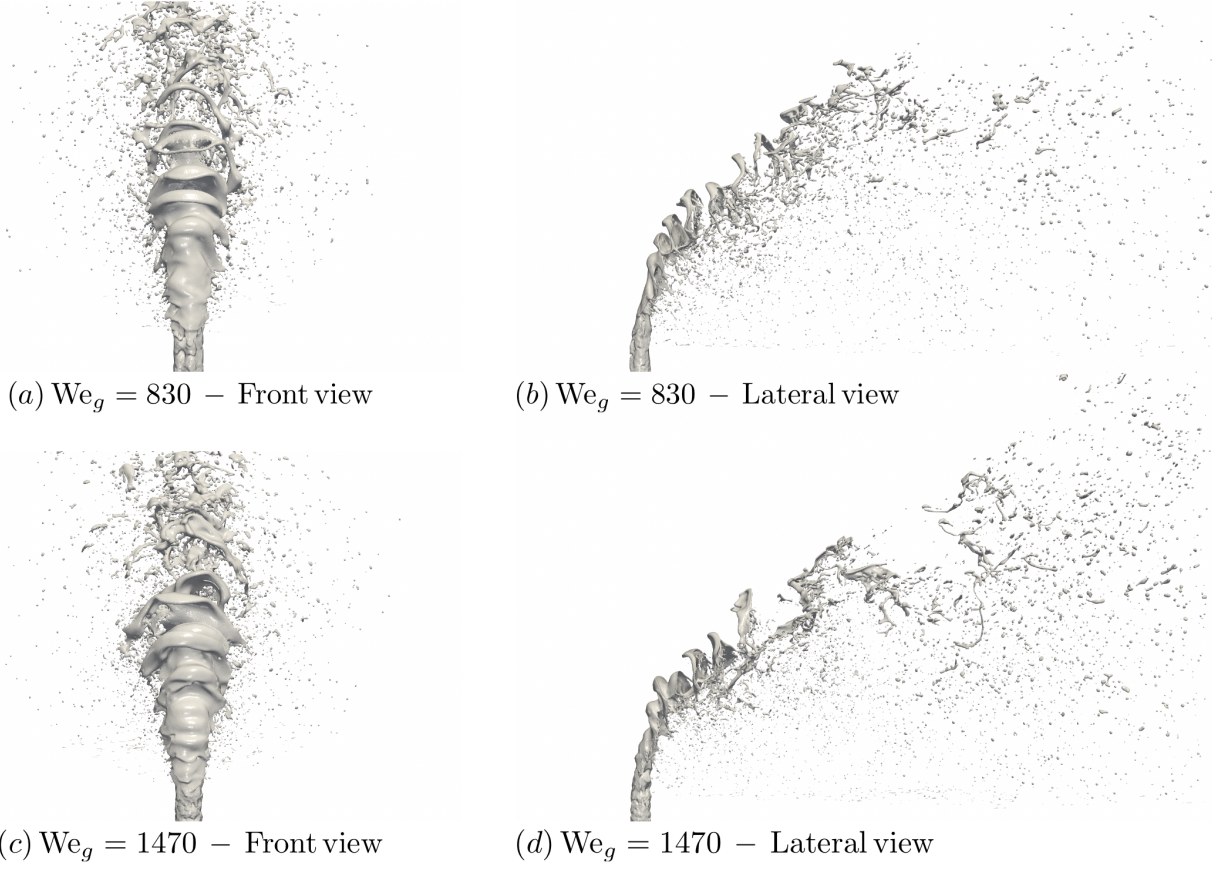


Figure 37: Instantaneous front (left) and lateral (right) views at time instant  $t = 0.3\text{ms}$  for the two operating conditions.  $d_{\text{inj}}/\Delta x_{\text{min}} = 45$ .  $q = 6$ .  $We_g = 830$  (top),  $We_g = 1470$  (bottom).

low-pressure jet ones (see Table 13 for comparison): while this latter jet is governed by capillary forces and aerodynamic effects are negligible, the jet in crossflow atomization is driven by aerodynamics and inertia (the limiting timestep of the simulations is always due to the CFL number).

It can be concluded that most of the computational resources allocated to the jet in crossflow cases are used for solving the AMR routine, making these computations highly expensive for cases where very small droplets are generated and a higher resolution of the atomization is required. A promising solution to this problem is coupling the presented methodology to a Lagrangian solver that could identify spherical droplets and convert them to Lagrangian Point Particles (LPP). This Eulerian-Lagrangian two-way coupling would allow spherical droplets to move following a ballistic motion and break up according to secondary atomization models. In this way, droplets would not be resolved with the Eulerian mesh by the ACLS/AMR methodology, saving huge computational resources.

Table 16: Number of computational cores, mesh cells and droplets generated for jet in crossflow simulations. All values correspond to time instant  $t = 0.3\text{ms}$ .

$We_g$	$d_{\text{inj}}/\Delta x_{\text{min}}$	$N_{\text{cores}}$	$N_{\text{cells}}$	$N_{\text{drops}}$
830	22.5	560	78M	769
830	45	5120	734M	3961
1470	22.5	800	117M	1174
1470	45	5960	876M	6615

Table 17: Computational performances for jet in crossflow simulations at time instant  $t = 0.3\text{ms}$ .

$We_g$	$d_{\text{inj}}/\Delta x_{\text{min}}$	RCT [ $\mu\text{s}$ ]	RCT <sub>ACLS</sub> [ $\mu\text{s}$ ]	RCT <sub>AMR</sub> [ $\mu\text{s}$ ]	RCT <sub>Poisson</sub> [ $\mu\text{s}$ ]	CPU time [h]/Phys. time [ms]
830	22.5	836.1	88.9	586.8	97.4	76,800
830	45	1612.0	116.8	1151.2	167.2	922,100
1470	22.5	833.4	100.4	503.9	152.6	105,600
1470	45	2158.3	113.2	1660.8	190.8	1,200,000

## 6 Conclusion

An efficient and robust overall procedure for the computation of two-phase flows on unstructured meshes has been presented. Spatial convergence and accuracy of the interface capturing technique have been first demonstrated through well-known interface transport tests, both in 2D and 3D, and confirmed by the results obtained on canonical two-phase flow examples. Capillary-driven flows are correctly computed on triangular and tetrahedral grids, despite the challenges that poses the use of these irregular meshes. The accuracy and convergence of the numerical framework presented in this paper can be further improved by using the higher-order finite-volume schemes of [41], which have proven to upgrade the operator discretization on unstructured grids and hence the interface curvature computation.

Adaptive mesh refinement allows significant computational savings, especially in the case of complex turbulent 3D flows: highly-resolved flows with large density ratios in complex geometries using unstructured meshes are now accessible. The results obtained in the low-pressure compound nozzle case demonstrate the potential of the method to access a deep numerical insight of jet instabilities and internal flow dynamics with 3D unstructured meshes. This is a major progress for the computation of realistic industrial flows. To further increase the computational gain, especially in the liquid jet in crossflow simulations where many droplets need to be resolved, implementing an Eulerian-Lagrangian two-way coupling to treat small droplets as lagrangian point particles (LPP) would allow local coarsening of the mesh. This coupling would thus result in much cheaper simulations at equivalent resolutions. Eventually, this multiscale approach would pave the way to both extremely-resolved liquid jet cores using unstructured meshes and accurate droplet statistics far from the injector.

## Acknowledgements

Financial support for the conducted research presented in this article has been provided by the French National Association for Research and Technology ANRT and Safran SA under the CIFRE convention No. 2018/0002,

and by the European Union Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement No. 765998 in the project ANNULIGHT. Computer resources have been provided by CRIANN, France, under the allocation 2012006, and by GENCI, France, under the allocations A0072B06880 and A0092B11072. The authors would also like to acknowledge their academic partners at LEGI laboratory, Grenoble, France, for the fruitful discussions on this topic.

## References

- [1] J. Leparoux, R. Mercier, V. Moureau, H. Musaefendic, Primary atomization simulation applied to a jet in cross-flow aeronautical injector with dynamic mesh adaptation, in: Proceedings ICLASS, 14th Triennial International Conference on Liquid Atomization and Spray Systems, 2018.
- [2] L. Jofre, O. Lehmkuhl, J. Castro, A. Oliva, A 3D Volume-of-Fluid advection method based on cell-vertex velocities for unstructured meshes, *Comput. Fluids* 94 (2014) 14-19.
- [3] L. Jofre, R. Borrell, O. Lehmkuhl, A. Oliva, Parallel load balancing strategy for Volume-of-Fluid methods on 3D unstructured meshes, *J. Comput. Phys.* 282 (2015) 269-288.
- [4] C. Ivey, P. Moin, Accurate interface normal and curvature estimates on three-dimensional unstructured non-convex polyhedral meshes, *J. Comput. Phys.* 300 (2015) 365-386.
- [5] C. Ivey, P. Moin, Conservative and bounded volume-of-fluid advection on unstructured grids, *J. Comput. Phys.* 350 (2017) 387-419.
- [6] F. Evrard, F. Denner, B. van Wachem, Estimation of curvature from volume fractions using parabolic reconstruction on two-dimensional unstructured meshes, *J. Comput. Phys.* 351 (2017) 271-294.
- [7] J.K. Patel, G. Natarajan, A novel consistent and well-balanced algorithm for simulations of multiphase flows on unstructured grids, *J. Comput. Phys.* 350 (2017) 207-236.
- [8] B. Xie, P. Jin, H. Nakayama, S. Liao, F. Xiao, A conservative solver for surface-tension-driven multiphase flows on collocated unstructured grids, *J. Comput. Phys.* 401 (2020) 109025.
- [9] T. Maric, D.B. Kothe, D. Bothe, Unstructured un-split geometrical Volume-of-Fluid methods – A review, *J. Comput. Phys.* 420 (2020) 109695.
- [10] R. Scardovelli, S. Zaleski, Direct numerical simulation of free-surface and interfacial flows, *Annu. Rev. Fluid Mech.* 31 (1999) 567-603.
- [11] M. Herrmann, A balanced force refined level set grid method for two-phase flows on unstructured flow solver grids, *J. Comput. Phys.* 227 (2008) 2674-2706.
- [12] C.E. Kees, I. Akkerman, M.W. Farthing, Y. Bazilevs, A conservative level set method suitable for variable-order approximations and unstructured meshes, *J. Comput. Phys.* 230 (2011) 4536-4558.
- [13] N. Balcazar, L. Jofre, O. Lehmkuhl, J. Castro, J. Rigola, A finite-volume/level-set method for simulating two-phase flows on unstructured grids, *Int. J. Multiph. Flow* 64 (2014) 55-72.
- [14] A. Bøckmann, M. Vartdal, A gradient augmented level set method for unstructured grids, *J. Comput. Phys.* 258 (2014) 47-72.
- [15] S. Lin, J. Yan, D. Kats, G.J. Wagner, A volume-conserving balanced-force level set method on unstructured meshes using a control volume finite element formulation, *J. Comput. Phys.* 380 (2019) 119-142.
- [16] 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.
- [17] J.A. Sethian, P. Smereka, Level Set Methods for Fluid Interfaces, *Annu. Rev. Fluid Mech.* 35 (2003) 341-372.
- [18] S. Popinet, Gerris: a tree-based adaptive solver for the incompressible Euler equations in complex geometries, *J. Comput. Phys.* 190 (2003) 572-600.
- [19] S. Popinet, An accurate adaptive solver for surface-tension-driven interfacial flows, *J. Comput. Phys.* 228 (2009) 5838-5866.
- [20] M. Dai, D.P. Schmidt, Adaptive tetrahedral meshing in free-surface flow, *J. Comput. Phys.* 208 (2005) 228-252.
- [21] N.R. Morgan, J.I. Waltz, 3D level set methods for evolving fronts on tetrahedral meshes with adaptive mesh refinement, *J. Comput. Phys.* 336 (2017) 492-512.



- [22] J.M. Rodriguez, O. Sahni, R.T. Lahey Jr., K.E. Jansen, A parallel adaptive mesh method for the numerical simulation of multiphase flows, *Comput. Fluids* 87 (2013) 115–131.
- [23] E. Olsson, G. Kreiss, A conservative level set method for two phase flow, *J. Comput. Phys.* 210 (2005) 225–246.
- [24] E. Olsson, G. Kreiss, S. Zahedi, A conservative level set method for two phase flow II, *J. Comput. Phys.* 225 (2007) 785–807.
- [25] O. Desjardins, V. Moureau, H. Pitsch, An accurate conservative level set/ghost fluid method for simulating turbulent atomization, *J. Comput. Phys.* 227 (2008) 8395–8416.
- [26] O. Antepara, N. Balcazar, A. Oliva, Tetrahedral adaptive mesh refinement for two-phase flows using conservative level-set method, *Int. J. Numer. Meth. Fluids* (2020) 1–23.
- [27] R. Chiodi, O. Desjardins, A reformulation of the conservative level set reinitialization equation for accurate and robust simulation of complex multiphase flows, *J. Comput. Phys.* 343 (2017) 186–200.
- [28] D. Adalsteinsson, J.A. Sethian, A Fast Level Set Method for Propagating Interfaces, *J. Comput. Phys.* 118 (2) (1995) 269–277.
- [29] V. Moureau, P. Domingo, L. Vervisch, Design of a massively parallel CFD code for complex geometries, *Comptes Rendus Mécanique* 339 (2-3) (2011) 141–148.
- [30] S. Grout, C. Dumouchel, J. Cousin, H. Nuglisch, Fractal analysis of atomizing liquid flows, *Int. J. Multiph. Flow* 33 (9) (2007) 1023–1044.
- [31] J. Cousin, A. Berlemont, T. Menard, S. Grout, Primary breakup simulation of a liquid jet discharged by a low-pressure compound nozzle, *Comput. Fluids* 63 (2012) 165–173.
- [32] J. Becker, C. Hassa, Breakup and atomization of a kerosene jet in crossflow at elevated pressure, *Atomization and Sprays* 11 (2002) 49–67.
- [33] R. Ragucci, A. Bellofiore, A. Cavaliere, Trajectory and momentum coherence breakdown of a liquid jet in high-density air cross-flow, *Atomization and Sprays* 17 (1) (2007) 47–70.
- [34] S. Mirjalili, S.S. Jain, M.S. Dodd, Interface-capturing methods for two-phase flows: An overview and recent developments, *Center for Turbulence Research Annual Research Briefs* (2017) 117–135.
- [35] R. Janodet, G. Vaudor, G. Lartigue, P. Bénard, V. Moureau, R. Mercier, An unstructured conservative level-set algorithm coupled with dynamic mesh adaptation for the computation of liquid-gas flows, in: *Proceedings ILASS-Europe, 29th Conference on Liquid Atomization and Spray Systems*, 2019.
- [36] R. Goldman, Curvature formulas for implicit curves and surfaces, *Comput. Aided Geom. Des.* 22 (2005) 632–658.
- [37] J.M. Delhay, Jump conditions and entropy sources in two-phase systems. Local instant formulation, *Int. J. Multiph. Flow* 1 (1974) 395–409.
- [38] R. Fedkiw, T. Aslam, B. Merriman, S. Osher, A Non-oscillatory Eulerian Approach to Interfaces in Multimaterial Flows (the Ghost Fluid Method), *J. Comput. Phys.* 152 (2) (1999) 457–492.
- [39] M. Malandain, N. Maheu, V. Moureau, Optimization of the deflated Conjugate Gradient algorithm for the solving of elliptic equations on massively parallel machines, *J. Comput. Phys.* 238 (2013) 32–47.
- [40] C. Dapogny, P. Frey, Computation of the signed distance function to a discrete contour on adapted triangulation, *Calcolo* 49 (3) (2010) 1–27.
- [41] M. Bernard, G. Lartigue, G. Balarac, V. Moureau, G. Puigt, A framework to perform high-order deconvolution for finite-volume method on simplicial meshes, *Int. J. Numer. Meth. Fluids* (2020) 1–33.
- [42] M. Kraushaar, Application of the compressible and low-Mach number approaches to Large-Eddy Simulation of turbulent flows in aero-engines, PhD Thesis, Université de Toulouse, 2011.
- [43] A.J. Chorin, The numerical solution of the Navier-Stokes equations for an incompressible fluid, *Bull. Amer. Math. Soc.* 73 (1967) 928–931.
- [44] O. Desjardins, V. Moureau, Methods for multiphase flows with high density ratio, *Center for Turbulence Research Proceedings of the Summer Program* (2010) 313–322.
- [45] P. Bénard, G. Balarac, V. Moureau, C. Dobrzynski, G. Lartigue, Y. D’Angelo, Mesh adaptation for large-eddy simulations in complex geometries, *Int. J. Numer. Meth. Fluids* 81 (2016) 719–740.
- [46] Mmg Platform - Robust, Open-source, and Multidisciplinary Software for Remeshing. <https://www.mmgtools.org>.

- [47] S.T. Zalesak, Fully multidimensional flux-corrected transport algorithms for fluids, *J. Comput. Phys.* 31 (3) (1979) 335-362.
- [48] M. Owkes, O. Desjardins, A mass and momentum conserving unsplit semi-Lagrangian framework for simulating multiphase flows, *J. Comput. Phys.* 332 (2017) 21-46.
- [49] R.J. LeVeque, High-resolution conservative algorithms for advection in incompressible flow, *SIAM J. Numer. Anal.* 33 (2) (1996) 627-665.
- [50] A. Prosperetti, Motion of two superposed viscous fluids, *Phys. Fluids* 24 (1981) 1217-1223.
- [51] T. Menard, S. Tanguy, A. Berlemont, Coupling level set/VOF/ghost fluid methods: Validation and application to 3D simulation of the primary break-up of a liquid jet, *Int. J. Multiph. Flow* 33 (2007) 510-524.
- [52] C. Weber, Zum Zerfall eines Flüssigkeitsstrahles, *Z. Angew. Math. Mech.* 11 (1931) 136-141.
- [53] A.M. Sterling, C.A. Sleicher, The instability of capillary jets, *J. Fluid Mech.* 68(3) (1975) 477-495.
- [54] N. Ashgriz, J.Y. Poo, Coalescence and separation in binary collisions of liquid drops, *J. Fluid Mech.* 221 (1990) 183-204.
- [55] M. Germano, U. Piomelli, P. Moin, W.H. Cabot, A dynamic subgrid-scale eddy viscosity model, *Phys. Fluids A* 3 (7) (1991) 1760-1765.
- [56] D.K. Lilly, A proposed modification of the Germano subgrid-scale closure method, *Phys. Fluids A* 4 (3) (1991) 633-635.