

An accurate adaptive solver for surface-tension-driven interfacial flows

Stéphane Popinet

▶ **To cite this version:**

Stéphane Popinet. An accurate adaptive solver for surface-tension-driven interfacial flows. Journal of Computational Physics, 2009, 228, pp.5838 - 5866. 10.1016/j.jcp.2009.04.042 . hal-01445445

HAL Id: hal-01445445
https://hal.science/hal-01445445

Submitted on 24 Jan 2017

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.

An accurate adaptive solver for surface-tension-driven interfacial flows

Stéphane Popinet

National Institute of Water and Atmospheric Research
P.O. Box 14-901, Kilbirnie, Wellington, New Zealand

Email: s.popinet@niwa.co.nz

April 2, 2009

Abstract

A method combining an adaptive quad/octree spatial discretisation, geometrical Volume-Of-Fluid interface representation, balanced-force continuum-surface-force surface tension formulation and height-function curvature estimation is presented. The extension of these methods to the quad/octree discretisation allows adaptive variable resolution along the interface and is described in detail. The method is shown to recover exact equilibrium (to machine accuracy) between surface-tension and pressure gradient in the case of a stationary droplet, irrespective of viscosity and spatial resolution. Accurate solutions are obtained for the classical test case of capillary wave oscillations. An application to the capillary breakup of a jet of water in air further illustrates the accuracy and efficiency of the method. The source code of the implementation is freely available as part of the Gerris flow solver.

AMS: 65M50; 76D05; 76D25; 76D45; 76M12; 76T10

Keywords: Adaptive mesh refinement; Surface tension; Navier-Stokes equations; Height-Function; Parasitic currents; Volume-Of-Fluid; Octree

1 Introduction

Flows of immiscible fluids are ubiquitous in Nature: waves on the sea, waterfalls, rain drops or a garden sprinkler are familiar examples. In general air-water flows exhibit the typical features associated with many important two-phase-flow phenomena: relatively large density ratios, high surface-tension and low viscosity at practical length scales. Furthermore most of these flows tend to generate complex and evolving interface geometries on spatial scales ranging over several orders of magnitude. This is true not only of obviously complex phenomena such as wave breaking or jet atomisation, but also of the apparently simpler sub-phenomena such as jet/droplet breakup and coalescence.

Numerical methods using an implicit representation of the interface such as Volume-Of-Fluid (VOF) [1, 2, 3, 4] or Levelset (LS) [5, 6, 7, 8, 9, 10, 11, 12, 13] can robustly and efficiently represent evolving, topologically complex interfaces. The accurate representation of surface tension within these methods is typically more delicate than for schemes using an explicit representation of the interface [14, 15, 16, 17] and care must be taken to avoid any imbalance between discrete surface tension and pressure gradient terms [18, 19, 15, 20, 21, 22, 23]. For VOF methods the accurate evaluation of the geometrical properties of the interface such as curvature is also an active research topic [24, 21, 25, 23, 26]. Finally dynamic mesh-adaptive methods can deal efficiently with phenomena involving a wide range of spatial scales [27, 28, 29, 6, 7, 30, 8, 31, 17, 9, 32, 33, 34].

In the following I will describe a new combination of a VOF and quad/octree adaptive mesh refinement method. The basis for the scheme is the adaptive, incompressible quad/octree Gerris solver described in [35]. It is coupled with a classical geometrical VOF scheme which is generalised to work on the quad/octree in order to allow variable spatial resolution along the interface. This is in contrast with many previous implementations of adaptive methods with VOF or LS which restricted adaptivity to regions away from the interface (i.e. resolution along the interface was kept constant) [28, 7, 36, 30, 8, 31, 37, 9, 38, 39, 33, 40]. A recent article by Malik et al [41] describes a two-dimensional quadtree implementation allowing variable resolution along the interface but their study is limited to the advection problem (not coupled to the momentum equation). Block-structured AMR and levelset methods were used in [6, 29] to provide variable resolution along the interface. Here we present a novel quad/octree discretisation.

Surface tension representation draws on the "balanced-forced" continuum surface force (CSF) concept introduced by Renardy and Renardy [21] and Francois et al [23], coupled with a height-function (HF) curvature calculation [42, 25] generalised for the quad/octree. The classical HF curvature calculation becomes inconsistent at low interface resolution. This is addressed by a generalised version of the HF curvature concept.

The robustness and accuracy of the generalised surface-tension method is assessed for classical test cases: curvature estimates, stationary droplet with surface tension, capillary waves

and rising bubbles and compared with published results from other methods. Particular attention is paid to the issue of "parasitic currents" around a stationary droplet [18, 15, 23, 43, 26, 40], and we improve upon existing methods by enabling a unique combination of varying resolution along the interface, improved height-function curvature discretisation and a balanced-force approach. Finally the method is applied to the adaptive solution of the surface-tension-driven breakup of a three-dimensional cylindrical jet of water in air.

The full implementation of the method as well as most of the test cases presented in this article are available under the General Public License on the Gerris web site [44, 45].

2 Numerical scheme

The numerical scheme is a direct extension of the scheme described in [35]. Consequently I will only give a summary of the elements of the method which are not specific to the Volume-Of-Fluid and surface tension implementations. Details regarding the quad/octree spatial discretisation, multilevel Poisson solver, approximate projection and momentum advection scheme can be found in [35].

2.1 Temporal discretisation

The incompressible, variable-density, Navier–Stokes equations with surface tension can be written

$$\begin{aligned} \rho\left(\partial_t \boldsymbol{u} + \boldsymbol{u}\cdot\boldsymbol{\nabla}\boldsymbol{u}\right) &= -\boldsymbol{\nabla} p + \boldsymbol{\nabla}\cdot\left(2\,\mu\,\boldsymbol{D}\right) + \sigma\,\kappa\,\delta_s\,\boldsymbol{n}, \\ \partial_t \rho + \boldsymbol{\nabla}\cdot\left(\rho\,\boldsymbol{u}\right) &= 0, \\ \boldsymbol{\nabla}\cdot\boldsymbol{u} &= 0, \end{aligned}$$

with $\boldsymbol{u} = (u,\, v,\, w)$ the fluid velocity, $\rho \equiv \rho(\boldsymbol{x},\, t)$ the fluid density, $\mu \equiv \mu(\boldsymbol{x},\, t)$ the dynamic viscosity and $\boldsymbol{D}$ the deformation tensor defined as $D_{\mathrm{ij}} \equiv (\partial_i u_j + \partial_j u_i)/2$. The Dirac distribution function δ_s expresses the fact that the surface tension term is concentrated on the interface; σ is the surface tension coefficient, κ and $\boldsymbol{n}$ the curvature and normal to the interface.

For two-phase flows we introduce the volume fraction $c(\boldsymbol{x}, t)$ of the first fluid and define the density and viscosity as

$$\rho(\tilde{c}) \equiv \tilde{c}\,\rho_1 + (1-\tilde{c})\,\rho_2, \tag{1}$$

$$\mu(\tilde{c}) \equiv \tilde{c}\,\mu_1 + (1-\tilde{c})\,\mu_2, \tag{2}$$

with ρ_1, ρ_2 and μ_1, μ_2 the densities and viscosities of the first and second fluids respectively. Field $\tilde{c}$ is either identical to c or is constructed by applying a smoothing spatial filter to c. Using a smoothed field to define the viscosity was found to improve the results for some of the test cases I studied. Unless otherwise specified spatial filtering is not used in the

applications presented in the following. When spatial filtering is used, field $\tilde{c}$ is constructed by averaging the four (respectively eight in 3D) corner values of c obtained by bilinear interpolation from the cell-centred values. When spatial filtering is used, the properties associated with the interface are thus "smeared" over three discretisation cells.

The advection equation for the density can then be replaced with an equivalent advection equation for the volume fraction

$$\partial_t c + \boldsymbol{\nabla}\cdot(c\,\boldsymbol{u}) = 0.$$

A staggered in time discretisation of the volume-fraction/density and pressure leads to the following formally second-order accurate time discretisation

$$\rho_{n+\frac{1}{2}}\left[\frac{\boldsymbol{u}_{n+1}-\boldsymbol{u}_n}{\Delta t} + \boldsymbol{u}_{n+\frac{1}{2}}\cdot\boldsymbol{\nabla}\boldsymbol{u}_{n+\frac{1}{2}}\right] = -\boldsymbol{\nabla}p_{n+\frac{1}{2}} + \boldsymbol{\nabla}\cdot\left[\mu_{n+\frac{1}{2}}(\boldsymbol{D}_n+\boldsymbol{D}_{n+1})\right] + (\sigma\,\kappa\,\delta_s\,\boldsymbol{n})_{n+\frac{1}{2}},$$
$$\frac{c_{n+\frac{1}{2}}-c_{n-\frac{1}{2}}}{\Delta t} + \boldsymbol{\nabla}\cdot(c_n\,\boldsymbol{u}_n) = 0,$$
$$\boldsymbol{\nabla}\cdot\boldsymbol{u}_n = 0$$

This system is further simplified using a classical time-splitting projection method [46]

$$\rho_{n+\frac{1}{2}}\left[\frac{\boldsymbol{u}_\star-\boldsymbol{u}_n}{\Delta t} + \boldsymbol{u}_{n+\frac{1}{2}}\cdot\boldsymbol{\nabla}\boldsymbol{u}_{n+\frac{1}{2}}\right] = \boldsymbol{\nabla}\cdot\left[\mu_{n+\frac{1}{2}}(\boldsymbol{D}_n+\boldsymbol{D}_\star)\right] + (\sigma\,\kappa\,\delta_s\,\boldsymbol{n})_{n+\frac{1}{2}}, \tag{3}$$
$$\frac{c_{n+\frac{1}{2}}-c_{n-\frac{1}{2}}}{\Delta t} + \boldsymbol{\nabla}\cdot(c_n\,\boldsymbol{u}_n) = 0, \tag{4}$$
$$\boldsymbol{u}_{n+1} = \boldsymbol{u}_\star - \frac{\Delta t}{\rho_{n+\frac{1}{2}}}\boldsymbol{\nabla}p_{n+\frac{1}{2}}, \tag{5}$$
$$\boldsymbol{\nabla}\cdot\boldsymbol{u}_{n+1} = 0$$

which requires the solution of the Poisson equation

$$\boldsymbol{\nabla}\cdot\left[\frac{\Delta t}{\rho_{n+\frac{1}{2}}}\boldsymbol{\nabla}p_{n+\frac{1}{2}}\right] = \boldsymbol{\nabla}\cdot\boldsymbol{u}_\star \tag{6}$$

This equation is solved using the quad/octree-based multilevel solver described in detail in [35]. The iterative solution procedure is stopped whenever the maximum relative change in the volume of any discretisation element (due to the remaining divergence of the velocity field) is less than a given threshold γ_p. Unless otherwise specified this threshold is set to 10^{-3}. The number of Gauss–Seidel relaxations per level is set to four.

It is well-known that the standard multigrid scheme can exhibit slow convergence in the case of elliptic equations with discontinuous coefficients and/or source terms [47, 48, 49]. Depending on the problem and interface topology, this can lead to an important degradation of performance for large-density-ratio flows. While this was not an issue for the applications

presented in this paper, it is an aspect of the method which warrants further improvements.

The discretised momentum equation (3) can be re-organised as

$$\frac{\rho_{n+\frac{1}{2}}}{\Delta t}\,\boldsymbol{u}_\star - \boldsymbol{\nabla}\cdot\left[\mu_{n+\frac{1}{2}}\,\boldsymbol{D}_\star\right] = \boldsymbol{\nabla}\cdot\left[\mu_{n+\frac{1}{2}}\,\boldsymbol{D}_n\right] + (\sigma\,\kappa\,\delta_s\,\boldsymbol{n})_{n+\frac{1}{2}} + \rho_{n+\frac{1}{2}}\left[\frac{\boldsymbol{u}_n}{\Delta t} - \boldsymbol{u}_{n+\frac{1}{2}}\cdot\boldsymbol{\nabla}\boldsymbol{u}_{n+\frac{1}{2}}\right] \tag{7}$$

where the right-hand-side depends only on values at time n and $n + 1/2$. This is an Helmholtz-type equation which can be solved using a variant of the multilevel Poisson solver. The resulting Crank–Nicholson discretisation of the viscous terms is formally second-order accurate and unconditionally stable. The criterion for convergence of the multilevel solver is the relative error γ_u in each component of the velocity field. Unless otherwise specified this threshold is set to 10^{-6}. The number of Gauss–Seidel relaxations per level is set to four.

The velocity advection term $\boldsymbol{u}_{n+\frac{1}{2}}\cdot\boldsymbol{\nabla}\boldsymbol{u}_{n+\frac{1}{2}}$ is estimated using the Bell–Colella–Glaz second-order unsplit upwind scheme [50, 35]. This scheme is stable for CFL numbers smaller than one.

2.2 Spatial discretisation

Space is discretised using a graded quadtree partitioning (octree in three dimensions). We refer the reader to [35] and references therein for a more detailed presentation of this data structure and just give a summary of the definitions necessary for the description of the algorithms presented in this article:

Root cell. The base of the cell tree. The root cell does not have a parent cell and all cells in the tree are descendants of the root cell.

Children cells. The direct descendants of a cell. Cells other than leaf cells have four children in two dimensions (quadtree) and eight in three dimensions (octree).

Parent cell. The direct ancestor of a given cell.

Leaf cells. The highest cells in the cell tree. Leaf cells do not have children.

Cell level. By convention the root cell is at level zero and each successive generation increases the cell level by one.

Coarser cell. Cell $\mathcal{A}$ is coarser than cell $\mathcal{B}$ if $\text{level}(\mathcal{A}) < \text{level}(\mathcal{B})$ and conversely for finer cells.

All the variables (components of the momentum, pressure and passive tracers) are collocated at the centre of each square in 2D (resp. cubic in 3D) discretisation volume. Consistently with a finite-volume formulation, the variables are interpreted as the volume-averaged values for the corresponding discretisation volume. The choice of a collocated definition of all variables makes momentum conservation simpler when dealing with mesh adaptation [35]. It is also a necessary choice in order to use the Godunov momentum advection scheme of Bell, Colella and Glaz [50], and it simplifies the implementation of the Crank–Nicholson discretisation of the viscous terms; however one has to be careful to avoid the classic problem of decoupling of the pressure and velocity field.

To do so, an approximate projection method [51, 35] is used for the spatial discretisation of the pressure correction equation (5) and the associated divergence in the Poisson equation (4). In a first step the auxiliary cell-centred velocity field $\boldsymbol{u}_\star^c$ is computed using equation (7). An auxiliary face-centred velocity field $\boldsymbol{u}_\star^f$ is then computed using averaging of the cell-centred values on all the faces of the Cartesian discretisation volumes. When faces are at the boundary between different levels of refinement of the quad/octree mesh, averaging is performed so as to guarantee consistency of the corresponding volume fluxes (see [35] for details).

The divergence of the auxiliary velocity field appearing on the right-hand-side of equation (6) is then computed for each control volume as the finite-volume approximation

$$\boldsymbol{\nabla} \cdot \boldsymbol{u}_\star = \frac{1}{\Delta} \sum_f \boldsymbol{u}_\star^f \cdot \boldsymbol{n}^f,$$

with $\boldsymbol{n}^f$ the unit normal vector to the face and Δ the length scale of the control volume.

After solving equation (6), the pressure correction is applied to the face-centred auxiliary field

$$\boldsymbol{u}_{n+1}^f = \boldsymbol{u}_\star^f - \frac{\Delta t}{\rho^f_{n+\frac{1}{2}}} \boldsymbol{\nabla}^f p_{n+\frac{1}{2}}, \tag{8}$$

where $\boldsymbol{\nabla}^f$ is a simple face-centred gradient operator (consistent at coarse/fine volume boundaries, see section 4.1 of [35]). The resulting face-centred velocity field $\boldsymbol{u}_{n+1}^f$ is exactly non-divergent by construction.

The cell-centred velocity field at time $n+1$ is obtained by applying a cell-centred pressure correction

$$\boldsymbol{u}_{n+1}^c = \boldsymbol{u}_\star^c - \left| \frac{\Delta t}{\rho^f_{n+\frac{1}{2}}} \boldsymbol{\nabla}^f p_{n+\frac{1}{2}} \right|^c, \tag{9}$$

where the $||^c$ operator denotes averaging over all the faces delimiting the control volume. The resulting cell-centred velocity field $\boldsymbol{u}_{n+1}^c$ is approximately divergence-free.

The Poisson equation (6), advection–diffusion equation (7) and pressure corrections (8) and (9) require an estimate of the face-centred values of the density $\rho^f_{n+1/2}$ or viscosity $\mu^f_{n+1/2}$. Both quantities are obtained using the volume-fraction-weighted averages (1) and (2) which in turn requires an estimate of $c^f_{n+1/2}$. This face-centred volume fraction is estimated using a simple average of the cell-centred values in the case of neighbouring cells at the same level. At fine–coarse and coarse–fine boundaries a second-order interpolation stencil is used, identical to the stencil used to evaluate the face-centred gradient operator $\boldsymbol{\nabla}^f$ (see section 4.1 of [35]). Other choices are possible, including a local face-centred VOF

reconstruction which would eliminate the smoothing effect of the interpolation stencil at coarse–fine boundaries. Preliminary tests suggest that the interpolation procedure works well in practice however.

3 Volume Of Fluid advection scheme

To solve the advection equation (4) for the volume fraction we use a piecewise-linear geometrical Volume Of Fluid (VOF) scheme [4] generalised for the quad/octree spatial discretisation. Geometrical VOF schemes classically proceed in two steps:

1. Interface reconstruction
2. Geometrical flux computation and interface advection

3.1 Interface reconstruction

In a piecewise-linear VOF scheme the interface is represented in each cell by a line (resp. plane in three dimensions) described by equation

$$\boldsymbol{m} \cdot \boldsymbol{x} = \alpha, \tag{10}$$

where $\boldsymbol{m}$ is the local normal to the interface and $\boldsymbol{x}$ is the position vector. Given $\boldsymbol{m}$ and the local volume fraction c, α is uniquely determined by ensuring that the volume of fluid contained in the cell and lying below the plane is equal to c. This volume can be computed relatively easily by taking into account the different ways a square (resp. cubic) cell can be cut by a line (resp. plane) which leads to matched linear and quadratic (resp. cubic) functions of α. This step has been described in detail in several papers [19, 52]. In what follows we will assume that given a normal $\boldsymbol{m}$, a volume fraction c and coefficient α of equation (10) defined in a coordinate system with origin at the cell centre and in which the cell size is unity, we have the relations:

$$\begin{aligned} c &= \mathcal{V}(\boldsymbol{m}, \alpha), \\ \alpha &= \mathcal{V}^{-1}(\boldsymbol{m}, c). \end{aligned}$$

In practice, we use the $\mathcal{V}$ and $\mathcal{V}^{-1}$ routines implemented by Ruben Scardovelli, based on analytical formulae for the solution of quadratic (resp. cubic) equations [52]. The routines have been well tested and are free from inconsistencies (such as $\mathcal{V}(\boldsymbol{m}, \mathcal{V}^{-1}(\boldsymbol{m}, c)) \neq c$) which may occur due to round-off errors in limiting cases.

A number of schemes have been proposed for interface normal estimation [4, 53, 54]. Most of these schemes only require information in a *compact neighbourhood* of the cell considered: typically a 3×3 (resp. $3 \times 3 \times 3$) regular Cartesian stencil. Given this stencil, the methods use finite-difference estimates and/or minimisation techniques and return an estimate for $\boldsymbol{m}$. Generalising this approach to a quad/octree discretisation is trivial if the interfacial

region (i.e. the compact neighbourhood) is resolved with a constant resolution. In this case the local discretisation simply reduces to a regular Cartesian discretisation [30, 31, 36].

In order to generalise the method to variable spatial resolution along the interface, we need to be able to reconstruct a local regular Cartesian stencil when cells in the compact neighbourhood vary in size. For the sake of clarity we describe the stencil-computation algorithm for two-dimensional quadtrees. The extension to three dimensions is straightforward.

Given a reference cell $\mathcal{C}$ centred on $\boldsymbol{x}_0$, of size Δ, and containing the interface ($0 < c < 1$), the stencil-computation algorithm proceeds as follows:

Algorithm 1 *Stencil computation ($\mathcal{C}$)*

1. *set* $\boldsymbol{\mathcal{X}} \longleftarrow (1,0)$, $\boldsymbol{\mathcal{Y}} \longleftarrow (0,1)$.
2. ***for each*** *position* $\boldsymbol{x}_{i,j} \longleftarrow \boldsymbol{x}_0 + \Delta\,(i\,\boldsymbol{\mathcal{X}} + j\,\boldsymbol{\mathcal{Y}})$, $-1 \leqslant i \leqslant 1$, $-1 \leqslant j \leqslant 1$
   a) *locate the smallest cell* $\mathcal{N}$ *of size larger than or equal to* Δ *containing* $\boldsymbol{x}_{i,j}$
   b) *set the stencil volume fraction* $c_{i,j}$ *to the volume fraction of a (virtual) cell* $\mathcal{N}_\Delta$ *of size* Δ*, centred on* $\boldsymbol{x}_{i,j}$ *and entirely contained in* $\mathcal{N}$

Step 1.2.a of the algorithm is performed using a conditional traversal of the quadtree starting from the root cell and stopping whenever either a leaf cell is reached or the size of the cell is equal to Δ. This leads to an efficient point-location algorithm with cost of order $\mathcal{O}(\log n)$ where n is the number of leaf cells in the quadtree.

For step 1.2.b, several cases must be considered:

Algorithm 2 *Equivalent volume fraction ($\mathcal{C}$,$\boldsymbol{x}_{i,j}$,$\mathcal{N}$)*

1. *the size of* $\mathcal{N}$ *is equal to the size of* $\mathcal{C}$*, then* $\mathcal{N}_\Delta = \mathcal{N}$ *and* $c_{i,j}$ *is simply set to the value of the volume fraction in cell* $\mathcal{N}$
2. *the size of* $\mathcal{N}$ *is larger than* Δ *(i.e.* $\mathcal{N}$ *is a coarse neighbour of* $\mathcal{C}$ *and is also a leaf cell)*
   a) $\mathcal{N}$ *does not contain the interface:* $c_{i,j}$ *is set to the volume fraction in cell* $\mathcal{N}$ *(0 or 1)*
   b) $\mathcal{N}$ *contains the interface: by definition* $\mathcal{N}_\Delta$ *does not exist in this case and the equivalent volume fraction* $c_{i,j}$ *needs to be computed from the interface reconstructed in cell* $\mathcal{N}$

Note that in case 2.1, cell $\mathcal{N}$ is not necessarily a leaf cell, however consistent values for the volume fraction can easily be defined on all levels of the quadtree. This is done before interface reconstruction by traversing the quadtree from the leaf cells to the root cell and setting the volume fractions of non-leaf cells to the average of their children's volume fractions.

For the treatment of case 2.2.b, we will assume that the equation $\boldsymbol{m} \cdot \boldsymbol{x} = \alpha_\mathcal{N}$ is known for the fragment of interface contained in cell $\mathcal{N}$. In order to be able to compute the equivalent volume fraction of the virtual cell $\mathcal{N}_\Delta$ using function $\mathcal{V}$, we need to transform the interface equation into the coordinate system centred on $\mathcal{N}_\Delta$. The coordinate transformation is simply

$$\boldsymbol{x}' = \frac{n\,\Delta\,\boldsymbol{x} + \boldsymbol{x}_\mathcal{N} - \boldsymbol{x}_{i,j}}{\Delta}, \tag{11}$$

where $n\ \Delta$ and $\boldsymbol{x}_{\mathcal{N}}$ are the size and position of cell $\mathcal{N}$. The interface equation in the coordinate system of the virtual cell $\mathcal{N}_\Delta$ is then

$$\boldsymbol{m}\cdot\boldsymbol{x}' = n\,\alpha_{\mathcal{N}} + \frac{\boldsymbol{m}}{\Delta}\cdot(\boldsymbol{x}_{\mathcal{N}} - \boldsymbol{x}_{i,j}), \tag{12}$$

and the corresponding stencil volume fraction is given by

$$c_{i,j} = \mathcal{V}(\boldsymbol{m}, n\,\alpha_{\mathcal{N}} + \frac{\boldsymbol{m}}{\Delta}\cdot(\boldsymbol{x}_{\mathcal{N}} - \boldsymbol{x}_{i,j})). \tag{13}$$

Note that in practice we restrict the quad/octree discretisation so that the sizes of neighbouring cells never differ by more than a factor of two [35] i.e. n is always equal to two in the expressions above.

Once the $c_{i,j}$ stencil has been filled, a standard Cartesian routine for computing the interface normal $\boldsymbol{m}$ is called and the interface is reconstructed by evaluating $\alpha = \mathcal{V}^{-1}(\boldsymbol{m}, c_{0,0})$. In practice we use the Mixed-Youngs-Centred (MYC) implementation of Aulisa et al [54] to evaluate $\boldsymbol{m}$.

As should be clear from the description of algorithm 2, reconstructing an interface in a cell will require access to the reconstructed interface in any coarser neighbouring cell, however access to cells finer than the cell considered is never required. Taking this into account, the general interface reconstruction algorithm can be summarised as:

Algorithm 3 *Interface reconstruction*

1. ***for each*** *non-leaf cell (traversing from leaf to root)*
   a) *set the volume fraction as the average of the children's volume fractions*
2. ***for each*** *cell $\mathcal{C}$ containing the interface (traversing from root to leaf)*
   a) *fill the $c_{i,j}$ regular Cartesian stencil using Algorithm 1*
   b) *compute $\boldsymbol{m}$ using $c_{i,j}$ and the* MYC *scheme*
   c) *compute $\alpha = \mathcal{V}^{-1}(\boldsymbol{m}, c_{0,0})$*
   d) *store $\boldsymbol{m}$ and α as state variables of cell $\mathcal{C}$*

3.2 Interface advection and geometrical flux computation

The basis for volume fraction advection is the direction-split scheme originally implemented by DeBar [55] and re-implemented by others [42, 3, 56]. Considering a single two-dimensional Cartesian cell, one timestep of this scheme can be summarised as

$$c^{i,j}_{\star}\,V^{i,j}_{\star} = c^{i,j}_{n-1/2}\,V^{i,j}_{n-1/2} + \phi^{i-1/2,j}_{n-1/2} - \phi^{i+1/2,j}_{n-1/2}, \tag{14}$$

$$c^{i,j}_{n+1/2}\,V^{i,j}_{n+1/2} = c^{i,j}_{\star}\,V^{i,j}_{\star} + \phi^{i,j-1/2}_{\star} - \phi^{i,j+1/2}_{\star}, \tag{15}$$

where $V^{i,j}$ is the *effective volume* of the cell and $\phi^{i+1/2,j}$ is the volume flux of the first phase through the right-hand boundary of the cell (respectively left-hand, top and bottom boundaries for the other indices). The effective volume is defined by taking into account the compression or expansion of the cell due to the divergent one-dimensional velocity field, which gives

$$\begin{aligned} V_\star^{i,j} &\equiv V_{n-1/2}^{i,j} + (u_n^{i-1/2,j} - u_n^{i+1/2,j})\,\Delta\,\Delta t, \\ V_{n+1/2}^{i,j} &\equiv V_\star^{i,j} + (u_n^{i,j-1/2} - u_n^{i,j+1/2})\,\Delta\,\Delta t. \end{aligned}$$

Since the face-centred velocity field $\boldsymbol{u}_n^f$ is discretely non-divergent, the second relation reduces to $V_{n+1/2}^{i,j} \equiv V_{n-1/2}^{i,j} \equiv \Delta^2$. Also, in practice the order of the split advection directions is reversed at each timestep. In two dimensions, equations (14) and (15) are equivalent to the Eulerian-implicit and Eulerian-explicit formulations of [53] respectively and the scheme as a whole is identical to the Eulerian-implicit-explicit scheme described by [3]. Extension of this scheme to three dimensions is straightforward.

Several schemes have been proposed for the computation of the volume fluxes ϕ. Classical analytical schemes can be used in principle but usually lead to numerical diffusion of the interface. The original Volume-Of-Fluid scheme of Hirt and Nichols [1] relied on a combination of diffusive and anti-diffusive advection schemes to limit interface diffusion, an idea initially proposed by Zalesak [57] and subsequently expanded by others [42, 58]. In contrast to this analytical approach, volume fluxes can also be estimated using the geometry of the reconstructed interface [55, 59, 2]. This geometrical approach is efficient and simple to implement for Cartesian discretisation elements but is not easily adapted to more general spatial discretisations. As will be shown below this is not an issue when using a quad/octree spatial discretisation.

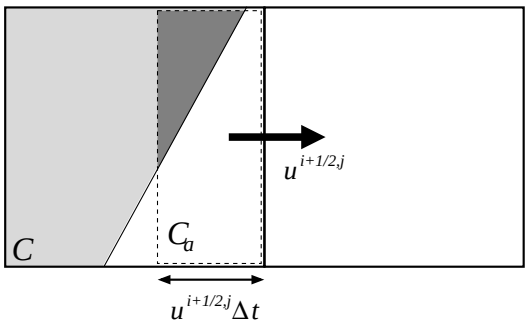


Fig. 1. Geometrical flux estimation.

The principle of geometrical flux estimation is illustrated in Figure 1. The total volume which will be fluxed to the right-hand neighbour is delimited with a dashed line. The fraction of this volume occupied by the first phase is indicated by the dark grey triangle. By definition the area of this triangle is an estimate of the volume flux $\phi^{i+1/2,j}$. This area can easily be calculated using an approach similar to that presented in the preceding section (relations (11), (12) and (13)).

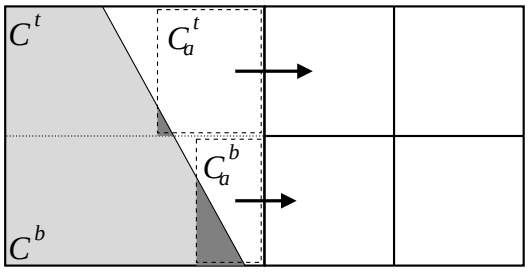


Fig. 2. Special case for geometrical flux computation on a quadtree discretisation.

Generalising this advection scheme to a quad/octree discretisation only involves dealing with the special case illustrated in Figure 2. In this case cell C is a coarse cell and either or both volume fluxes from C to its finer neighbours are positive. The fluxes then need to be computed independently for the top- and bottom-halves of the coarse cell (C^t and C^b in Figure 2). The equation of the interface in both halves can be determined easily using relations similar to (11), (12) and (13). Once this is done, calculation of the fluxes in each half-cell is identical to the standard case.

As has been noted before [3], this VOF advection scheme is not strictly conservative because small over- or under-shoots of the volume fraction may occur for each direction-sweep. Since interface reconstruction and geometrical flux calculation depend on consistent (i.e. bounded between zero and one) volume fractions, it is necessary to reset any volume fraction violating this constraint before proceeding to the next direction sweep, which will lead to loss of exact mass conservation. In practice we have found that this scheme is simple to implement on quad/octrees with mass conservation usually not being an issue, however it would be interesting to investigate the extension to quad/octrees of flux-redistribution schemes [3, 56] or exactly mass-preserving geometrical schemes [53, 60].

3.3 Conservation of interface spatial resolution

Depending on the criterion used for spatial adaptivity there may be cases when the interface spatial resolution could change during interface advection. Given two neighbouring cells C^i and C^{i+1} this would happen whenever the following conditions are met:

1. C^{i+1} is coarser than C^i,
2. $u^{i+1/2} > 0$,
3. C^{i+1} does not contain the interface (i.e. $c^{i+1} = 0$ or $c^{i+1} = 1$),
4. $\phi^{i+1/2} \neq c^{i+1}$.

In this case the interface fragment contained in cell C^i will be advected to the coarser cell C^{i+1} resulting in a loss of spatial resolution of the interface.

In order to avoid this problem all cells verifying the conditions above are refined prior to interface advection for each of the direction sweeps. Initialising the volume fractions of the new children cells of C^{i+1} is trivial since by definition $c^{i+1} = 0$ or $c^{i+1} = 1$ (condition 3),

however initialising the face-centred velocities $\boldsymbol{u}^f$ of the new cells requires some thought.

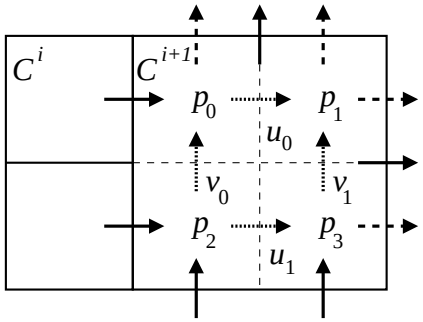


Fig. 3. Local construction of discretely divergence-free face-centred velocities.

The face-centred velocities need to be discretely divergence-free (in order to ensure mass conservation during interface advection). They are usually obtained through the projection step (8) of section 2.2. In the present case however it would not be practical to reproject the whole face-centred velocity field just to get the new face-centred velocities in the few cells requiring refinement. A simple approach is to use a *local projection* to build the face-centred velocity field locally (Figure 3). In a first step the new face-centred velocities on the *boundary* of C^{i+1} (dashed or solid arrows) are initialised by copying the existing face-centred velocities of C^{i+1} and its neighbours (solid arrows). The second step initialises the remaining *interior* face-centred velocities u_0, u_1, v_0 and v_1 using an auxiliary potential p such that

$$\begin{aligned} u_0 &\equiv p_1 - p_0, \\ u_1 &\equiv p_3 - p_2, \\ v_0 &\equiv p_0 - p_2, \\ v_1 &\equiv p_1 - p_3. \end{aligned}$$

By construction the face-centred velocities in each fine cell will be conservative provided

$$\begin{aligned} u_0 - v_0 &= d_0, \\ -u_0 - v_1 &= d_1, \\ v_0 + u_1 &= d_2, \\ v_1 - u_1 &= d_3, \end{aligned}$$

with d_0, d_1, d_2 and d_3 the sums of the incoming face-centred velocities on the boundaries of the corresponding fine cells (dashed or solid arrows). These conditions lead to the following discrete Poisson equation for field p

$$\begin{pmatrix} -2 & 1 & 1 & 0 \\ 1 & -2 & 0 & 1 \\ 1 & 0 & -2 & 1 \\ 0 & 1 & 1 & -2 \end{pmatrix} \boldsymbol{p} = \boldsymbol{d},$$

which has the solutions

$$\boldsymbol{p} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 3/4 & 1/4 & 1/2 \\ 0 & 1/4 & 3/4 & 1/2 \\ 0 & 1/2 & 1/2 & 1 \end{pmatrix} \boldsymbol{d} + \text{arbitrary constant},$$

provided the solvability condition $\sum d_i = 0$ is verified (i.e. the face-centred velocity field on the boundary of C^{i+1} must be divergence-free). This approach is easily generalised to three dimensions.

4 Balanced-force surface tension calculation

The accurate estimation of the surface tension term $(\sigma\ \kappa\ \delta_s\ \boldsymbol{n})_{n+\frac{1}{2}}$ in the discretised momentum equation (3) has proven one of the most difficult aspect of the application of VOF methods to surface-tension-driven flows.

In other methods such as front-tracking (e.g. the "pressure-gradient correction method" of [15]) or levelset coupled with the "ghost fluid method" (GFM) [20], the continuous description of the interface location can be used to derive accurate finite-volume or finite-difference estimates of the surface-tension and pressure-gradient terms taking into account the discontinuities at the interface. Unfortunately front-tracking methods cannot deal simply with topology changes and levelset methods have trouble conserving mass. Coupling VOF, levelset and GFM can be used to overcome the individual drawbacks of each method [23, 61, 43], however this introduces two different representations of the same interface with the associated complexity, efficiency and consistency issues.

In the context of VOF methods, the original Continuum-Surface-Force (CSF) approach of Brackbill (1992) proposes the following approximations

$$\begin{aligned} \sigma\,\kappa\,\delta_s\,\boldsymbol{n} &\approx \sigma\,\kappa\,\boldsymbol{\nabla}\tilde{c}, \\ \kappa &\approx \boldsymbol{\nabla}\cdot\tilde{\boldsymbol{n}} \quad \text{with} \quad \tilde{\boldsymbol{n}} \equiv \frac{\boldsymbol{\nabla}\tilde{c}}{|\boldsymbol{\nabla}\tilde{c}\,|}, \end{aligned}$$

where $\tilde{c}$ is a spatially-filtered volume fraction field. This approach is known to suffer from problematic *parasitic* currents when applied to the case of a stationary droplet in theoretical equilibrium [18, 15, 22]. More recently Renardy & Renardy [21] and Francois et al. [23] noted that in this case, since the discretised momentum equation reduces to

$$-\boldsymbol{\nabla} p_{n+\frac{1}{2}} + \sigma\,\kappa\,(\delta_s\,\boldsymbol{n})_{n+\frac{1}{2}} = \boldsymbol{0},$$

or equivalently using the CSF approximation

$$-\boldsymbol{\nabla} p_{n+\frac{1}{2}} + \sigma\,\kappa\,(\boldsymbol{\nabla} c)_{n+\frac{1}{2}} = \mathbf{0}, \tag{16}$$

it is possible to recover exact discrete equilibrium between surface tension and pressure gradient provided:

1. the discrete approximations of both gradient operators in equation (16) are compatible,
2. the estimated curvature κ is constant,

then

$$p_{n+1/2} = \sigma\,\kappa\,c_{n+1/2} + \text{arbitrary constant},$$

is the exact discrete equilibrium solution. An earlier implementation of a scheme with compatible pressure and interface gradient operators was also described in [6].

Condition 1 can easily be violated in naive implementations of the CSF scheme [62, 23]. For example it may seem simpler to compute the surface tension force at the discrete locations where the volume fraction is defined. When using a classical staggered discretisation the surface-tension forces then need to be estimated on the faces of the control volume which is usually done by averaging the cell-centred estimates. The resulting discrete gradient operator applied to c in equation (16) is thus a spatially-averaged variant of the discrete gradient operator applied to p and condition 1 is violated.

In the present collocated scheme, the cell-centred pressure gradient is computed by averaging the face-centred pressure gradients as indicated in equation (9). The corresponding surface-tension steps verifying condition 1 are:

a) apply the surface-tension force to the auxiliary face-centred velocity field $\boldsymbol{u}^f_\star$ as

$$\boldsymbol{u}^f_\star \longleftarrow \boldsymbol{u}^f_\star + \frac{\Delta t\,\sigma\,\kappa^f_{n+\frac{1}{2}}}{\rho(c^f_{n+\frac{1}{2}})}\boldsymbol{\nabla}^f c_{n+\frac{1}{2}}, \tag{17}$$

b) apply the corresponding cell-centred surface-tension force to $\boldsymbol{u}^c_\star$ as

$$\boldsymbol{u}^c_\star \longleftarrow \boldsymbol{u}^c_\star + \left|\frac{\Delta t\,\sigma\,\kappa^f_{n+\frac{1}{2}}}{\rho^f_{n+\frac{1}{2}}}\boldsymbol{\nabla}^f c_{n+\frac{1}{2}}\right|^c.$$

These steps are applied immediately before the projection steps. In practice the implementation of steps a) and b) is identical to the implementation of the pressure corrections (8) and (9) respectively. Parameters are used to apply the routines either to $c_{n+1/2}$ or $p_{n+1/2}$ with

weights $\sigma\,\kappa_{n+1/2}$ or -1 respectively. This avoids duplicating similar code and guarantees that condition 1 is verified exactly.

5 Generalised height-function curvature calculation

Condition 2 depends on the accuracy of the curvature estimation. Estimating curvature has traditionally been the Achilles' heel of Volume-Of-Fluid schemes and has prompted many to recommend levelset [5], coupled VOF/levelset [63, 64] or front-tracking schemes [15, 16] as alternatives.

Besides the original curvature estimation of Brackbill, several methods have been proposed for estimating curvature from volume fraction fields. The height-function (HF) curvature calculation initially proposed by Torrey et al [65] has recently been shown to give practical estimates of the curvature which are comparable in accuracy to estimates obtained using the differentiation of exact levelset functions [25]. The HF method is also substantially simpler to implement than other high-order methods such as parabolic fitting (PROST [21]) or spline interpolation [24]. An important shortcoming of the standard HF method which is rarely discussed, however, is that it becomes inconsistent when the radius of curvature of the interface becomes comparable to the mesh size. This case of an under-resolved interface often occurs in practice, particularly when dealing with topology changes.

In the following, I will present a height-function calculation generalised to quad/octree discretisations and which uses a hierarchy of consistent approximations to deal with the case of under-resolved interfaces.

5.1 Optimised height-function formulation on quad/octree discretisations

The standard height-function curvature calculation on 2D Cartesian grids [25] can be summarised as:

1. Consider a 3×7 (resp. 7×3) stencil centred on the cell where the curvature is to be evaluated; an estimate of the interface orientation is used to choose the stencil best aligned with the normal direction to the interface.
2. Build a discrete approximation of the interface height $y = h(x)$ (resp. $x = h(y)$) by summing the volume fractions in each column (resp. line).
3. Use finite-difference approximations of the derivatives of the discretised height-function to compute the curvature

$$\kappa = \left.\frac{h''}{(1+h'^2)^{3/2}}\right|_{x=0}$$

While this scheme could easily be extended to a quad/octree discretisation using the approach described in section 3.1 (Algorithm 1), it would imply systematically reconstructing a large stencil around each cell where the curvature needs to be evaluated. In many cases the stencil does not need to be seven cells high (Figure 4.a) while some cases may require stencils higher than seven cells in order to obtain a consistent estimate of the interface height (Figure 4.b).

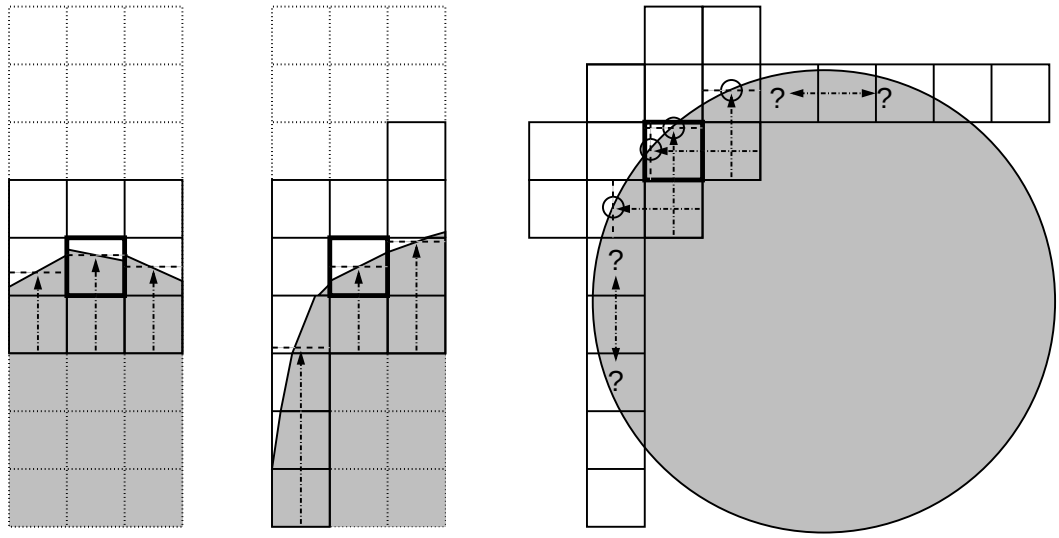


Fig. 4. (a) A 3×3 symmetric stencil is sufficient to obtain a consistent estimate of the interface height for weakly-curved interfaces aligned with the grid. (b) Symmetric stencils higher than seven cells may be required to obtain consistent interface heights (here the 9×3 symmetric stencil indicated by the dotted lines is necessary). More compact asymmetric stencils can be built independently for each column (solid lines). (c) In this case only two of the three interface heights are consistent for either the vertical or the horizontal stencils.

An alternative to using wide symmetric stencils is to build local asymmetric stencils adapted to the geometry of the interface (in a manner similar to that proposed in [26]). For example in Figure 4.b local stencils of varying heights can be constructed independently for each of the three columns where the interface height is to be evaluated: a 1×6 stencil for the left column, a 1×3 stencil for the central column and a 1×4 stencil for the right column. The elementary operation is thus the evaluation of the interface height in a given column, for which I propose the following algorithm:

Algorithm 4 *Interface height ($\mathcal{C}$, top)*

1. *set* $H \longleftarrow c(\mathcal{C})$
2. *set* $\mathcal{N} \longleftarrow \mathcal{C}$, $c_t \longleftarrow c(\mathcal{C})$
3. ***if*** $c_t < 1$ ***then*** *set* $\mathcal{I} \longleftarrow$ true ***else*** *set* $\mathcal{I} \longleftarrow$ false
4. ***while*** $\mathcal{I} =$ false ***or*** $\mathcal{N}$ *contains the interface*
   a) *replace* $\mathcal{N}$ *with its top neighbour, update* $c_t \longleftarrow c(\mathcal{N})$
   b) *update* $H \longleftarrow H + c_t$
   c) ***if*** $\mathcal{N}$ *contains the interface* ***then*** *set* $\mathcal{I} \longleftarrow$ true
5. ***if*** $c_t \neq 0$ ***then return*** inconsistent height
6. *set* $\mathcal{N} \longleftarrow \mathcal{C}$, $c_b \longleftarrow c(\mathcal{C})$
7. ***if*** $c_b > 0$ ***then*** *set* $\mathcal{I} \longleftarrow$ true ***else*** *set* $\mathcal{I} \longleftarrow$ false

8. ***while*** $\mathcal{I} = \text{false}$ ***or*** $\mathcal{N}$ *contains the interface*
   a) *replace* $\mathcal{N}$ *with its bottom neighbour, update* $c_b \longleftarrow c(\mathcal{N})$
   b) *update* $H \longleftarrow H + c_b$
   c) ***if*** $\mathcal{N}$ *contains the interface* ***then*** *set* $\mathcal{I} \longleftarrow \text{true}$
9. ***if*** $c_b \neq 1$ ***then return*** inconsistent height
10. ***return*** H *and* $\mathcal{N}$

$\mathcal{C}$ is a cell belonging to the column and can be seen as an *initial guess* of the position of the interface. The *top* and *bottom* directions used in steps 4.a and 8.a respectively are defined relative to the interface orientation so that for a consistent interface the cells at the top of the column are empty ($c = 0$) while the cells at the bottom are full ($c = 1$). In a first stage (steps 2 to 5) volume fractions in the top direction are summed until the interface has been found ($\mathcal{I} = \text{true}$) and a full or empty cell has been reached. If the cell which has been reached is full a consistent stencil cannot be formed and an error is returned (condition 5). In a second stage (steps 6 to 9) the same procedure is repeated in the bottom direction, looking this time for a full cell. Given a good enough initial guess $\mathcal{C}$ this algorithm will be equivalent to using optimal local asymmetric stencils. Also, no a priori assumption is made on the maximum height of the stencil required to obtain a consistent interface height.

Algorithm 4 is easily generalised to a quad/octree discretisation by considering *virtual* rather than *actual* cells in a manner similar to Algorithm 1. The virtual cells are constructed on demand when evaluating $c(\mathcal{C})$ and $c(\mathcal{N})$.

For a given cell $\mathcal{C}$ and direction *top*, the full height-function curvature estimation algorithm can then be summarised as:

Algorithm 5 *Height-function curvature (*$\mathcal{C}$*, top)*

1. *set* h_0 *and* $\mathcal{N}_0$ $\longleftarrow$ *Interface height (*$\mathcal{C}$*, top)*
2. ***foreach*** *of the 2 (8 in* 3D*) columns neighbouring* $\mathcal{C}$
   a) *define* $\mathcal{N}$ *as the closest neighbouring (virtual) cell to* $\mathcal{C}$ *in the plane perpendicular to the top direction*
   b) *set* h_i *and* $\mathcal{N}_i$ $\longleftarrow$ *Interface height (*$\mathcal{N}$*, top)*
   c) *set a common origin* $h_i \longleftarrow h_i + h(\mathcal{N}_i) - h(\mathcal{N}_0)$
3. ***if*** *all heights are consistent*
   a) ***return*** *the curvature estimated using finite-difference approximations of the derivatives of the discretised height function* h_i
4. ***else***
   a) ***return*** *all the interface positions deduced from the consistent heights*

Step 2.c is necessary since the stencils for each column are formed independently and do not necessarily share a common origin (cf. arrows in Figure 4.b). The function $h(\mathcal{C})$ returns the absolute height of the centre of cell $\mathcal{C}$.

5.2 Generalised height-function formulation for under-resolved interfaces

An important drawback of the standard height-function method is that even moderately

curved interfaces can lead to configurations where consistent interface heights cannot be formed. An example of such a case is given in Figure 4.c where both the standard horizontal and vertical HF curvature estimations fail. The radius of curvature of the interface is $4\,\Delta$ which is not a very coarse approximation for many practical applications.

While neither the horizontal nor the vertical stencils on their own can be used to construct a twice-differentiable discrete approximation of the interface height it is clear that a combination of all the stencils can allow such an approximation. In the case of Figure 4.c for example, four average interface positions can be obtained from the consistent interface height estimates (the circles in Figure 4.c). Fitting a curve (e.g. a parabola in 2D, paraboloid in 3D) through these points and differentiating the resulting analytical function will give an estimate of the curvature.

More generally given a cell $\mathcal{C}$ and n estimated interface positions $\{\boldsymbol{x}_1, ..., \boldsymbol{x}_n\}$, the following algorithm can be used to estimate the curvature

Algorithm 6 *Parabola-fitted curvature ($\mathcal{C}$, $\{\boldsymbol{x}_1, ..., \boldsymbol{x}_n\}$)*

1. ***if*** *the number of* independent *interface positions is smaller than three (resp. six in* 3D*)*
   a) *a meaningful least-squares fit cannot be achieved,* ***return*** *an error*
2. *retrieve normal $\boldsymbol{m}$ to the interface in cell $\mathcal{C}$ (pre-computed using Algorithm 3)*
3. *compute the coordinates $\boldsymbol{o}$ of the barycentre of the reconstructed interface fragment contained in $\mathcal{C}$*
4. *define an orthonormal coordinate system $\mathcal{R} \equiv \{\boldsymbol{o}, (\boldsymbol{i}', \boldsymbol{m})\}$ (resp. $\{\boldsymbol{o}, (\boldsymbol{i}', \boldsymbol{j}', \boldsymbol{m})\}$ in* 3D*)*
5. *compute the transformed coordinates $\{\boldsymbol{x}'_1, ..., \boldsymbol{x}'_n\}$ of the interface positions in $\mathcal{R}$*
6. *fit a parabola (resp. paraboloid in* 3D*) by minimising*

$$\mathcal{F}(a_i) \equiv \sum_{1 \leqslant j \leqslant n} \left[z'_j - f(a_i, \boldsymbol{x}'_j) \right]^2,$$

with

$$f(a_i, \boldsymbol{x}) \equiv \begin{cases} a_0\, x^2 + a_1\, x + a_2 & \text{in 2D} \\ a_0\, x^2 + a_1\, y^2 + a_2\, x\, y + a_3\, x + a_4\, y + a_5 & \text{in 3D} \end{cases}$$

7. ***return*** *the mean curvature at the origin $\boldsymbol{o}$ of $\mathcal{R}$*

$$\kappa \equiv \begin{cases} \dfrac{2\, a_0}{(1 + a_1^2)^{3/2}} & \text{in 2D} \\ 2\, \dfrac{a_0\,(1 + a_4^2) + a_1\,(1 + a_3^2) - a_2\, a_3\, a_4}{(1 + a_3^2 + a_4^2)^{3/2}} & \text{in 3D} \end{cases}$$

The condition in step 1 uses the number of *independent* positions rather than the total number of positions n. Two interface positions $\boldsymbol{a}$ and $\boldsymbol{b}$ are considered *independent* provided $|\boldsymbol{a} - \boldsymbol{b}| \geqslant \Delta$. This condition ensures that the resulting minimisation problem is well-conditioned. The minimisation only requires the solution of a 3×3 (resp. 6×6 in 3D) linear system.

In practice this scheme is sufficient to compute the curvature in most cases where the

standard HF technique fails, however the number of independent positions – given by consistent interface height estimates – may become too small for complicated interfaces at very coarse resolutions. In this case a new set of interface positions is constructed by computing the coordinates of the barycentres of the reconstructed interface fragments in each cell of a 3×3 stencil (resp. $3 \times 3 \times 3$ in 3D). If this approach in turn fails to provide enough independent positions, the cell considered most probably contains an isolated/degenerate interface fragment and the curvature is simply set to zero.

The overall algorithm for computing the interface mean curvature in a cell $\mathcal{C}$ containing the interface can then be summarised as

Algorithm 7 *curvature ($\mathcal{C}$)*

1. *retrieve normal $\boldsymbol{m}$ to the interface in cell $\mathcal{C}$ (pre-computed using Algorithm 3)*
2. *compute a set $\mathcal{S}$ of two (resp. three in 3D) spatial directions in decreasing order of alignment with normal $\boldsymbol{m}$*
3. ***foreach** top direction in $\mathcal{S}$*
   a) *compute $\kappa =$ Height-function curvature ($\mathcal{C}$, top)*
   b) ***if** κ is consistent **return** κ*
   c) ***else** add the interface positions deduced from the consistent heights to set $\mathcal{I}$*
4. ***if** the number of independent positions in $\mathcal{I}$ is smaller than three (resp. six in 3D)*
   a) *replace $\mathcal{I}$ with the set of interface positions built from the barycentres of the reconstructed interface fragments in a 3×3 (resp. $3 \times 3 \times 3$ in 3D) stencil centred on $\mathcal{C}$*
   b) ***if** the number of independent positions in $\mathcal{I}$ is still too small **return** zero*
5. ***return** Parabola-fitted curvature ($\mathcal{C}$, $\mathcal{I}$)*

The algorithm is hierarchical. In the majority of cases it will return within the first iteration of loop 3 (i.e. the standard HF method). In some cases the direction "best-aligned" with normal $\boldsymbol{m}$ will lead to inconsistent interface heights while the next best direction will not and the algorithm will return after more than one iteration of loop 3. Most other cases will be dealt with directly by step 5, rarer cases will require step 4.a and still rarer cases step 4.b. While estimating curvature using parabola fitting is substantially more expensive than the standard HF method, in practice the overall cost is thus largely dominated by the cost of the standard HF method.

The balanced-force surface tension correction (17) requires face-centred interface curvature estimates. The face-centred curvatures are computed either by averaging the cell-centred curvatures of the neighbouring cells when they both contain the interface, or by taking the value of the cell-centred curvature in either cell containing the interface.

6 Results

6.1 Convergence of the generalised height-function curvature

The standard height-function method has been shown to provide asymptotically second-

order accurate estimates of the curvature when given an exact volume fraction field [25]. In order to assess the consistency of the generalisation proposed in the previous section, we repeat the test case of a circular interface but over a larger range of spatial resolutions. The radius of the circle is varied from less than $2\,\Delta$ to more than $16\,\Delta$ and the corresponding volume fractions are initialised exactly. The position of the centre of the circle is varied randomly in the interval ($[0,\,\Delta]$, $[0,\,\Delta]$), so that a representative sample of the various interface configurations is considered for each radius. The following relative error norms are defined

$$
\begin{aligned}
L_2(\kappa_{\text{exact}}) &\equiv \frac{1}{\kappa_{\text{exact}}}\sqrt{\frac{\sum_r \sum_i \left(\kappa_{r,i}-\kappa_{\text{exact}}\right)^2}{\sum_r \sum_i}}, \\
L_\infty(\kappa_{\text{exact}}) &\equiv \frac{1}{\kappa_{\text{exact}}}\max_{r,i}\left(|\kappa_{r,i}-\kappa_{\text{exact}}|\right),
\end{aligned}
$$

where r is the index of the random circle position and i the index of a cell containing an interface fragment (i.e. for which $0 < c < 1$). In addition two independent L_∞ norms are computed: one for the cells where the curvature is estimated using the standard HF method, the other for the cells using the parabola-fitting of Algorithm 6 (note that steps 4.a,b of Algorithm 7 are never necessary for this test case). The results obtained for 100 random circle positions (for each radius) are summarised in Figure 5.

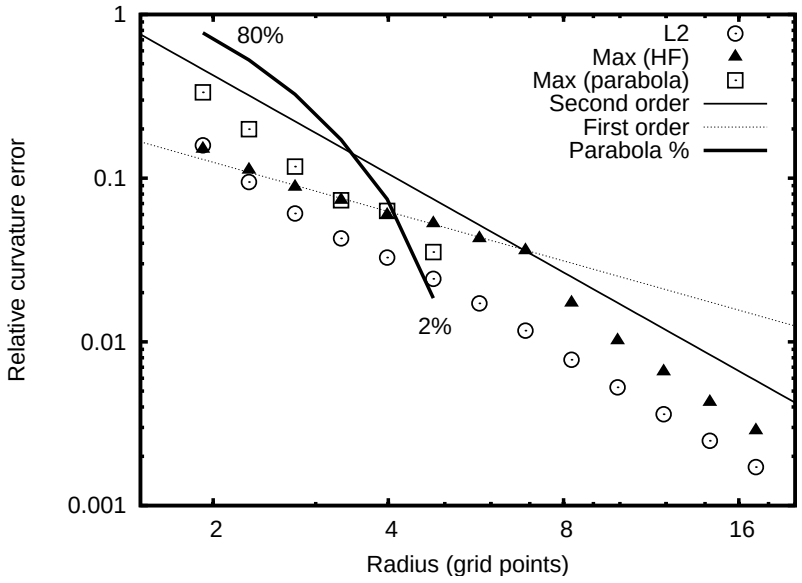


Fig. 5. Convergence of the relative curvature errors as a function of radius for a circular interface. Two sets of L_∞ (max) norms are given depending on whether the curvature was calculated using the standard height-function method (HF) or the parabola-fitting technique (parabola).

The thick curve in the figure indicates the proportion of cases where parabola fitting was necessary (from $\approx 80\%$ for a radius of $2\,\Delta$ down to $\approx 2\%$ for a radius of $5\,\Delta$). Above a radius of approximately $5\,\Delta$, the standard HF method is always consistent and use of the parabola-fitting technique is not necessary. In this regime an asymptotically second-order convergence in both the L_2 and L_∞ norms is observed as in previous studies.

At low resolutions the largest errors are obtained for the cases in which the parabola-fitting technique is used. This is not surprising since at low resolution the HF method is consistent only for nearly horizontal or vertical interfaces for which good curvature estimates can be found. The parabola-fitting errors show second-order convergence.

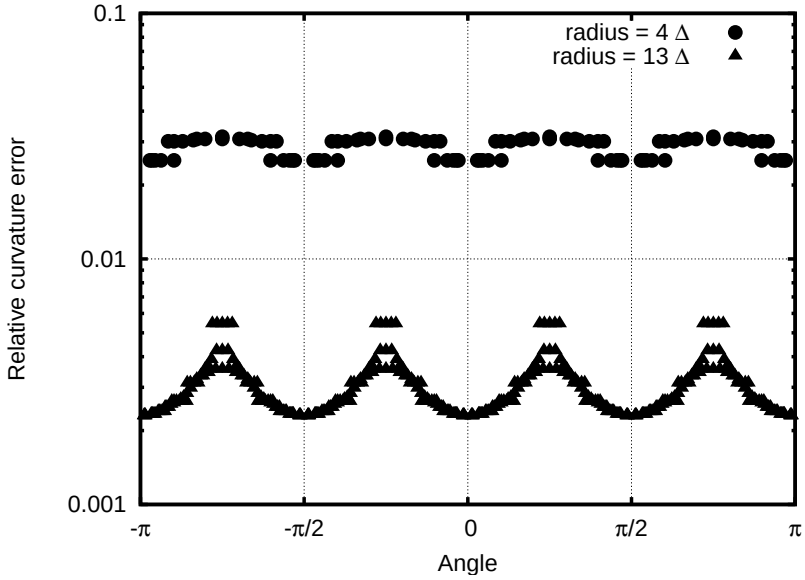


Fig. 6. Distribution of relative curvature errors along the circular interface at two resolutions.

At intermediate resolutions (radius $\approx 4\ \Delta$) the maximum errors for both methods are comparable and correspond to cases where the interface is inclined at 45° (Figure 6). Interestingly, in this regime the L_∞ norm of the standard HF method only shows first-order convergence.

Over the whole range of resolutions these results are consistent with second-order convergence of the generalised height-function technique in both the L_2 and L_∞ norms.

6.2 Circular droplet in equilibrium

6.2.1 Stationary droplet

A circular interface with surface tension should remain at rest with the pressure jump at the interface exactly balancing the surface tension force (Laplace's law). In practice, depending on the method used for discretising the surface tension force and the pressure gradient, an exact numerical balance is difficult to obtain and often leads to so-called "spurious" or "parasitic" velocities fed by this imbalance [18, 15, 22]. This severely limits the range of physical parameters which can be accurately simulated using CSF-based techniques. The balanced-force surface tension formulation presented above is an important step toward an exact numerical solution of Laplace's problem, however an interesting recent study by Francois et al. [23] concluded that better methods for curvature estimation are required before this can be achieved. In contrast, in the following I will show that the combination of height-function curvature estimation and balanced-force surface tension described previously is sufficient to recover exact (to within round-off errors) numerical balance for Laplace's problem *provided* the shape of the interface is given enough time to relax to its *numerical* equilibrium shape.

I will consider a circular interface centred on the top-left corner of the unit square. Symmetry conditions are applied on the top and left boundaries so that only a quarter of the droplet is simulated on a 32×32 grid. The diameter of the droplet is $D \equiv 0.8$ and the corresponding volume fractions are initialised exactly. The divergence criterion γ_p is set to 10^{-6}. The only velocity scale for the inviscid problem is

$$U_\sigma \equiv \sqrt{\frac{\sigma}{\rho\, D}},$$

with ρ the constant density. This can be interpreted as the scale of the velocities associated with a capillary wave of wavelength comparable to D. The corresponding timescale is

$$T_\sigma \equiv \sqrt{\frac{\rho\, D^3}{\sigma}},$$

which is proportional to the period of the capillary wave. For a viscous fluid another timescale can be defined as

$$T_\nu \equiv D^2/\nu,$$

with ν the kinematic viscosity. The time it takes for momentum to diffuse across the droplet is proportional to T_ν. Finally the ratio of these two timescales is

$$\frac{T_\nu}{T_\sigma} = \sqrt{\frac{\sigma D}{\rho \nu^2}} = \sqrt{\mathrm{La}},$$

with La the Laplace number.

For our time-explicit discretisation of the surface tension term, numerical stability requires that the timestep be smaller than the period of the shortest capillary wave i.e.

$$\Delta t \leqslant \sqrt{\frac{\rho \Delta^3}{\pi \sigma}}. \tag{18}$$

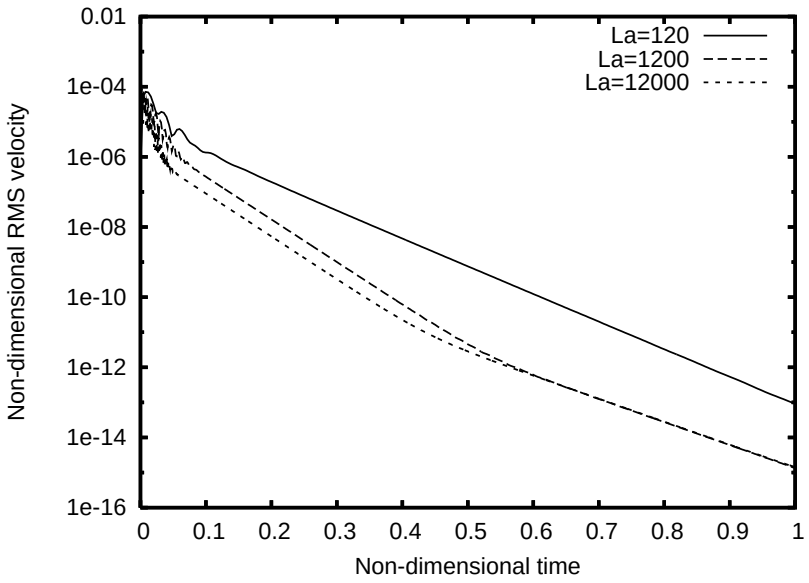


Fig. 7. Evolution of the RMS velocity around a circular droplet in theoretical equilibrium for different Laplace numbers. Time and velocity are made non-dimensional using T_ν and U_σ as reference scales respectively.

Figure 7 illustrates the evolution of the root-mean-square (RMS) velocity with time for the range of Laplace numbers indicated in the legend. Time and velocity are made non-dimensional using T_ν and U_σ as reference scales respectively. The RMS velocity shows a roughly exponential decrease with time at all Laplace numbers. At long enough times ($t \approx T_\nu$) exact balance (to within round-off errors) is recovered between surface tension and pressure and the velocity tends toward zero. This result holds independently of the spatial resolution. Figure 8 gives a different representation of the initial oscillations apparent in Figure 7, with time made non-dimensional using T_σ as reference scale. The case of an inviscid fluid ($\mathrm{La} = \infty$) has also been added.

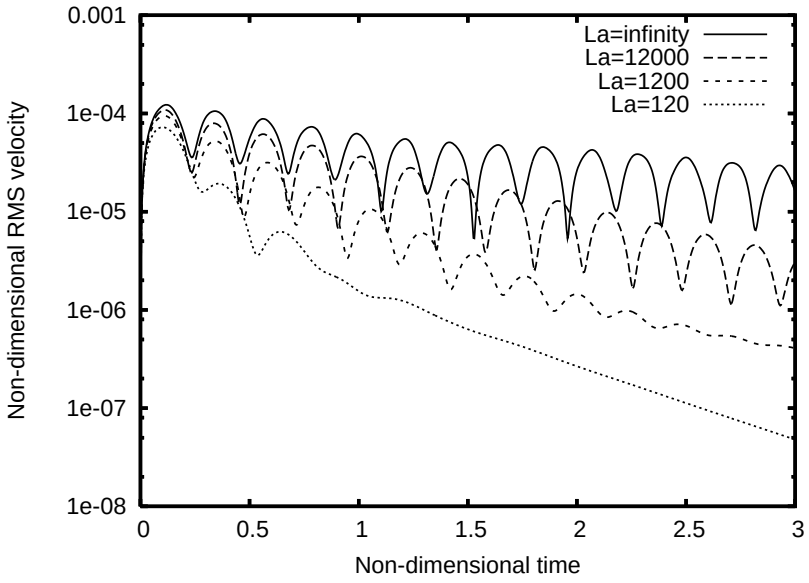
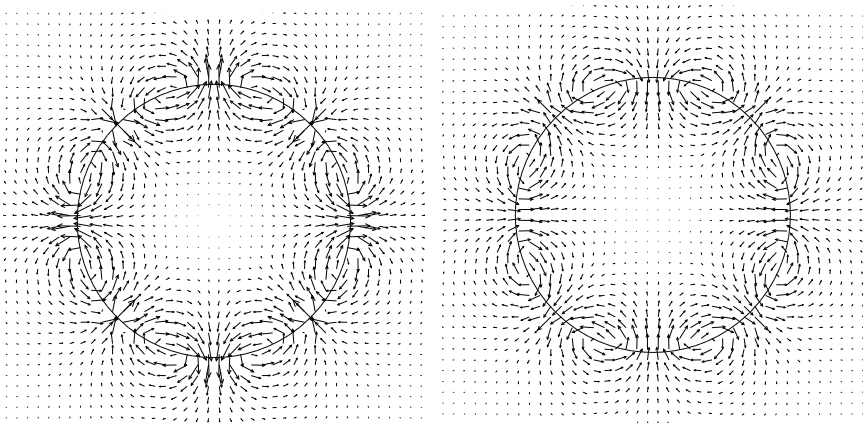


Fig. 8. Initial evolution of the RMS velocity around a circular droplet in theoretical equilibrium. Time and velocity are made non-dimensional using T_σ and U_σ as reference scales respectively.



(a) (b)

Fig. 9. Velocity fields at times (a) $t \approx T_\lambda/4$ and (b) $t \approx 3\,T_\lambda/4$ for $\mathrm{La} = \infty$.

The results for an inviscid fluid are comparable to those presented in Francois et al [23], who then concluded that this non-vanishing velocity was a measure of the *spurious currents* induced by an insufficiently accurate estimate of the curvature. I believe this is an incorrect interpretation of these results. From Figure 8 and looking at the details of the velocity field evolution (Figure 9 above and Figure 17.b,d in [23]) it is clear that the oscillations of the velocity field correspond to the oscillations of capillary waves of period $\approx 0.4\,T_\sigma$ (independent of the Laplace number). An estimate of the corresponding wavelength λ can be obtained using the relation [14]

$$T_\lambda \equiv \sqrt{\frac{\rho\,\lambda^3}{\pi\,\sigma}} \approx 0.4\,T_\sigma,$$

which gives

$$\lambda \approx 0.8\,D \approx \pi\,D/4,$$

which is consistent with the fourfold symmetry observed in the velocity field and in the initial curvature errors (case radius $= 13\,\Delta$ in Figure 6). These capillary waves exist because of the difference between the *exact* equilibrium interface shape (imposed as initial condition) and the *numerical* equilibrium shape (eventually reached at $t \approx T_\nu$). I suggest the correct

interpretation is in fact that the velocity fluctuations observed are the result of a physically-consistent numerical solution of the evolution of a perturbed initial interface shape. The resulting capillary waves are then exponentially damped by viscosity – again in a physically consistent manner – on timescales of order T_ν (with a damping coefficient proportional to ν as illustrated by the roughly constant slopes in Figure 7). This is in contrast with what is usually understood by *spurious* or *parasitic* currents which are continuously fed energy by an unphysical unbalance, do not vanish with time and cannot generally be interpreted as consistent physical entities (such as capillary waves).

When the numerical equilibrium interface shape is reached, we expect the estimated curvature to be exactly constant so that condition 2 of section 4 is verified. This is indeed the case as illustrated in Figure 10 where the saturation of the standard deviation at $O(10^{-11})$ is due to round-off errors.

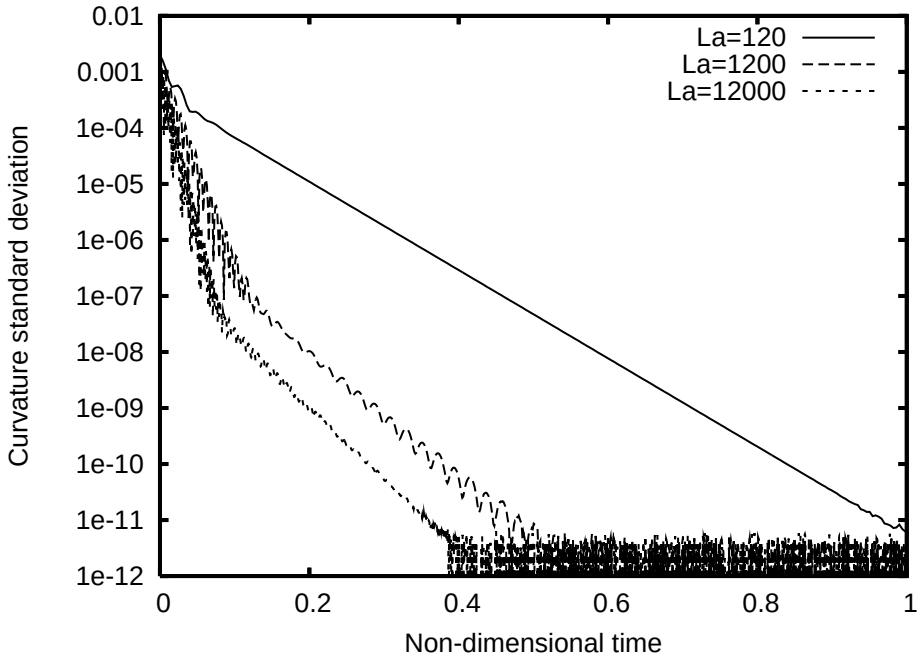


Fig. 10. Evolution of the standard deviation along the interface of the numerical curvature as a function of time for the Laplace numbers indicated in the legend. Time is made non-dimensional using T_ν as reference scale.

Finally the accuracy of the numerical equilibrium solution can be assessed by comparing the final shape of the interface with the exact equilibrium shape (the initial circular shape). Two measures are used: the relative curvature error $|\kappa_{\text{equilibrium}} - \kappa_{\text{exact}}|/\kappa_{\text{exact}}$ and the shape error defined as

$$L_2(\text{shape}) \equiv \sqrt{\frac{\sum_i (c_i - c_i^{\text{exact}})^2}{\sum_i}},$$

$$L_\infty(\text{shape}) \equiv \max_i (|c_i - c_i^{\text{exact}}|),$$

where c^{exact} is the exact volume fraction field. Figure 11 summarises the convergence of both measures when the spatial resolution is varied. Close to second-order convergence is obtained for both measures and the absolute values of the errors are small even at coarse resolutions.

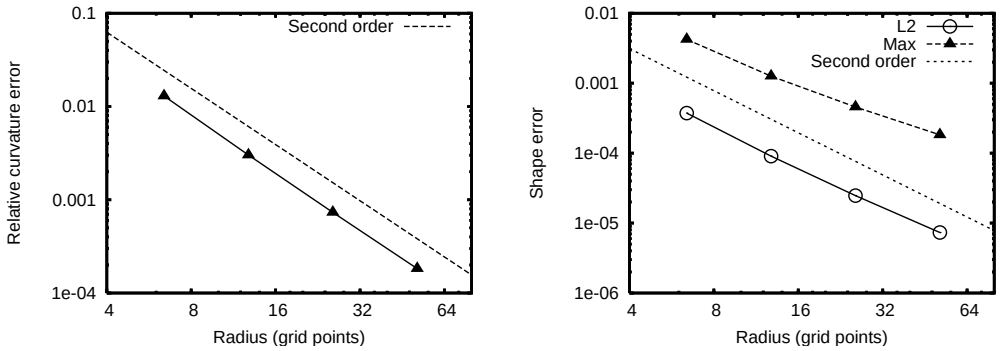


Fig. 11. Convergence with spatial resolution of two error measures of the equilibrium interface shape. (a) Relative error on the equilibrium curvature: $|\kappa_{\text{equilibrium}} - \kappa_{\text{exact}}|/\kappa_{\text{exact}}$. (b) L_2 and L_∞ norms of the shape error.

To conclude, the results presented in this section demonstrate that the combination of a balanced-force CSF implementation and a height-function curvature estimation is sufficient to obtain an exact equilibrium solution (for velocity) in the case of a stationary droplet. The numerical equilibrium shape obtained is very close to the theoretical equilibrium shape and shows second-order convergence with spatial refinement. The timescale necessary to reach the numerical equilibrium solution is comparable to the viscous dissipation timescale T_ν as expected from physical considerations. In practice this means that test cases designed to evaluate the accuracy of a given surface tension implementation (for the stationary droplet problem) need to make sure that simulations are run for timescales comparable to T_ν (this was not the case in the study of Francois et al. [23]). For shorter timescales the results obtained will only reflect the accuracy of the initial curvature estimation. Estimating this initial error is better done using a test case similar to that of Section 6.1.

6.2.2 Translating droplet

While the previous test case demonstrates convergence toward the exact velocity solution in the case of a stationary droplet with surface tension, it is only a weak test of the combined accuracy of the interface advection scheme and the surface tension discretisation. In most practical applications droplets do not stand still but are advected around by the mean flow. A simple but more representative variant can be built from the previous case by adding a constant, uniform background velocity field. The exact solution in the moving frame of reference is unchanged but the numerical solution requires advection of the interface across

the fixed computational grid. Errors in interface advection will induce fluctuating errors in curvature estimates and hence – through surface tension – in the velocity field. Numerous studies have considered the case of a stationary droplet with surface tension [18, 15, 21, 16, 22, 23] and/or the advection of a circular interface without coupling to the momentum equation [2, 3, 53, 64, 33, 41] but I am not aware of any study using a combination of both, hence the present test case.

The following results are obtained when considering a circular interface of diameter $D = 0.4$ contained in the unit square with a mesh refinement equivalent to a 64×64 regular spatial resolution. Periodic boundary conditions are used in the horizontal direction, symmetry conditions on the top and bottom boundaries. A constant velocity U is initialised in the horizontal direction. Compared to the previous test case, this introduces a new timescale $T_U \equiv D/U$, with $T_\sigma/T_U = \sqrt{\text{We}}$ and We the Weber number.

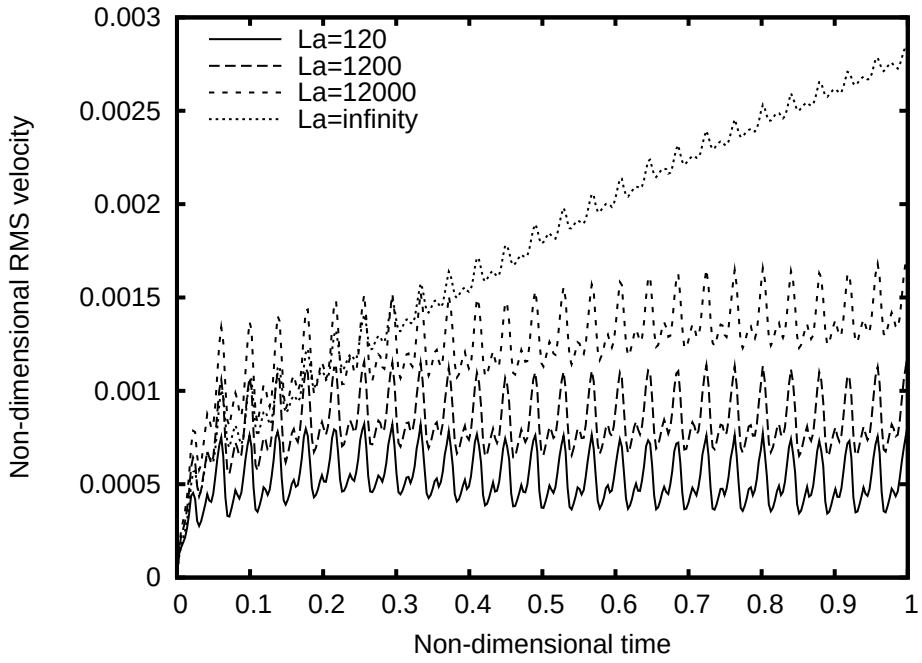


Fig. 12. Evolution with time of the norm of the velocity in the translating frame of reference for different Laplace numbers. Time and velocity are made non-dimensional using T_U and U as reference scales respectively.

Figure 12 illustrates the evolution with time of the non-dimensional root-mean-square velocity – in the translating frame of reference – for different Laplace numbers. The Weber number is 0.4. Time and velocity are made non-dimensional using T_U and U as reference scales respectively. In contrast to the results of the previous section, the velocity does not converge toward the exact solution with time. The frequency associated with the oscillations apparent in the figure scales like U/Δ. This is due to advection errors continually perturbing the interface shape and thus the estimated curvature. The amplitude of this "spurious" velocity field is indicative of the accuracy of the combination of: advection scheme, curvature estimation and surface tension scheme. This simulation was repeated for a range of spatial

resolutions and the convergence results are summarised in Figure 13. Global L_2 and L_∞ spatial-norms are computed by taking the maximum of either norms over time. The maximum errors in velocity are of the order of 5% of U. Less than first-order convergence is observed for the maximum error and close to first-order convergence for the RMS error.

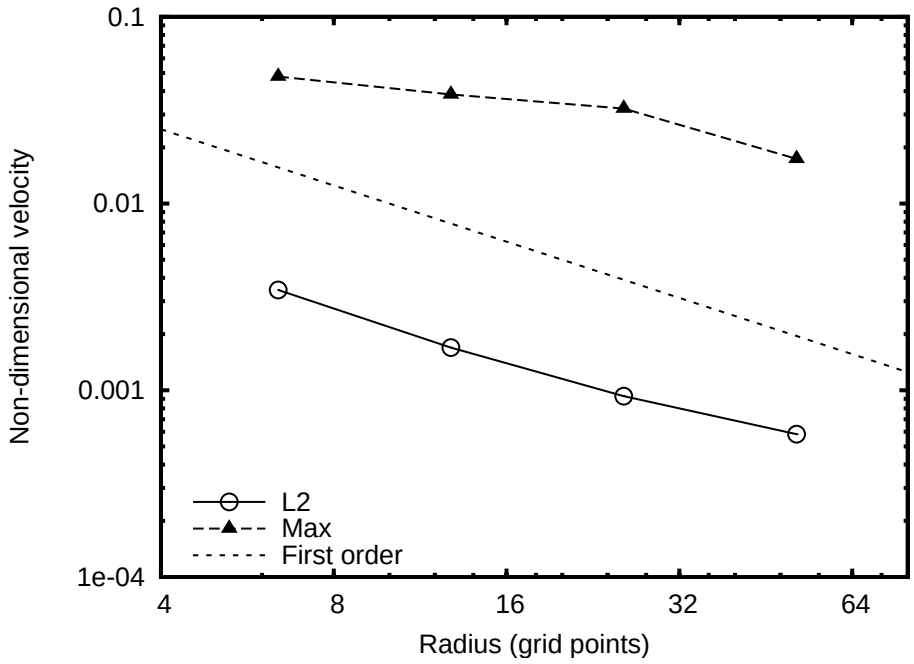


Fig. 13. Convergence with spatial resolution of the non-dimensional velocity norms in the translating frame of reference. $\mathrm{La} = 12,000$.

The shape errors can be computed using the exact translating interface shape as reference. The maximum over time of the curvature and shape error norms are summarised in Figure 14. The roughly first-order convergence rates are consistent with the convergence rate of the error in velocity.

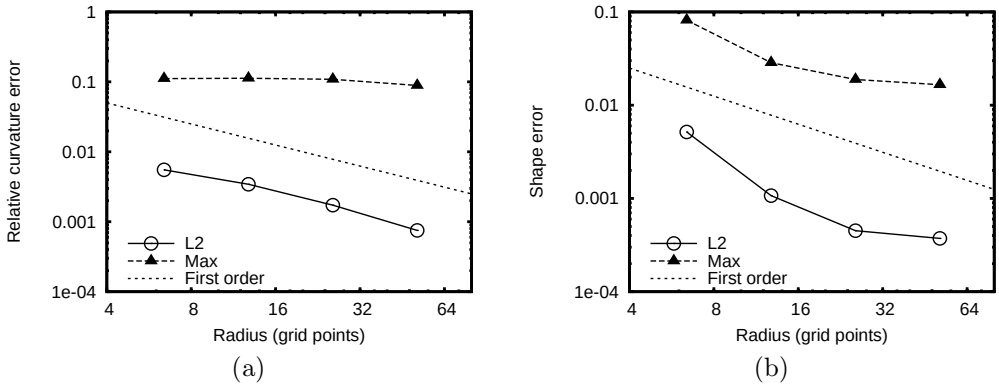


Fig. 14. Convergence with spatial resolution of two error measures of the translating interface shape. (a) Relative error on the equilibrium curvature: $|\kappa_{\mathrm{equilibrium}} - \kappa_{\mathrm{exact}}|/\kappa_{\mathrm{exact}}$. (b) L_2 and L_∞ norms of the shape error.

Finally the dependences of the relative error in velocity on the Weber and Laplace numbers are illustrated in Figure 15.a and 15.b. The *relative* error scales like $\mathrm{We}^{-1/2}$ or equivalently like U^{-1} (σ being kept constant) which means that the *absolute* error in velocity is essentially independent from U. The relative error in velocity is also seen to be weakly dependent on the Laplace number (Figure 15.b)

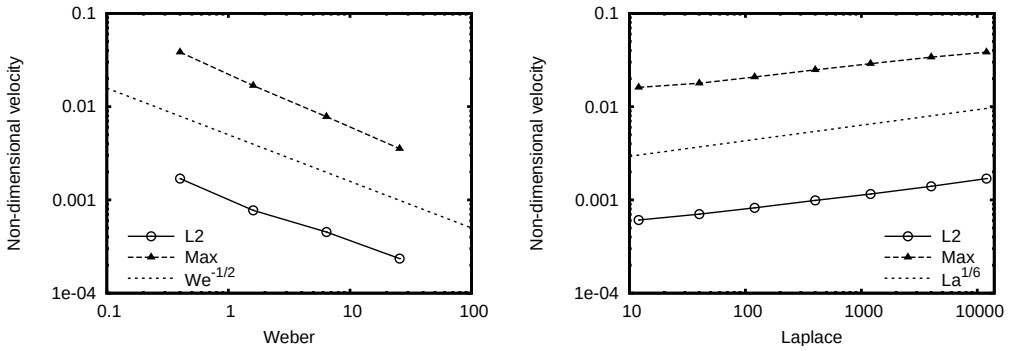


Fig. 15. Dependence of the relative error in velocity on: (a) the Weber number, La = 12, 000, (b) the Laplace number, We = 0.4 . Equivalent resolution 64×64.

The convergence toward the exact solution for a stationary droplet presented in the previous section has been one of the main goals of several recent sophisticated surface tension implementations [21, 16, 23]; however the results obtained in the present section – for a test case more relevant to real applications – underline the need for yet further advances. An avenue of investigation revealed by the translating droplet test case regards the tighter integration (and coupled evaluation) of the advection and curvature/surface tension schemes.

6.3 Capillary wave

Capillary waves are a basic feature of surface-tension-driven flows and their adequate numerical resolution is a prerequisite to more complex applications. The small-amplitude damped oscillations of a capillary wave are a now classical test case of the accuracy of numerical schemes for the time-dependent dynamics of viscous, surface-tension-driven two-phase flows [15]. For example a recent study by Gerlach et al. (2006) used this test case to assess the relative accuracies of a range of VOF and VOF/levelset techniques.

A sinusoidal perturbation is applied to a flat interface between two fluids initially at rest. Under the influence of surface tension the interface oscillates around its equilibrium position.

The amplitude of the oscillation decays due to viscous dissipation. The solution to this *initial value* problem is significantly different from the *normal mode* analysis of Lamb [67] (which predicts exponentially-damped oscillations) because the finite time of diffusion of vorticity generated at the interface into each of the phases introduces non-trivial history terms in the evolution equations. Nonetheless, Prosperetti [68] found an exact analytical solution in the limit of vanishingly small amplitudes which can be used as reference.

Prosperetti's theory is valid for infinite domains and we found that in order to get good agreement at high resolutions it is necessary to move the top and bottom boundaries far enough away from the interface. Using a $[-\lambda/2, \lambda/2][-3\,\lambda/2, 3\,\lambda/2]$ domain – with λ the wavelength of the perturbation – seems to be sufficient. As in Popinet and Zaleski [15] and Gerlach et al. [66] the initial perturbation amplitude is $\lambda/100$, the Laplace number 3000 and the densities and viscosities of both fluids are identical. The timestep is controlled by the explicit surface-tension stability criterion (18). The typical evolution of the relative maximum interface elevation over time is illustrated in Figure 16. The normal-mode oscillation frequency ω_0 is defined by the dispersion relation

$$\omega_0^2 = \frac{\sigma\, k^3}{2\,\rho},$$

with $k = 2\,\pi/\lambda$ the wavenumber.

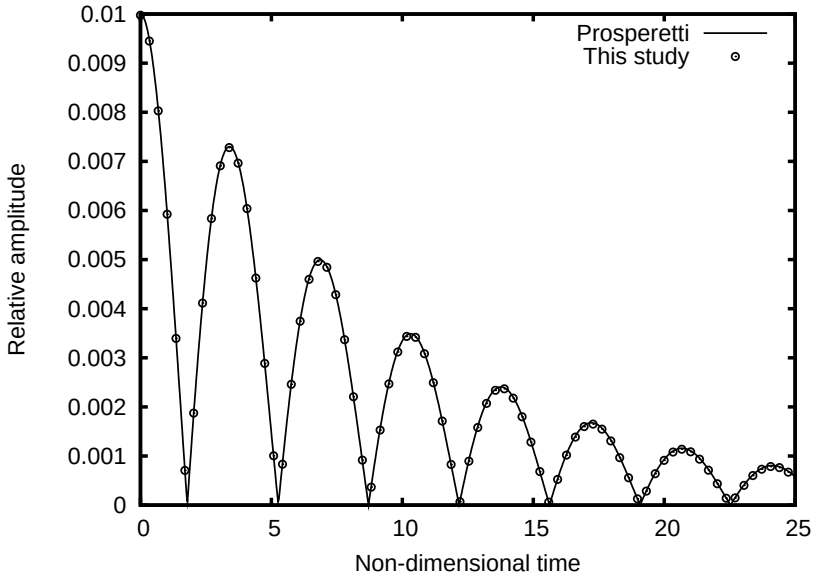


Fig. 16. Time evolution of the relative maximum interface height of a capillary wave. La = 3000. Time is made non-dimensional using $1/\omega_0$ as reference timescale. Equivalent resolution 64×192.

The error between the theoretical solution of Prosperetti and the numerical solution can be

evaluated using the L_2 norm

$$L_2 \equiv \frac{1}{\lambda}\sqrt{\frac{\omega_0}{25}\int_{t=0}^{25/\omega_0}(h-h_{\text{exact}})^2},$$

where h (resp. h_{exact}) is the maximum interface height obtained numerically (resp. theoretically). Table 1 and Figure 17 summarise the results obtained for: this study, the PROST [21] and CLSVOF [63] implementations of Gerlach et al [66], the CSF implementation of Gueyffier et al [19], the balanced-forced level-set method of Herrmann [40] and the front-tracking method of Popinet and Zaleski [15].

Method/Resolution	8	16	32	64	128
Present study	0.1568	0.0279	0.00838	0.0018	0.000545
PROST [66]	0.2960	0.0818	0.0069	0.0018	–
CLSVOF [66]	0.3169	0.0991	0.0131	0.0033	–
CSF [19]	–	–	0.1168	0.0132	0.007
RLSG [40]	–	0.1116	0.0295	0.0114	0.0067
Front-tracking [15]	0.3018	0.0778	0.0131	0.0082	0.00645

Table 1. Convergence of the L_2 error on the maximum interface height of viscous capillary wave oscillations for different methods. The resolution is given in number of grid points per wavelength. $\rho_1/\rho_2=1$, $\mu_1/\mu_2=1$, La$=3000$.

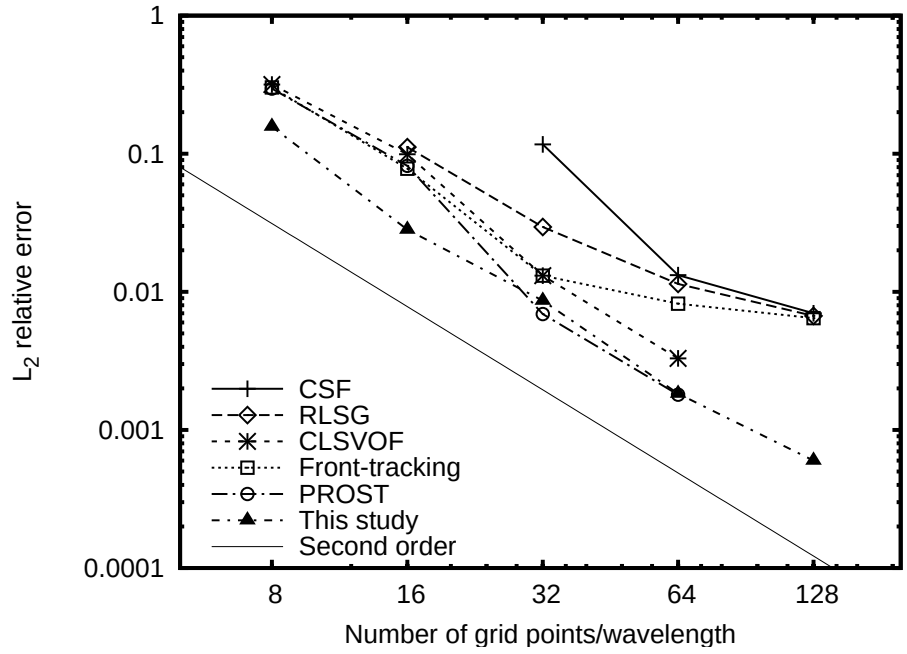


Fig. 17. Convergence of the L_2 error on the maximum interface height of viscous capillary wave oscillations for different methods.

Close to second-order convergence with spatial resolution is obtained for the present study, PROST and CLSVOF. The deviation from second-order convergence for the CSF and front-tracking methods at high resolutions may be due to the influence of the top and bottom boundaries (square domains were used in both these studies). The results of the present method are substantially more accurate at low resolutions than the results of the other methods given in Figure 17 and Table 1. This is important since in practice capillary waves are often the smallest features of interest and are usually captured at the limit of spatial resolution.

At high resolutions the present method is at least as accurate as PROST. The PROST method relies on systematic parabola-fitting through non-linear minimisation which has been demonstrated to be very accurate [21] but is also much more computationally expensive than the present method.

Finally the convergence results for an air/water capillary wave are given in Table 2. The density ratio is 850 and the dynamic viscosity ratio is 55.72. The Laplace number in the denser fluid is 3000 and the other parameters are identical to the constant density case above. The density and viscosity are defined using the filtered field $\widetilde{c}$ of equations (1) and (2). Although the errors are larger than for the constant density case they are still small and good convergence is obtained. These results can be compared with the results in the recent study by Herrmann [40] (Table 15) which show significantly larger errors and a saturation of convergence for resolutions above 64.

Method/Resolution	8	16	32	64	128
Present study	0.1971	0.0754	0.0159	0.00576	0.00313

Table 2. Convergence of the L_2 error on the maximum interface height of viscous capillary wave oscillations for parameters corresponding to an air/water interface. $\rho_1/\rho_2 = 850$, $\mu_1/\mu_2 = 55.72$, $\mathrm{La}_1 = 3000$.

6.4 Two-dimensional inviscid rising bubble

An inaccurate representation of surface tension is particularly noticeable for the case of a light gas bubble rising in a dense, low-viscosity fluid [20, 56, 23]. Figure 18 illustrates the results obtained for one of the cases considered by Francois et al [23]. A two-dimensional bubble of density 1.226 rises in a fluid of density 1000 due to a downward-acting gravity

acceleration of 9.81. The domain considered is $[-1,1]\times[-1,3]$ and the bubble of diameter $D=2/3$ is initially placed at $(0,0)$. The surface tension coefficient is $\sigma=728$ which gives a Bond number

$$\mathrm{Bo}=\frac{\rho\, g\, D^2}{\sigma}=5.983,$$

based on the external fluid properties. Both the internal and external fluids are inviscid. Although Francois et al considered viscous fluids, given the very large corresponding characteristic Reynolds number (of the order of 10^5 based on rise velocity) and the spatial resolutions considered in their study, their solution was dominated by numerical viscosity and it makes more sense in this case to simply consider the inviscid problem.

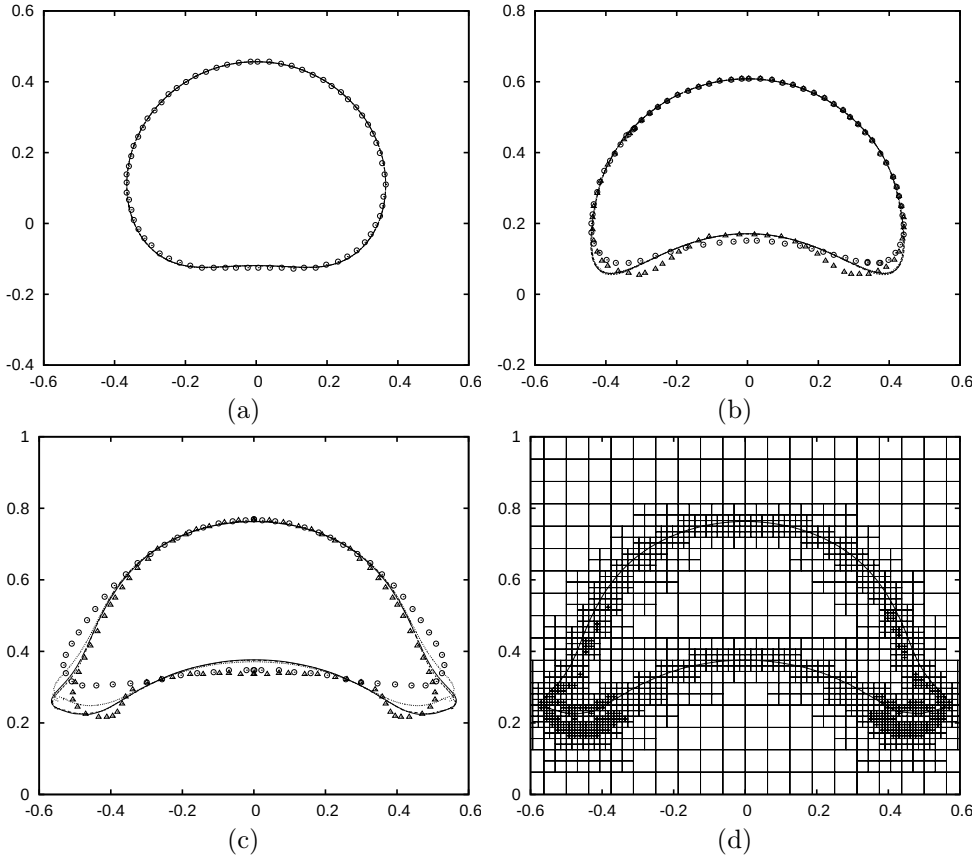


Fig. 18. Shape of the bubble at times (a) 0.2, (b) 0.35 and (c) 0.5 for different methods. Circles and triangles are the results obtained by Francois et al [23] (Figure 21) using their CSF and ghost-fluid (SSF) formulations respectively. The solid curve is the result of the current method for a constant resolution of 256×512, the dotted curve for a constant resolution of 128×256 and the dashed curve for an adaptive resolution with eight levels of refinement. (d) Adaptive mesh and interface at $t=0.5$.

As is clear from the symbols in Figure 18, Francois et al found that their results were sensitive to the details of the method used for the discretisation of the pressure: circle symbols for the continuous approach (CSF) and triangular symbols for the ghost-fluid sharp interface approach (SSF). The method described in the present article was applied at different resolutions with and without adaptive refinement. The solid line corresponds to the results obtained on the finest mesh using a constant resolution of 256×512. The dotted line gives the results for a constant resolution of 128×256 and the dashed line, the results for an adaptive resolution corresponding to an equivalent resolution of 256×512. The adaptive results were obtained by controlling the mesh size Δ so that the vorticity criterion

$$\Delta \, |\boldsymbol{\nabla} \times \boldsymbol{u}| < 1/2,$$

was verified at each timestep, with a limit on resolution of eight levels of refinement. A typical mesh is illustrated on Figure 18.(d). The vorticity generated at the interface is advected tangentially and accumulates at the stagnation points corresponding to the rear corners of the bubble. This increase in vorticity along the interface is matched by a corresponding increase in spatial resolution along the interface. The number and positions of the transitions between levels of refinement along the interface vary with bubble motion and shape evolution. This provides a good coverage of the consistency of the surface tension and curvature estimation schemes on the variable resolution quadtree mesh.

The results at $t = 0.2$ are identical for all methods and resolutions considered. At $t = 0.35$ the results for the current scheme are identical. They closely match the results of Francois et al in the front part of the bubble and fall between the CSF and SSF results for the rear part. At $t = 0.5$, the results for the current scheme also fall between the CSF and SSF results of Francois et al, but the coarse mesh results for the current scheme (dotted line) differ from the fine mesh results (a resolution dependence also observed by Francois et al). The constant fine mesh results and adaptive results are almost identical. This confirms the overall consistency of the adaptive scheme when resolution varies along the interface, for a flow with strong surface tension, low viscosity and large density ratio.

Resolution	# grid points	# timesteps	CPU time	Speed
256×512	203,161,600	1,550	12,550	16,188
128×256	18,022,400	550	1,067	16,890
256×512 (Adaptive)	4,368,344	1,317	405	10,786

Table 3. Total number of grid points advanced, number of timesteps, CPU time (seconds) and computing speed (grid point $\times$ timestep / second) for the rising bubble problem on different discretisations.

Table 3 gives a summary of the mesh sizes and CPU times (on a single CPU 2.5 GHz 32 bits Intel Pentium processor) for the different cases considered. The timestep is controlled so that the explicit surface tension stability constraint (18) is verified. The tolerance on the divergence of the velocity field γ_p is set to 10^{-3}. Setting a lower tolerance does not affect the results and mass is conserved to within less than 10^{-2} %. A maximum of eight multigrid iterations is sufficient to reach this tolerance in all cases, one to three iterations are sufficient for 90% of the timesteps. The computational speed on the adaptive mesh is smaller than the speed on the regular mesh due mainly to the extra cost incurred by the more complex

gradient operator at fine–coarse cell boundaries.

6.5 Three-dimensional capillary breakup of a liquid jet

The capillary instability of a liquid jet is a fundamental mechanism controlling the dynamics of many important two-phase flow phenomena such as liquid/gas atomisation. It is also well-suited to dynamic adaptive mesh refinement as a wide range of flow scales need to be resolved, particularly around the breakup point [69, 70, 71]. Dai and Schmidt [17] used their 3D adaptive unstructured tetrahedral mesh solver to study the case of the instability of a liquid cylinder with a free surface (i.e. the fluid motion of the gas phase was not resolved). Figure 19 reproduces their study (corresponding to Figure 16 of [17]) but for the case of a liquid jet surrounded by a lighter fluid.

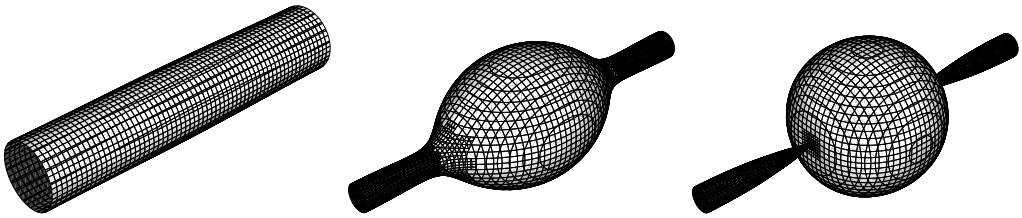


Fig. 19. Initial stages of a capillary jet instability: $k\, r_0 = 0.628$, $\epsilon = 0.02$, $\mathrm{La} = 238.34$. Physical parameters correspond to a jet of water in air. The non-dimensional times are from left to right 0, 11.32 and 12.1 (breakup time). The interface is represented using the VOF-reconstructed planar fragments in each cell.

The computational domain is a unit box centred on the origin. Symmetry boundary conditions are applied for $x = \pm\, 1/2$, $y = -\, 1/2$, $z = -\, 1/2$ so that only a quarter of the jet is simulated. Simple "outflow" boundary conditions are applied for $y = 1/2$ and $z = 1/2$ as

$$\begin{aligned} \partial_{\boldsymbol{n}} u_{\boldsymbol{n}} &= 0, \\ p &= 0, \end{aligned}$$

with $\boldsymbol{n}$ the direction normal to the boundary. The initial shape of the jet is given by radius

$$r(x) = r_0\,[1 + \epsilon \sin(k\,x)],$$

with $r_0 = 0.2$, $k = \pi$ and $\epsilon = 0.02$. In order to compare with the solutions obtained from linear stability analysis, the velocity field is initialised as

$$\boldsymbol{u} = \boldsymbol{\nabla}\phi \quad \text{with} \quad \phi \equiv B\, J_0(i\,k\,r) \sin(k\,x),$$

where the amplitude B is given by

$$B \equiv -\frac{\epsilon\, r_0\, c}{i\, k\, J_1(i\, k\, r_0)},$$

with c the inviscid growth rate obtained by Rayleigh [72] as

$$c^2 \equiv \frac{\sigma}{\rho\, r_0^3} \frac{I_1(k\, r_0)}{I_0(k\, r_0)}\, k\, r_0\, (1 - k^2\, r_0^2).$$

The Laplace number $\mathrm{La} \equiv \sigma\, r_0/(\rho\, \nu^2)$ is 238.34 as in [17]. The physical parameters for the surrounding fluid correspond to air/water density and viscosity ratios i.e. $\rho_g/\rho = 1.2/998$ and $\mu_g/\mu = 1.8 \times 10^{-5}/1.003 \times 10^{-3}$. The density and viscosity are defined using the filtered field $\widetilde{c}$ of equations (1) and (2).

The initial interface shape is resolved using five refinement levels. Every ten timesteps the resolution is adjusted so that for any cell containing the interface the condition $\Delta\, \kappa_{\max} < 1/5$ is verified. The maximum level of refinement is set arbitrarily to ten. The timestep is continuously adjusted so that the stability condition (18) is verified.

Figure 20 illustrates the clean and continuous increase of resolution when approaching the breakup point. One can also note that – despite being essentially discontinuous – the VOF-reconstructed planar fragments provide a high-quality description of the interface.

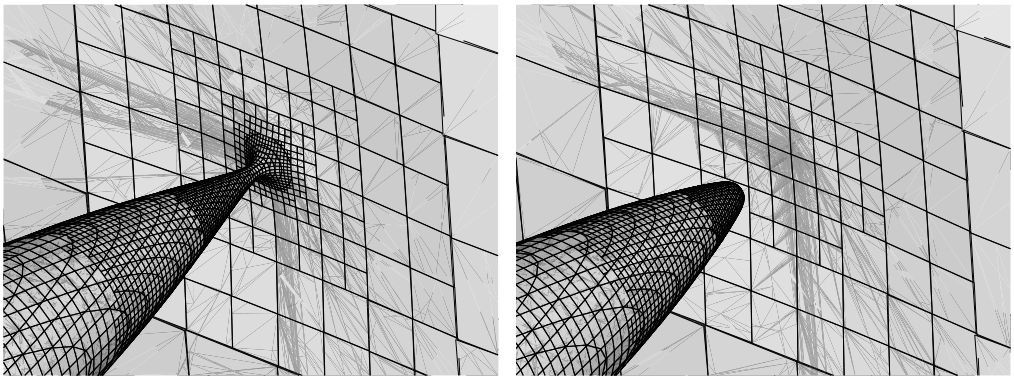


Fig. 20. Details of the interface just before ($(t_0 - t)/T_\sigma = 1.5 \times 10^{-3}$) and after breakup ($(t_0 - t)/T_\sigma = -8.5 \times 10^{-3}$). The interface is represented using the VOF-reconstructed planar fragments in each cell. $k\, r_0 = 0.628$, $\epsilon = 0.02$, $\mathrm{La} = 238.34$.

The results are comparable to those of Dai and Schmidt although the present simulation is not limited to a singly-connected topology. As with any VOF method no special treatment is required when breakup occurs and the simulation deals transparently with multiply

connected interfaces (Figure 21). The maximum variation in the total mass is less than 0.02%.

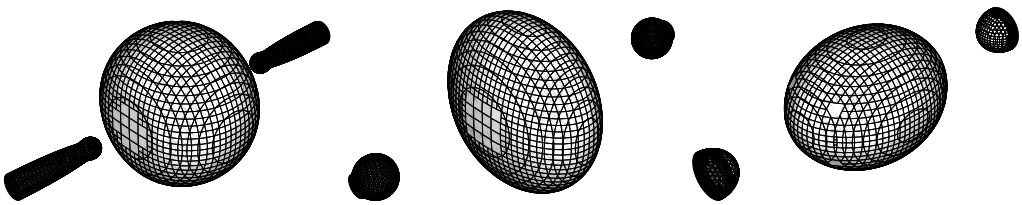


Fig. 21. Late stages of a capillary jet instability: $k\, r_0 = 0.628$, $\epsilon = 0.02$, La = 238.34. The non-dimensional times are from left to right 12.31, 13.03 and 15.36.

Weber extended the study of Rayleigh to the case of a viscous fluid [73] and obtained the following equation for the growth rate of the perturbation

$$c^2 + c\, 3\,\nu\, k^2 - \frac{\sigma}{2\,\rho\, r_0^3}\,(1 - k^2\, r_0^2)\, k^2\, r_0^2 = 0.$$

Figure 22 summarises the results obtained for the evolution of the initial perturbation. Two numerical estimates of the perturbation are considered: the relative deformation associated with the maximum radial extent of the interface $(r_{\max} - r_0)/(r_0\,\epsilon)$, and the relative deformation associated with the minimum radial extent $(r_0 - r_{\min})/(r_0\,\epsilon)$. Obviously the minimum radius vanishes at the time of breakup and the associated relative deformation takes a maximum value of $1/\epsilon$ (≈ 50). The time is made non-dimensional using $T_\sigma \equiv (\rho\, r_0^3/\sigma)^{1/2}$ as reference timescale. Simulations were performed at three different Laplace numbers: 2000, 238.34 and 23.834.

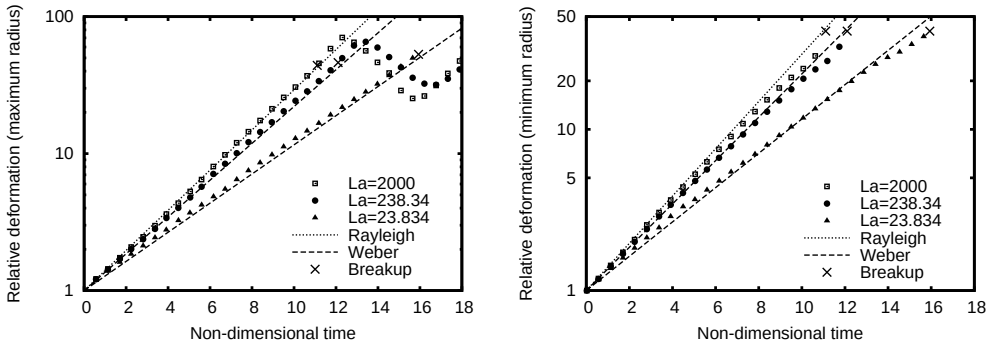


Fig. 22. Evolution of the relative deformations of the interface. (a) $(r_{\max} - r_0)/(r_0\,\epsilon)$. (b) $(r_0 - r_{\min})/(r_0\,\epsilon)$.

For the maximum-radius relative deformation excellent agreement is obtained with either the inviscid theory of Rayleigh (La = 2000) or the viscous solution of Weber. Remarkably the growth rate is close to constant right up to the point of breakup. The good agreement with the inviscid theory at high Laplace number is also indicative of the overall low numerical

dissipation of the discretisation scheme.

The minimum-radius relative deformation behaves similarly at initial times but departs from a constant growth rate before reaching the breakup point. The fluid motion asymptotically close to breakup can be shown to be a self-similar singularity of the Navier–Stokes equations with characteristic length scale

$$l_\nu = \frac{\nu^2 \rho}{\sigma}.$$

In this regime ($r_{\min}$ comparable to l_ν) the minimum radius is proportional to $t_0 - t$ with t_0 the time of breakup [74, 71]. This regime is easily observed experimentally and numerically provided the ratio r_0/l_ν (i.e. the Laplace number) is small enough. Far enough away from breakup ($l_\nu \ll r_{\min} \ll r_0$) and provided the Laplace number is large enough the fluid motion can be considered inviscid and scaling arguments give $r_{\min} \sim (t_0 - t)^{2/3}$ [75, 71].

Figure 23 shows the evolution of the minimum radius close to breakup for La = 2000 and 23.834. As expected two main regimes are clearly distinguished depending on the Laplace number. At La = 2000 the evolution of the minimum radius is accurately described by the inviscid theory for more than a decade of non-dimensional time. For La = 23.834 a transition between the two regimes occurs for $r_{\min}/r_0 \approx l_\nu/r_0 = 1/\mathrm{La}$, the viscous regime ($r_{\min}/r_0 < 1/\mathrm{La}$) being accurately described by the self-similar solution ($r_{\min} \sim t_0 - t$) while the inertial regime ($r_{\min}/r_0 > 1/\mathrm{La}$) is in reasonable agreement with the inviscid theory ($r_{\min} \sim (t_0 - t)^{2/3}$).

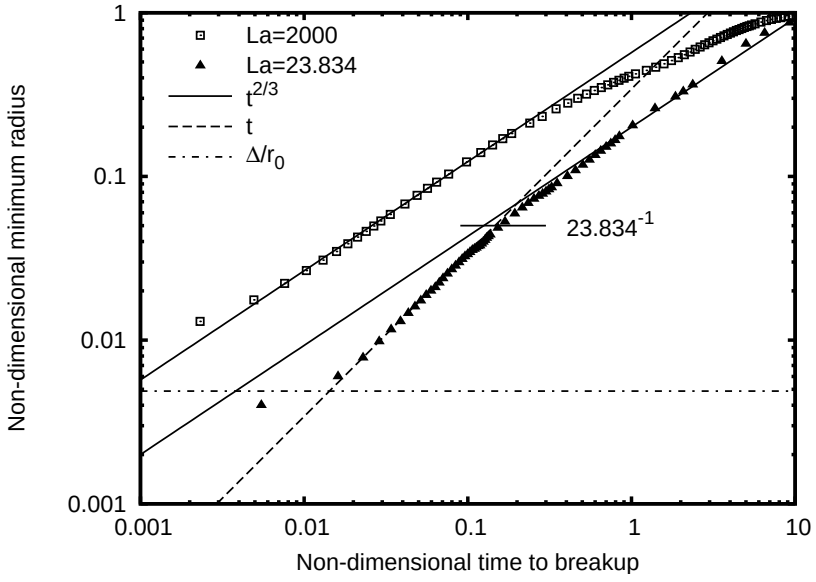
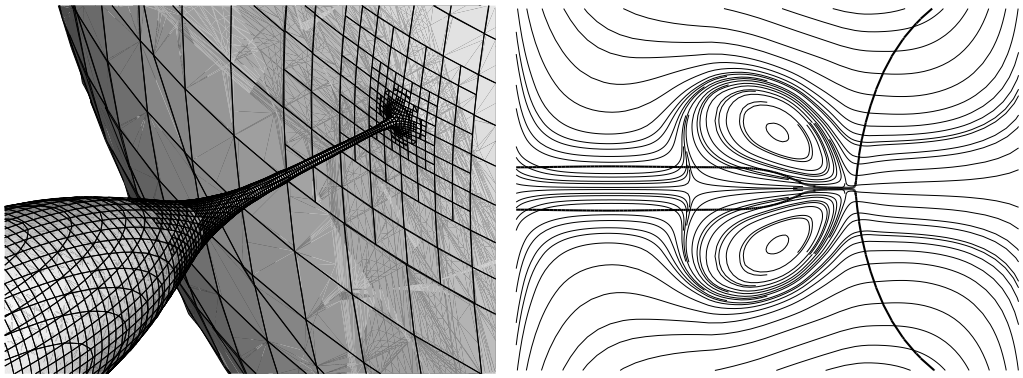


Fig. 23. Evolution of the non-dimensional minimal radius $r_{\min}/r_0$ as a function of the non-dimensional time to breakup $(t_0 - t)/T_\sigma$. The dash-dotted line (Δ/r_0) indicates the limit of spatial resolution of the simulations.

The self-similar solution close to breakup predicts the formation of a thin liquid thread ("microthread" [76]) connecting the two regions about to separate. Although these microthreads occur independently of the Laplace number, their characteristic length scale is l_ν and they will be more easily seen at low Laplace numbers. Figure 24 illustrates the microthread obtained numerically just before breakup for $\mathrm{La} = 23.834$. It looks very similar to the microthreads observed experimentally [76, 69, 71]. The complex structure of the corresponding velocity field is given in Figure 24.b. Interestingly the solution has two stagnation points on the axis of symmetry (aside from the stagnation points on the symmetry planes at $x = \pm 1/2$): one within the microthread (which will eventually lead to breakup) and a second one within the "neck" region. This result is consistent with the similarity solution obtained by Eggers [74] (see also Figure 17.b of [71]). A recirculation region within the gas phase is associated with these two stagnation points.



(a) (b)

Fig. 24. Detail of the interface close to breakup ($(t_0 - t)/T_\sigma = 1.5 \times 10^{-2}$) for a very viscous fluid: $k\,r_0 = 0.628$, $\epsilon = 0.02$, $\mathrm{La} = 23.834$. (a) VOF-reconstructed interface. (b) Interface and streamlines in a cross-section on the axis of symmetry.

Figure 25 represents the evolution with time of the total number of leaf cells used to discretise the problem. The step increments in the number of cells (at $t \approx 7$, 11, 12, ...) correspond to the addition of one additional level of refinement when the curvature of the interface locally reaches the threshold $\Delta\,\kappa_{\max} < 1/5$. As expected the maximum number of cells ($\approx$9,000) is reached at the time of breakup. After breakup the number of cells stabilises to about 2,500. This is larger than the initial number of cells ($\approx$1,500) because the newly-created, smaller droplets require a higher resolution (rightmost Figure 21). These numbers can be compared to the number of cells which would be required with:

1. a 3D static regular Cartesian grid with an equivalent resolution (ten levels of refinement): $(2^{10})^3 \approx 10^9$.
2. a 2D-axisymmetric static regular Cartesian grid: $(2^{10})^2 \approx 10^6$.
3. a dynamically-refined octree grid with all the cells containing the interface refined at the same resolution $\Delta(t)$ controlled by $\Delta(t) \max(\kappa_{\max})(t) < 1/5$: dashed curve in Figure 25.

It is clear that orders-of-magnitude savings in computation size can be achieved when using dynamic adaptation along the interface for this particular problem. Option 3 above which has been used in a number of recent studies [28, 30, 8, 36, 31, 9, 38, 64, 33, 39] is simple to implement within a VOF framework but does not make full use of adaptivity. The total CPU time ($\approx$6,000 timesteps) for the simulation of Figures 19–21, 25 is about three hours on a 2.5 GHz 32 bits Intel Pentium processor.

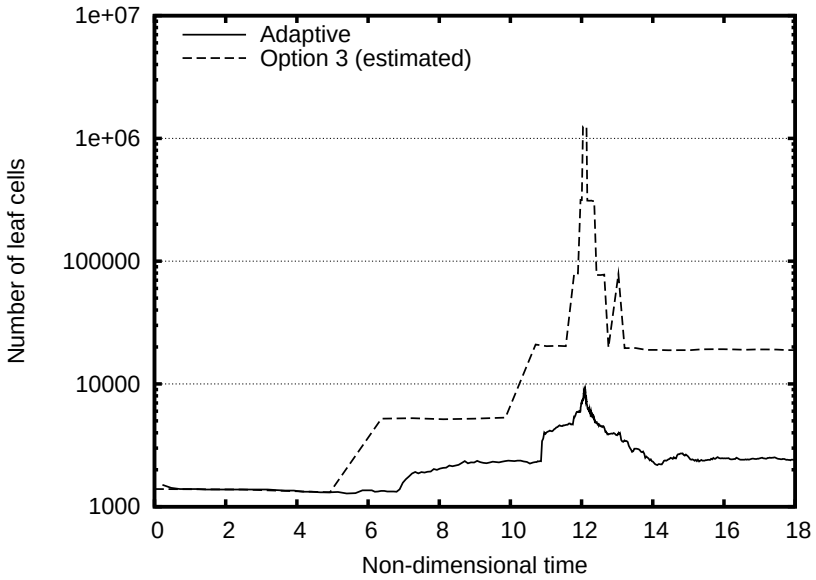


Fig. 25. Evolution of the total number of leaf cells required to discretise the jet while verifying the criterion $\Delta\,\kappa_{\max} < 1/5$. Maximum of ten levels of refinement. $k\,r_0 = 0.628$, $\epsilon = 0.02$, $\mathrm{La} = 238.34$.

Finally the three-dimensional results are cross-validated against axisymmetric results both with and without variable resolution along the interface. The axisymmetric results are obtained using a cylindrical-coordinates version of the code with $k\, r_0 = 0.628$, $\epsilon = 0.02$, $\mathrm{La} = 2000$. Two types of refinement are considered:

a) A dynamically refined quadtree grid with all the cells containing the interface refined at the same resolution $\Delta(t)$ controlled by $\Delta(t) \max\left(\kappa_{\max}(\boldsymbol{s})\right)(t) < 1/5$, with $\boldsymbol{s}$ the surface coordinate on the interface (equivalent to option 3 above).
b) A dynamically refined quadtree grid with variable resolution along the interface $\Delta(\boldsymbol{s}, t)$ controlled by $\Delta(\boldsymbol{s}, t)\, \kappa_{\max}(\boldsymbol{s}, t) < 1/5$ (corresponding to the 3D simulation).

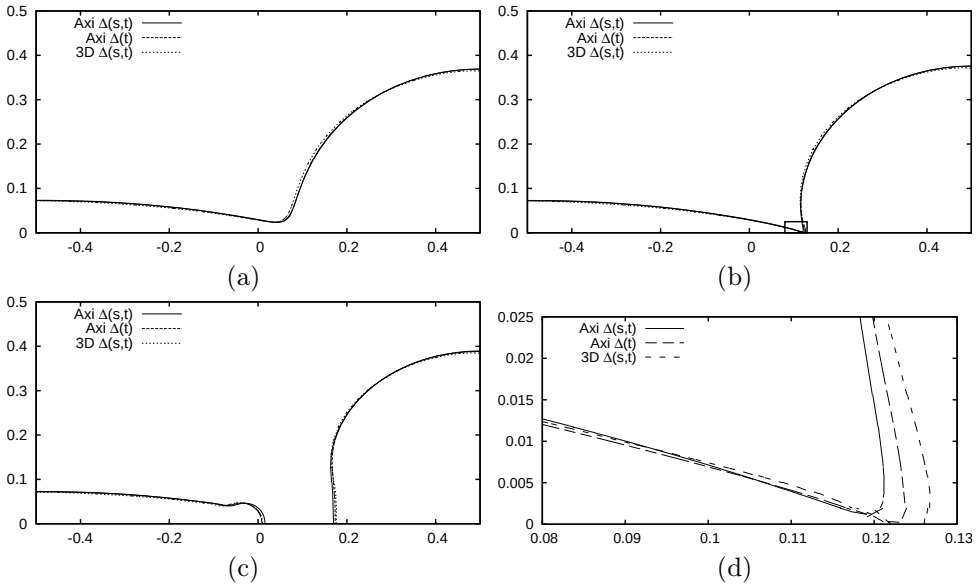


Fig. 26. Comparison of axisymmetric and three-dimensional interface profiles. $k\, r_0 = 0.628$, $\epsilon = 0.02$, $\mathrm{La} = 2000$. (a) $t/T_\sigma = 11.51$. (b) $t/T_\sigma = 11.61$. (c) $t/T_\sigma = 11.79$. (d) Detail corresponding to the rectangular area in (b).

Figure 26 summarises the results for these three cases: before breakup (a), close to breakup (b) and (d), and some time after breakup (c). The three-dimensional profile is obtained as a cross-section on the axis of symmetry of the 3D VOF-reconstructed surface. The 3D solution is almost exactly axisymmetric and the choice of cross-section plane does not affect the results. The general agreement between the three solutions is excellent. Before breakup the axisymmetric results are almost identical. The small discrepancy between the 3D and axisymmetric results could be due to the fact that the second principal radius of curvature (in the plane perpendicular to the plane of axisymmetry) is computed completely differently in the axisymmetric and 3D cases. Breakup occurs slightly later for the axisymmetric $\Delta(\boldsymbol{s}, t)$ case which explains the somewhat larger discrepancy between the axisymmetric results at later times (Figure 26.c).

Table 4 gives the corresponding mesh sizes, number of timesteps and speeds (per grid point and timestep). The numbers for the 3D $\Delta(t)$ case are estimated assuming the same computing speed as in the 3D $\Delta(\boldsymbol{s}, t)$ case. The axisymmetric $\Delta(\boldsymbol{s}, t)$ and 3D cases save

factors of 5 and 49 respectively, in mesh sizes and CPU time compared to the corresponding $\Delta(t)$ cases.

Discretisation	# grid points	# timesteps	CPU time	Speed
3D $\Delta(t)$ (estimated)	1,281,387,600	5,952	628,747	2,038
3D $\Delta(\boldsymbol{s},t)$	25,810,164	5,952	12,660	2,038
Axisymmetric $\Delta(t)$	9,070,300	5,017	1,482	6,120
Axisymmetric $\Delta(\boldsymbol{s},t)$	1,805,694	5,049	247	7,310

Table 4. Total number of grid points advanced, number of timesteps, CPU time (seconds) and computing speed (grid point $\times$ timestep / second) for the capillary breakup problem on different discretisations. $k\,r_0 = 0.628$, $\epsilon = 0.02$, $\mathrm{La} = 2000$.

7 Conclusion

The combination of a quad/octree discretisation, balanced-force CSF surface tension scheme and generalised height-function curvature estimation implemented within the Gerris solver has been demonstrated to give accurate and efficient solutions for surface-tension-driven flows. The VOF method, height-function and CSF formulations have been generalised to a fully-adaptive quad/octree discretisation allowing refinement along the interface. For the case of capillary breakup of a three-dimensional liquid jet, this leads to a reduction by a factor of fifty of the mesh size compared to using a constant resolution along the interface. As a result the dynamics of the breakup can be accurately and efficiently modelled across the spatial range spanning the transition from inertial to viscous regimes.

The height-function curvature estimation technique has been generalised to be consistent even at low interface resolutions. Test cases demonstrate second-order curvature convergence across the whole range of spatial resolutions. Clean and robust surface-tension-driven interface breakup is demonstrated using this generalised technique.

In contrast to the recent study of Francois et al [23] the long-standing problem of "parasitic currents" around a stationary droplet is shown to be solved by the combination of appropriate implementations of a balanced-force CSF approach and height-function curvature estimation. Exact balance (to machine accuracy) between surface-tension force and pressure gradient is recovered irrespective of spatial resolution and viscosity provided the interface is allowed enough time to relax to its equilibrium position. As expected this relaxation time is shown to be comparable to the viscous dissipation timescale. Furthermore the equilibrium shape of the interface shows second-order convergence towards the exact circular equilibrium shape as a function of spatial resolution.

Several issues require further investigation. The new "translating droplet in equilibrium" test case shows that the elimination of parasitic currents in the stationary case does not guarantee that the numerical solutions of other problems will be free from surface-tension-induced velocity errors. Also, while the balanced-force CSF method is shown to perform well it has the important shortcoming of not conserving momentum (in contrast to the continuum surface stress (CSS) approach of [19]). Initial tests suggest that this can be a major issue for problems involving long time-integrations and high surface-tension such as the rise of millimetric air bubbles in water.

Finally, I hope that providing the full implementation of the method as part of the freely-modifiable open source software Gerris [44] will encourage other people to tackle these issues.

Acknowledgements

This work was funded by the New Zealand Foundation for Research, Science and Technology under contract no. CO1X0204.

References

[1] C. Hirt, B. Nichols, Volume of fluid method for the dynamics of free boundaries, Journal of Computational Physics 39 (1) (1981) 201–225.

[2] J. Li, Calcul d'interface affine par morceaux, Comptes rendus de l'Académie des sciences. Série II, Mécanique, physique, chimie, astronomie 320 (8) (1995) 391–396.

[3] W. Rider, D. Kothe, Reconstructing volume tracking, Journal of Computational Physics 141 (2) (1998) 112–152.

[4] R. Scardovelli, S. Zaleski, Direct numerical simulation of free-surface and interfacial flow, Annual Review of Fluid Mechanics 31 (1) (1999) 567–603.

[5] M. Sussman, P. Smereka, S. Osher, A level set approach for computing solutions to incompressible two-phase flow, Journal of Computational Physics 114 (1) (1994) 146–159.

[6] M. Sussman, A. Almgren, J. Bell, P. Colella, L. Howell, M. Welcome, An adaptive level set approach for incompressible two-phase flows, Journal of Computational Physics 148 (1) (1999) 81–124.

[7] V. Sochnikov, S. Efrima, Level set calculations of the evolution of boundaries on a dynamically adaptive grid, International Journal for Numerical Methods in Engineering 56 (2003) 1913–1929.

[8] H. Kohno, T. Tanahashi, Numerical analysis of moving interfaces using a level set method coupled with adaptive mesh refinement, International Journal for Numerical Methods in Fluids 45 (2004) 921–944.

[9] X. Zheng, J. Lowengrub, A. Anderson, V. Cristini, Adaptive unstructured volume remeshing–II: Application to two-and three-dimensional level-set simulations of multiphase flow, Journal of Computational Physics 208 (2) (2005) 626–650.

[10] H. Liu, S. Krishnan, S. Marella, H. Udaykumar, Sharp interface cartesian grid method ii: A technique for simulating droplet interactions with surfaces of arbitrary shape, Journal of Computational Physics 210 (1) (2005) 32 – 54.

[11] E. Marchandise, J.-F. Remacle, A stabilized finite element method using a discontinuous level set approach for solving two phase incompressible flows, Journal of Computational Physics 219 (2) (2006) 780 – 800.

[12] E. Marchandise, P. Geuzaine, N. Chevaugeon, J.-F. Remacle, A stabilized finite element method using a discontinuous level set approach for the computation of bubble dynamics, Journal of Computational Physics 225 (1) (2007) 949 – 974.

[13] O. Desjardins, V. Moureau, H. Pitsch, An accurate conservative level set/ghost fluid method for simulating turbulent atomization, Journal of Computational Physics 227 (18) (2008) 8395 – 8416.

[14] D. Fyfe, E. Oran, M. Fritts, Surface tension and viscosity with Lagrangian hydrodynamics on a triangular mesh, Journal of Computational Physics 76 (2) (1988) 349–384.

[15] S. Popinet, S. Zaleski, A front-tracking algorithm for accurate representation of surface tension, International Journal for Numerical Methods in Fluids 30 (6) (1999) 775–793.

[16] S. Shin, S. Abdel-Khalik, V. Daru, D. Juric, Accurate representation of surface tension using the level contour reconstruction method, Journal of Computational Physics 203 (2) (2005) 493–516.

[17] M. Dai, D. Schmidt, Adaptive tetrahedral meshing in free-surface flow, Journal of Computational Physics 208 (1) (2005) 228–252.

[18] B. Lafaurie, C. Nardone, R. Scardovelli, S. Zaleski, G. Zanetti, Modelling merging and fragmentation in multiphase flows with SURFER, Journal of Computational Physics 113 (1) (1994) 134–147.

[19] D. Gueyffier, A. Nadim, J. Li, R. Scardovelli, S. Zaleski, Volume of fluid interface tracking with smoothed surface stress methods for three-dimensional flows, Journal of Computational Physics 152 (1998) 423–456.

[20] M. Kang, R. Fedkiw, X. Liu, A boundary condition capturing method for multiphase incompressible flow, Journal of Scientific Computing 15 (3) (2000) 323–360.

[21] Y. Renardy, M. Renardy, PROST - A parabolic reconstruction of surface tension for the volume-of-fluid method, Journal of Computational Physics 183 (2) (2002) 400–421.

[22] D. Harvie, M. Davidson, M. Rudman, An analysis of parasitic current generation in volume of fluid simulations, Applied Mathematical Modelling 30 (10) (2006) 1056–1066.

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

[24] I. Ginzburg, G. Wittum, Two-phase flows on interface refined grids modeled with VOF, staggered finite volumes, and spline interpolants, Journal of Computational Physics 166 (2) (2001) 302–335.

[25] S. Cummins, M. Francois, D. Kothe, Estimating curvature from volume fractions, Computers and Structures 83 (6-7) (2005) 425–434.

[26] M. Sussman, M. Ohta, Improvements for calculating two-phase bubble and drop motion using an adaptive sharp interface method, Fluid Dyn. Mater. Proc. 3 (1) (2007) 21–36.

[27] A. Almgren, J. Bell, P. Colella, L. Howell, M. Welcome, A conservative adaptive projection method for the variable density incompressible Navier–Stokes equations, Journal of Computational Physics 142 (1) (1998) 1–46.

[28] J. Jeong, D. Yang, Finite element analysis of transient fluid flow with free surface using VOF (volume-of-fluid) method and adaptive grid, International Journal for Numerical Methods in Fluids 26 (10) (1998) 1127–1154.

[29] M. Sussman, A parallelized, adaptive algorithm for multiphase flows in general geometries, Computers & Structures 83 (6-7) (2005) 435 – 444.

[30] D. Greaves, A quadtree adaptive method for simulating fluid flows with moving interfaces, Journal of Computational Physics 194 (1) (2004) 35–56.

[31] A. Theodorakakos, G. Bergeles, Simulation of sharp gas–liquid interface using VOF method and adaptive grid local refinement around the interface, International Journal for Numerical Methods in Fluids 45 (4) (2004) 421–439.

[32] F. Losasso, R. Fedkiw, S. Osher, Spatially adaptive techniques for level set methods and incompressible flow, Computers and Fluids 35 (2006) 457–462.

[33] C. Min, F. Gibou, A second order accurate level set method on non-graded adaptive Cartesian grids, Journal of Computational Physics 225 (1) (2007) 300–321.

[34] G. Compère, E. Marchandise, J.-F. Remacle, Transient adaptivity applied to two-phase incompressible flows, Journal of Computational Physics 227 (3) (2008) 1923 – 1942.

[35] S. Popinet, Gerris: a tree-based adaptive solver for the incompressible Euler equations in complex geometries, Journal of Computational Physics 190 (2) (2003) 572–600.

[36] J. Wang, A. Borthwick, R. Taylor, Finite-volume-type VOF method on dynamically adaptive quadtree grids, International Journal for Numerical Methods in Fluids 45 (5) (2004) 485–508.

[37] F. Losasso, F. Gibou, R. Fedkiw, Simulating water and smoke with an octree data structure, ACM Transactions on Graphics (TOG) 23 (3) (2004) 457–462.

[38] D. Greaves, Simulation of viscous water column collapse using adapting hierarchical grids, International Journal for Numerical Methods in Fluids 50 (6) (2006) 693–711.

[39] N. Nikolopoulos, A. Theodorakakos, G. Bergeles, Three-dimensional numerical investigation of a droplet impinging normally onto a wall film, Journal of Computational Physics 225 (1) (2007) 322–341.

[40] M. Herrmann, A balanced force refined level set grid method for two-phase flows on unstructured flow solver grids, Journal of Computational Physics 227 (4) (2008) 2674–2706.

[41] M. Malik, E. Sheung-Chi Fan, M. Bussmann, Adaptive VOF with curvature-based refinement, International Journal for Numerical Methods in Fluids 55 (7) (2007) 693–712.

[42] M. Rudman, Volume-tracking methods for interfacial flows calculations, International Journal for Numerical Methods in Fluids 24 (7) (1997) 671–691.

[43] M. Sussman, K. Smith, M. Hussaini, M. Ohta, R. Zhi-Wei, A sharp interface method for incompressible two-phase flows, Journal of Computational Physics 221 (2) (2007) 469–505.

[44] S. Popinet, The Gerris Flow Solver, http://gfs.sf.net (2007).

[45] S. Popinet, Gerris Test Suite, http://gfs.sourceforge.net/tests/tests/index.html (2007).

[46] A. Chorin, On the convergence of discrete approximations to the Navier–Stokes equations, Mathematics of Computation 23 (106) (1969) 341–353.

[47] R. Alcouffe, A. Brandt, J. Dendy Jr, J. Painter, The multi-grid method for the diffusion equation with strongly discontinuous coefficients, SIAM Journal on Scientific and Statistical Computing 2 (1981) 430.

[48] O. Tatebe, N. Melson, T. Manteuffel, S. McCormick, The multigrid preconditioned conjugate gradient method, in: Sixth Copper Mountain Conference on Multigrid Methods, NASA, 1993, pp. 621–634.

[49] T. Chan, W. Wan, Robust multigrid methods for nonsmooth coefficient elliptic linear systems, Journal of Computational and Applied Mathematics 123 (1-2) (2000) 323–352.

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

[51] A. Almgren, J. Bell, W. Crutchfield, Approximate projection methods: part I. inviscid analysis, SIAM Journal on Scientific Computing 22 (4) (2000) 1139–1159.

[52] R. Scardovelli, S. Zaleski, Analytical relations connecting linear interfaces and volume fractions in rectangular grids, Journal of Computational Physics 164 (1) (2000) 228–237.

[53] R. Scardovelli, S. Zaleski, Interface reconstruction with least-square fit and split eulerian–lagrangian advection, International Journal for Numerical Methods in Fluids 41 (2003) 251–274.

[54] E. Aulisa, S. Manservisi, R. Scardovelli, S. Zaleski, Interface reconstruction with least-squares fit and split advection in three-dimensional Cartesian geometry, Journal of Computational Physics 225 (2) (2007) 2301–2319.

[55] R. DeBar, Fundamentals of the KRAKEN code, Tech. rep., UCID-17366, California Univ., Livermore (USA). Lawrence Livermore Lab. (1974).

[56] D. Lörstad, L. Fuchs, High-order surface tension VOF-model for 3D bubble flows with high density ratio, Journal of Computational Physics 200 (1) (2004) 153–176.

[57] S. Zalesak, Fully multidimensional flux-corrected transport algorithms for fluids, Journal of Computational Physics 31 (3) (1979) 335–362.

[58] O. Ubbink, R. Issa, A method for capturing sharp fluid interfaces on arbitrary meshes, Journal of Computational Physics 153 (1) (1999) 26–50.

[59] W. Noh, P. Woodward, SLIC/simple line interface calculation, International Conference on Numerical Methods in Fluid Dynamics, 5th, Enschede, Netherlands, June 28-July 2, 1976, .

[60] E. Aulisa, S. Manservisi, R. Scardovelli, S. Zaleski, A geometrical area-preserving volume-of-fluid method, Journal of Computational Physics 192 (2003) 355–364.

[61] M. Sussman, A second order coupled level set and volume-of-fluid method for computing growth and collapse of vapor bubbles, Journal of Computational Physics 187 (1) (2003) 110 – 136.

[62] J. Brackbill, D. Kothe, C. Zemach, A continuum method for modeling surface tension, Journal of Computational Physics 100 (2) (1992) 335–354.

[63] M. Sussman, E. Puckett, A coupled level set and volume of fluid method for computing 3D and axisymmetric incompressible two-phase flows, Journal of Computational Physics 162 (2) (2000) 301–337.

[64] X. Yang, A. James, J. Lowengrub, X. Zheng, V. Cristini, An adaptive coupled level-set/volume-of-fluid interface capturing method for unstructured triangular grids, Journal of Computational Physics 217 (2) (2006) 364–394.

[65] M. Torrey, L. Cloutman, R. Mjolsness, C. Hirt, NASA-VOF2D: a computer program for incompressible flows with free surfaces, Tech. rep., Los Alamos National Laboratory (1985).

[66] D. Gerlach, G. Tomar, G. Biswas, F. Durst, Comparison of volume-of-fluid methods for surface tension-dominant two-phase flows, International Journal of Heat and Mass Transfer 49 (3-4) (2006) 740–754.

[67] H. Lamb, Hydrodynamics, Cambridge University Press, 1971.

[68] A. Prosperetti, Motion of two superposed viscous fluids, Physics of Fluids 24 (1981) 1217.

[69] X. Shi, M. Brenner, S. Nagel, A cascade of structure in a drop falling from a faucet, Science 265 (5169) (1994) 219–222.

[70] M. Brenner, J. Eggers, K. Joseph, S. Nagel, X. D. Shi, Breakdown of scaling in droplet fission at high Reynolds number, Physics of Fluids 9 (6) (1997) 1573–1590.

[71] J. Eggers, Nonlinear dynamics and breakup of free-surface flows, Reviews of Modern Physics 69 (3) (1997) 865–930.

[72] L. Rayleigh, On the instability of a cylinder of viscous liquid under capillary force, Philosophical Magazine 34 (207) (1892) 145–154.

[73] C. Weber, Disintegration of liquid jets, Zeitschrift für Angewandte Mathematik und Mechanik 11 (2) (1931) 136–159.

[74] J. Eggers, Universal pinching of 3D axisymmetric free-surface flow, Physical Review Letters 71 (21) (1993) 3458–3460.

[75] D. Peregrine, G. Shoker, A. Symon, The bifurcation of liquid bridges, Journal of Fluid Mechanics 212 (1990) 25–39.

[76] T. Kowalewski, On the separation of droplets from a liquid jet, Fluid Dynamics Research 17 (1996) 121–145.