An unstructured finite element model for incompressible two-phase flow based on a monolithic conservative level set method

<table>
<thead>
<tr>
<th>Item Type</th>
<th>Article</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authors</td>
<td>Quezada de Luna, Manuel; Haydel Collins, J.; Kees, Christopher E.</td>
</tr>
<tr>
<td>Eprint version</td>
<td>Post-print</td>
</tr>
<tr>
<td>DOI</td>
<td>10.1002/fld.4817</td>
</tr>
<tr>
<td>Publisher</td>
<td>Wiley</td>
</tr>
<tr>
<td>Journal</td>
<td>International Journal for Numerical Methods in Fluids</td>
</tr>
<tr>
<td>Rights</td>
<td>Archived with thanks to International Journal for Numerical Methods in Fluids</td>
</tr>
<tr>
<td>Download date</td>
<td>2023-11-01 04:28:28</td>
</tr>
<tr>
<td>Link to Item</td>
<td><a href="http://hdl.handle.net/10754/662253">http://hdl.handle.net/10754/662253</a></td>
</tr>
</tbody>
</table>
An unstructured finite element model for incompressible two-phase flow based on a monolithic conservative level set method

Manuel Quezada de Luna *  J. Haydel Collins †  Christopher E. Kees ‡

Abstract

We present a robust numerical method for solving incompressible, immiscible two-phase flows. The method extends both a monolithic phase conservative level set method with embedded redistancing and a semi-implicit high-order projection scheme for variable-density flows. The level set method can be initialized conveniently via a simple phase indicator field instead of a signed distance function. To process the indicator field into a signed distance function, we propose a new PDE-based redistancing method. We also improve the monolithic level set scheme to provide more accuracy and robustness in full two-phase flow simulations. Specifically, we perform an extra step to ensure convergence to the signed distance level set function and simplify other aspects of the original scheme. Lastly, we introduce consistent artificial viscosity to stabilize the momentum equations in the context of the projection scheme. This stabilization is algebraic, has no tunable parameters and is suitable for unstructured meshes and arbitrary refinement levels. The overall methodology includes few numerical tuning parameters; however, for the wide range of problems that we solve, we identify only one parameter that strongly affects performance of the computational model and provide a value that provides accurate results across all the benchmarks presented. This methodology results in a robust, accurate, and efficient two-phase flow model, which is mass- and volume-conserving on unstructured meshes and has low user input requirements, making it attractive for real-world applications.

1 Introduction

Understanding the behavior of immiscible fluids with distinct material characteristics (e.g. water and air) is important for many engineering, industrial applications, and water resources. Oil and gas reservoirs, combustion engines, and open navigation channels are all examples of domains in which flow phenomena involving multiple fluids, i.e. multiphase flow, are critical. In order to accurately describe the evolving interface between individual fluid subdomains, high fidelity numerical models are required. The methods should aim for computational efficiency along with preservation of qualitative properties of the material continuum. In the case of multiphase simulations concerning divergence-free velocity fields, maintaining the conservation of mass and volume is imperative.

In general, multiphase flow is modeled by solving the Navier-Stokes equations with spatially varying material parameters. We consider an Eulerian description for the fluid motion along with continuous Galerkin finite elements for discretization of all equations in space. For Eulerian methods, the fluid motion is calculated from a stationary frame of reference on a fixed computational grid. These techniques can resolve large fluid deformations, however, a representation of the phase interface is required.

Two routinely implemented methods for representing dynamic interfaces between discrete phases are the Volume of Fluid (VOF) method by [21] and level set techniques by [38, 43]. The VOF method assigns phase identities, e.g. fluids A and B, via a characteristic function that equals one in fluid A and zero in fluid B. Grid cell averaging of this phase indicator function results in the designation of a cell volume fraction in the range [0, 1]. An interface is then reconstructed in cells with intermediate volume fraction and the VOF re-initialized. For many multiphase flow applications, the characteristic function is transported using velocities obtained from the (incompressible) Navier-Stokes equations. This solution produces density and viscosity fields requiring subsequent interfacial reconstruction. Once the phase boundary location is accurately reconstructed, material properties can then be designated for each fluid subdomain.

*KAUST CEMSE, manuel.quezada@kaust.edu.sa. Research performed as a post-doc at USACE-ERDC-CHL.
†USACE-MVN, jason.h.collins@usace.army.mil. Research performed as a participant in ERDC University.
‡USACE-ERDC-CHL, christopher.e.kees@usace.army.mil.
In the level set method, the interface between the phases is represented implicitly by a level set of a scalar function defined on the entire domain; e.g., as the zero level set of the signed distance function (SDF) to the interface. The SDF returns a zero value at the interface and either a positive or a negative distance for subdomains fluid A and fluid B respectively. The level set function is advected by the fluid velocity field produced by the (incompressible) Navier-Stokes equations.

Both the VOF and the level set methods have limitations when implemented alone. Despite maintaining volume (or phase) conservation, the VOF requires reconstruction of interface locations from less precise cell averages. The interface is effectively smeared over some distance specific to the numerical scheme and then reconstructed to be sharp based on geometric approximations. Level set methods do not require interface reconstruction; however, they lack a discrete conservation property; phases enclosed by the interface can suffer from a significant loss of volume due to the accumulation of discrete conservation errors over time.

There are hybrid methods combining ideas of the VOF, level set, and particle methods. See for instance [14, 24, 42]. In this work we consider a new hybrid level set and volume-of-fluid that builds on the weak variational form of phase conservation proposed in [25], the optimal control formulation of [6], and the recent reformulation and extension of these approaches to a monolithic scheme in [39]. These methods simultaneously maintain the convenient and precise signed distance representation of the phase geometry and a phase conservation property.

Despite the numerical convenience of the signed distance field, it can be difficult for model users to provide initial conditions, since it generally requires solution of the nonlinear Eikonal equation from some other description of the interface. In this work we provide a method that only requires input of the initial phase indicator field, which we pre-process to obtain the SDF. To do this, we solve the nonlinear Eikonal equation with a stabilization term that blends ideas from [32], the elliptic redistancing by [6] and [39]. In addition, we modify the method in [39] by simplifying the numerical discretization and solving (at every time step) a redistancing pre-stage based on an extension of the stabilized Hamilton-Jacobi formulation of redistancing from [25], which ensures convergence of the elliptic redistancing approach to the correct SDF.

The velocity field is obtained via the well-established second order projection scheme for the incompressible Navier-Stokes equations with variable material parameters by [18]. We use Taylor-Hood finite elements. Additionally, we incorporate artificial viscosity based on [1, 5, 17, 29] and surface tension as proposed in [22]. The artificial viscosity has no tunable parameters and is algebraic, which makes it robust and suitable for unstructured meshes and arbitrary refinement levels.

The rest of this work is organized as follows. In §2 we describe the finite element spatial discretization. In §3, we describe the overall algorithm. In §4, we review the conservative level set method by [39]. We consider only the continuous model and propose a simpler discretization in time. Furthermore, we propose a redistancing stage that can be used to obtain the SDF from a discontinuous indicator field. In §5 we describe the full Navier-Stokes discretization. In this section we propose an algebraic and robust stabilization for the momentum equations. Section 6 is devoted to the numerical examples. Finally, we close with some discussion in §7.

2 Finite element spatial discretization

We use continuous Galerkin finite elements to discretize all equations in space. Let \( \mathbb{R}^d \) be the number of spatial dimensions and \( \Omega \subset \mathbb{R}^d \) be a bounded domain with boundary \( \partial \Omega \subset \mathbb{R}^{d-1} \) which we decompose into \( \partial \Omega^- = \{ x \in \partial \Omega \mid u \cdot n < 0 \} \) and \( \partial \Omega^+ = \{ x \in \partial \Omega \mid u \cdot n \geq 0 \} \). Given an SDF \( \phi \) we define \( \Gamma(t) = \{ x \in \Omega \mid \phi(x, t) = 0 \} \) to be the material interface. Time-dependent variables are defined on the time interval \( t \in [0, T] \), where \( T > 0 \). Given a computational mesh \( T_h \), we consider the finite element space \( X_h^p = \{ w \in C^0(\Omega) \mid w|_K \in \mathbb{P}_p, \forall K \in T_h \} \) with \( p \in \{ 1, 2 \} \); that is, we use only continuous piecewise linear and quadratic spaces. The spaces are spanned by basis functions \( \{ w_1, \ldots, w_{\text{dim}(X_h^p)} \} \) which possess the partition of unity property; namely, \( \sum_j w_j(x) = 1 \). The degrees of freedom associated with these basis functions are denoted by uppercase letters. The finite element solution \( u_h(x) \in X_h \) is given by \( u_h(x) = \sum_{j \in I(h)} U_j w_j(x) \), where \( I_h \) is the patch of elements containing node \( i \). Here, and in the rest of this paper, the notation \( I(z) \) is used for the index set containing the numbers of all basis functions whose support on \( z \) is of nonzero measure.
3 Two-phase flow algorithm based on operator splitting

The overall two-phase flow algorithm is driven by a simple operator splitting technique, as in [25, 31, 41]. In particular, operator splitting is used to decouple the level set and the Navier-Stokes stages. More sophisticated alternatives are possible as well, see for instance [28, 34, 36]. Following this approach one can discretize first the level set equation and subsequently the Navier-Stokes equations or vice versa; we choose the former, which is analogous to the splitting presented for variable-density flows in [18]. In the rest of this section, for simplicity of exposition, we assume a fixed time step; however, we apply this algorithm to a variable time stepping scheme, see §5.1. Let φ, u and p denote the level set, velocity and pressure fields respectively. For any given time step assume we know the solution at time \( t^n \) and \( t^{n-1} \); in particular, that we know \( \phi^n, u^n, u^{n-1}, p^n \) and \( p^{n-1} \). We proceed as follows:

1. Obtain a second order extrapolation of the velocity field: \( u^* = 2u^n - u^{n-1} \approx u^{n+1} \).
2. Using \( u^* \), solve the level set equation described in §4, to obtain \( \phi^{n+1} \).
3. Given the level set solution at \( t^{n+1} \), define the fluid density \( \rho^{n+1} \) and dynamic viscosity \( \mu^{n+1} \) via

\[
\rho^{n+1} = \rho_A H_e(\phi^{n+1}) + \rho_W[1 - H_e(\phi^{n+1})],
\mu^{n+1} = \mu_A H_e(\phi^{n+1}) + \mu_W[1 - H_e(\phi^{n+1})],
\]

where \( A \) and \( W \) denote the air and water phases respectively, and \( H_e \) is a regularized Heaviside function defined in §4.
4. Given \( \rho^{n+1} \) and \( \mu^{n+1} \), solve the variable-coefficient Navier-Stokes equations via a projection scheme, see §5.1, to obtain an updated velocity field \( u^{n+1} \).
5. Repeat until the final time is reached.

4 Monolithic conservative level set method

First note that a weak formulation of phase volume conservation for phase \( \alpha \), with phase indicator function \( \chi_\alpha \), and boundary described by the zero level set of an SDF \( \phi \), can be written as

\[
\int_{\Omega^*} (H(\phi) - \chi_\alpha) w dx = 0 \quad \forall w \in W(\Omega^*)
\]

where \( W(\Omega^*) \) is any suitable test space containing the constant function on subsets \( \Omega^* \) of \( \Omega \). For example, global conservation is ensured if \( \Omega^* = \Omega \) or local conservation if \( \Omega^* = K \). Equation (1) is ill-posed as an equation for \( \phi \), but, if \( W(\Omega^*) \) has the partition of unity property, then the conservation property is maintained even after addition of some regularization \( G \) [25]:

\[
\int_{\Omega^*} [(H(\phi) - \chi_\alpha) w + G \nabla w] dx = 0 \quad \forall w \in W(\Omega^*)
\]

In this work we consider a time-dependent form of equation (2) from [39] for solenoidal velocity field \( v \), which is given in strong form by

\[
\partial_t S_\epsilon(\phi) + \nabla \cdot \left[ v S_\epsilon(\phi) - \lambda (\nabla \phi - q) \right] = 0, \quad \forall x \in \Omega
\]

(3a)

\[
\sqrt{|\nabla \phi|^2 + \delta^2} q = \nabla \phi, \quad \forall x \in \Omega,
\]

(3b)

\[
S_\epsilon(\phi) = S_\epsilon(\phi^{BC}), \quad \forall x \in \partial \Omega^-,
\]

(3c)

\[
(\nabla \phi - q) \cdot n = 0, \quad \forall x \in \partial \Omega
\]

(3d)

where \( \phi \in \mathbb{R} \) denotes the SDF level set function, \( S_\epsilon \) is a smoothed sign function, \( \lambda > 0 \) is a user defined parameter and \( \delta > 0 \) is a small regularization parameter. We use \( \delta = 1 \times 10^{-10} \) in all simulations. The smoothed sign function makes conservation symmetric with respect to the phases and the regularization is based on an elliptic form of the Eikonal equation[6]. Equation (3) combines level set evolution, signed
particular inflow boundary. The second boundary condition \( \alpha \) not observe much sensitivity when \( h \)

Let \( h(x) \) denote a characterization of the local mesh size, then \( \epsilon = \alpha h(x) \) defines the thickness of the regularization of \( H_\epsilon \) and \( S_\epsilon \). We use \( \alpha = \frac{3}{4} \) in all simulations.

Equation (3a) is a conservation law for \( S_\epsilon(\phi) \). Under appropriate boundary conditions and a conservative finite element method (e.g., the partition of unity property), this implies that

\[
\partial_t \int_\Omega S_\epsilon(\phi(x, t)) \, dx = 0 \implies \partial_t \int_\Omega H_\epsilon(\phi(x, t)) \, dx = 0,
\]

and since \( \int_\Omega H_\epsilon(\phi) \, dx \) represents the (regularized) volume of one of the phases, the method is volume conservative. The terms \( \partial_t S_\epsilon(\phi) + \nabla \cdot v S_\epsilon(\phi) = 0 \) correspond to a volume of fluid like model, which imposes not only conservation, but also advects the interface to the correct position. This is true since the velocity field is assumed to be solenoidal (\( \nabla \cdot v = 0 \)) and \( \partial_t S_\epsilon(\phi) + v \cdot \nabla S_\epsilon(\phi) = 0 \implies \partial_t \phi + v \cdot \nabla \phi = 0 \). Therefore, these terms are important for consistency with the volume of fluid and level set methods. Henceforth, we refer to them as consistency terms of (3a). Since the Jacobian of \( \partial_t S_\epsilon(\phi) + v \cdot \nabla S_\epsilon(\phi) = 0 \) is singular, the terms \( -\nabla \cdot \lambda(\nabla \phi - q) \) are added to the model; i.e., they act as a regularization. Note that \( q \equiv \frac{\nabla \phi}{|\nabla \phi|}; \) hence, the regularization terms also penalize deviations of the level set from the distance function. The result is a model for the SDF level set that is volume conservative and contains a term that penalizes deviations from the distance function.

**Remark 4.0.1** (About the regularizing parameter \( \epsilon \)). In general, the level set model may be sensitive to \( \epsilon = \alpha h(x) \). Since this parameter determines the fictitious transition between air and water, it is desirable to keep it as small as possible. In all the experiments in this work we use \( \alpha = 3/2 \). In general, we do not observe much sensitivity when \( \alpha \) is between (around) 1 and 3. Making \( \alpha \) smaller generally makes the nonlinear problem less robust.

**Remark 4.0.2** (About the divergence free constraint on the velocity field). Since we solve the level set model (3) via (continuous) finite elements (see §4.2.3), we only need the velocity field to be weakly divergence free. This is important to remark considering that the velocity field that we obtain is only weakly divergence free, as explained in remark 5.4.1.

**Remark 4.0.3** (Boundary conditions). Note that technically one has to impose boundary conditions for the level set \( \phi \) in the inflow boundary \( \partial \Omega^- \). However, this information is only applied through the smoothed sign function \( S_\epsilon \). Therefore, all we need to know is \( S_\epsilon(\phi^{BC}) \); i.e., the phase (e.g., water or air) at that particular inflow boundary. The second boundary condition \( (\nabla \phi - q) \cdot n = 0 \), \( \forall x \in \partial \Omega \), is imposed to guarantee conservation. Since the regularization term is integrated by parts during the finite element spatial discretization, applying this boundary condition is trivial.

**Remark 4.0.4** (About the parameter \( \lambda \)). The parameter \( \lambda \) controls the amount of regularization and penalization introduced to the model. For any given problem, it is important to guarantee consistency of (3a) w.r.t. the volume of fluid equation. Therefore, the authors in [39] propose to scale \( \lambda \) by \( h(x) \). We follow their definition and use

\[
\lambda = \tilde{\lambda} \left( \frac{v^{\text{max}}}{C} \right) \frac{h(x)}{|\phi| - |\phi||L^\infty(\Omega)},
\]

where \( \tilde{\lambda} = O(1) \) is a dimensionless user defined parameter, \( v^{\text{max}} = \max_x |v| \), \( C < 1 \) is the Courant number and \( |\phi| = \frac{1}{\|\|} \int_\Omega |\phi| \, dx \).

We identify this parameter as the most important in the overall methodology. In all the problems in §6 we obtain satisfactory results using \( \tilde{\lambda} = 10 \). However, it is essential to realize that the size of \( |\phi| - |\phi||L^\infty(\Omega) \)
changes depending on the domain. In particular, if the domain allows the maximum value of $|\phi|$ to grow from one problem to another, then the effective value of $\lambda$ is reduced. If $\lambda$ is too small we observe perturbations in the free surface. And if $\lambda$ is too large more dissipation is added. In general, for any given problem, we want to choose $\lambda$ to be as small as possible without introducing large perturbations to the free surface. In [39], the authors propose to use automated control theory to adjust this parameter at every time step. See for instance [6].

4.1 Discontinuous initialization of the distance function level set

The conservative level set model (3) requires an initial condition $\phi(x, t = 0)$ to be given by an SDF. From a practical point of view this might be inconvenient in some situations. Instead, we propose to start the algorithm by considering an initial configuration given by a discontinuous function $\hat{\phi}\in\{-1,0,1\}$ representing the two phases and re-distance it to obtain an SDF. Due to the fact that the underlying equations for the SDF are nonlinear and the proposed initial conditions are far from the root, we use a robust multi-stage approach to ensure convergence and accuracy. This somewhat complex approach is used only at initialization and requires no user input beyond the initial phase configuration. The main objective is a practical and robust SDF calculation on unstructured meshes. See figure 1a where we consider $\Omega = (0,1)^2$ and

$$\hat{\phi} = \begin{cases} 
1, & \text{if } \text{dist}(x,x_0) < R, \\
0, & \text{if } \text{dist}(x,x_0) = 0, \\
-1, & \text{if } \text{dist}(x,x_0) > R,
\end{cases}$$

where $x_0 = (0.5,0.5)$ and $R = 0.25$. First we project $\hat{\phi}$ onto the finite element space. Afterwards, the main algorithm is based on solving the Eikonal equation several times. The first time we re-distance the solution away from the interface. Then we concentrate on the cells containing the interface. And finally we perform a global redistancing to polish the result. We now provide details about this process.

Step 1: Lumped $L^2$-projection. We start by doing a lumped $L^2$-projection to obtain $\hat{\phi}_h = \sum_j \hat{\phi}_j \omega_j(x)$ where, under defining $\omega_i = \int_{\Omega} w_i dx$, $\hat{\phi}_i = \frac{\omega_i}{m_i} \int \hat{\phi} \omega_i dx$. Here $h$ is a characteristic element size; e.g., $h_\epsilon = \frac{1}{\pi} \left[ \max_{x \in \Omega} h(x) + \min_{x \in \Omega} h(x) \right]$. Note that due to the scaling by $h_\epsilon$, $\hat{\phi}_h$ has units of distance. This step is important to introduce some dissipation into the initial condition. In figure 1c we show $\hat{\phi}_h$.

Step 2: Redistancing away from the interface. Given $\hat{\phi}_h \in X_h^1$, we solve the (viscous) Eikonal equation away from the interface to obtain $\phi_h^* \in X_h^1$ as follows:

$$\int_{\Omega} \epsilon \left| \nabla \phi_h^* \right| - 1 \right| w(x) dx + c \int_{\Omega} h(x) \nabla \phi_h^* \cdot \nabla w(x) dx + \int_{\Omega} \tau \left[ \nabla \phi_h^* - q_h(\phi_h^*) \right] \cdot \nabla w(x) dx = 0, \ \forall w(x) \in X_h^1,$$

(5)

During this process, we freeze the DOFs associated with the interface $\tilde{\Gamma} = \{ x \in \Omega | \hat{\phi}_h = 0 \}$ by imposing strongly $\Phi^*_i = \Phi_i, \ \forall i \in I(\tilde{\Gamma})$. The first term in (5) is the consistency term w.r.t. the Eikonal equation. In the second term $c = O(1)$, we use $c = 0.1$, controls the amount of background dissipation. We introduce this dissipation to aim the solution to converge to the viscous solution of the Eikonal equation. The third term acts as nonlinear stabilization and penalization from the distance function, see remark 4.1.1. In §4.2.1 we provide details about the discretization of $q_h(\cdot)$. In figure 1d we show the result of this step.

Step 3: Redistancing close to the interface. Now we re-distance the solution close to the interface. Given $\phi_h^* \in X_h^1$, we solve the Eikonal equation to obtain $\phi_h^{**} \in X_h^1$ as follows:

$$\int_{\Omega} \epsilon \left| \nabla \phi_h^{**} \right| - 1 \right| w(x) dx + c \int_{\Omega} h(x) \nabla \phi_h^{**} \cdot \nabla w(x) dx + \int_{\Omega} \tau \left[ \nabla \phi_h^{**} - q_h(\phi_h^{**}) \right] \cdot \nabla w(x) dx = 0, \ \forall w(x) \in X_h^1,$$

(6)

During this process, we impose strongly $\Phi^*_i = \Phi_i, \ \forall i \in I(\Omega) \setminus I(\tilde{\Gamma})$; i.e., we allow changes only on the DOFs associated to the interface. By doing this, we aim to limit drastic movement of the interface. We perform only one Newton iteration of this step. In figure 1e we plot $\phi_h^{**}$.

Step 4: Global redistancing. Finally, we obtain $\phi_h^0 \in X_h^1$ by solving

$$a m_i \Delta^\ast^\ast(\Phi_i^0 - \Phi_i^{**}) + \int_{\Omega} \epsilon \left| \nabla \phi_h^0 \right| - 1 \right| w(x) dx + c \int_{\Omega} h(x) \nabla \phi_h^0 \cdot \nabla w(x) dx + \int_{\Omega} \tau \left[ \nabla \phi_h^0 - q_h(\phi_h^0) \right] \cdot \nabla w(x) dx = 0, \ \forall w(x) \in X_h^1,$$

(7)
The first term, inspired by [6], is a penalization term to prevent large changes in the interface. Here \( \alpha \in \mathbb{R} \) is a penalization constant, we use \( \alpha = 10^9 \), and \( m^{\Gamma_{ii}^*} = \int_\Omega \delta_i (\phi^*_h) w_i \, dx \approx \int_{\Gamma_{ii}^*} w_i \, ds \), where \( \Gamma_{ii}^* = \{ x \in \Omega \mid \phi^*_h = 0 \} \). We perform only one Newton iteration of this step. We show \( \phi^0_h \) in figure 1f.

To solve equations (5)-(7) we use a quasi-Newton method. Here we show only the approximate Jacobian of equation (7). The approximate Jacobian at the \( k \)-th Newton iteration is given by

\[
J^{k}_{ij} := \alpha m^{\Gamma_{ii}^*} \delta_{ij} + \int_\Omega S_\epsilon (\phi^*_h) \frac{\nabla \phi^*_h \cdot \nabla w_j}{|\nabla \phi^*_h|} + 10^{-10} w_i \, dx + \int_\Omega [ch(x) + \tau] \nabla w_i \cdot \nabla w_j \, dx,
\]

where \( \delta_{ij} \) is the Kronecker delta function. Note that, for simplicity, we drop the part related to \( q_h(\phi^0_h) \).

**Remark 4.1.1** (About the stabilization term). The last term in equations (5)-(7) acts as stabilization. Indeed, \( \nabla \phi - q(\phi) \approx \left( 1 - \frac{1}{|\nabla \phi|} \right) \nabla \phi \). Therefore, this term behaves like the weak discretization of a Laplacian term with coefficient given by the residual of the Eikonal equation. This idea is similar to the high-order stabilization proposed in [32, §5] and the nonlinear residual-based variational multiscale stabilization used in [25]. The parameter \( \tau \) controls the strength of the stabilization. We use \( \tau = \frac{h(x)}{2} |V| \) with \( |V| = 1 \) which is reasonable since \( V = S_\epsilon (\phi) \frac{\nabla \phi}{|\nabla \phi|} \) is the redistancing velocity. Note that this stabilization term is similar to the regularization and penalization terms in (3a). Moreover, the terms in (3a) can be interpreted as nonlinear stabilization where the coefficient is given not by the residual of the level set equation but by the residual of the Eikonal equation. See [39, Remark 3.2.2].

**Remark 4.1.2** (About the sign of the stabilization term). Note that \( \left( 1 - \frac{1}{|\nabla \phi|} \right) \) < 0 if \( |\nabla \phi| < 1 \). In this situation anti-diffusion is applied which helps in the convergence to \( |\nabla \phi| = 1 \). This is similar to parabolic redistancing by [10] and elliptic redistancing by [6]. Moreover, equations (5)-(7) can be seen as a blend between hyperbolic and elliptic redistancing. By choosing \( \tau = O(h) \) we favor hyperbolic redistancing.

We consider again \( \hat{\phi} \) to be given by (4) and perform a convergence test. The results are shown in table 1.

![Figure 1: Redistancing from a discontinuous configuration. We show the (a) initial state, (b) the elements containing the interface and (c)-(f) the different steps of this process. In all figures we plot the interface in solid white. The mesh is uniform with mesh size given by \( h = \frac{1}{160} \).](image)
| $h$       | N-DOFs | $||\phi_h^n - \phi^{\text{exact}}||_{L^2(\Omega)}$ | Rate |
|----------|--------|---------------------------------|------|
| 2.50E-2  | 1,681  | 6.75E-3                         | –    |
| 1.25E-2  | 6,561  | 2.68E-3                         | 1.33 |
| 6.25E-3  | 25,921 | 1.15E-3                         | 1.22 |
| 3.12E-3  | 103,041| 5.88E-4                         | 0.96 |
| 1.56E-3  | 410,881| 3.05E-4                         | 0.94 |
| 7.81E-4  | 1,640,961| 1.28E-4                      | 1.24 |

Table 1: Convergence of redistancing from discontinuous configuration.

### 4.2 Full discretization of the conservative level set method

#### 4.2.1 $C^0$ normal reconstruction

The vector field $\mathbf{q}_h(\phi_h)$ in (3b) approximates a normal field to the interface. We follow [39] and consider a weighted lumped $L^2$-projection given by

$$\mathbf{q}_h(\phi_h) = \sum_j \mathbf{Q}_j(\phi_h)w_j(x),$$

(8a)

where the components of $\mathbf{Q}_j \in \mathbb{R}^d$, $j \in \mathcal{I}(\Omega)$ are calculated using

$$Q^{(k)}_i(\phi_h) = \frac{\int_{\Omega} \partial_k \phi_h w_i \, dx}{\int_{\Omega} \sqrt{|\nabla \phi_h|^2 + \delta^2 w_i^2} \, dx}.$$  

(8b)

#### 4.2.2 Redistancing pre-stage

We begin each time step with a pre-redistancing stage using the algorithm in §4.1. The motivation behind this step is to introduce some hyperbolicity into the redistancing process. Our aim is to prompt the redistancing at time $t^n$ we find $\phi_h^n \in X_h^1$ such that

$$\alpha m_{t}^{\Gamma^n}(\Phi^{n}_t - \Phi^{n}_i) + \int_{\Omega} S_e(\phi_h^n) [\nabla \phi_h^n | -1] w(x)dx + \int_{\Omega} \tau [\nabla \phi_h^n - \mathbf{q}_h(\phi_h^n)] \cdot \nabla w(x)dx = 0, \hspace{0.5cm} \forall w(x) \in X_h^1,$$  

(9)

where $m_{t}^{\Gamma^n} = \int_{\Gamma^n} \delta_i(\phi_h^n)w_i \, dx$, $\alpha = 10^9$ and $\tau$ is given as in remark 4.1.1. To freeze the interface $\Gamma^n = \{x \in \Omega \mid \phi_h^n = 0\}$, we can set $\alpha = 0$ and impose strongly $\Phi^{n}_i = \Phi^{n}_i$, $\forall i \in \mathcal{I}(\Gamma^n)$. In this case, contrary to §4.1, we keep $\mathbf{q}_h$ independent of the solution $\phi_h^n$. This is done to avoid extra computational effort during this pre-stage.

#### 4.2.3 Discretization of the level set equation

In [39] it is noted that one can use linear continuous Galerkin finite elements with no extra stabilization provided that the advection term is treated implicitly. We follow this idea and solve model (3a) via

$$R(\phi_h^{n+1}, w) := \int_{\Omega} \frac{S_e(\phi_h^{n+1}) - S_e(\phi_h^n)}{\Delta t} \mathbf{w}dx - \frac{1}{2} \int_{\Omega} \left[ S_e(\phi_h^n) \mathbf{v}^n + S_e(\phi_h^{n+1}) \mathbf{v}^{n+1} \right] \cdot \nabla wdx$$

$$+ \int_{\Omega} \lambda [\nabla \phi_h^{n+1} - \mathbf{q}_h(\phi_h^n)] \cdot \nabla wdx + \frac{1}{2} \int_{\partial \Omega} \left[ S_e(\phi_h^n) \mathbf{v}^n + S_e(\phi_h^{n+1}) \mathbf{v}^{n+1} \right] \cdot \mathbf{n} dw x = 0, \hspace{0.5cm} \forall w \in X_h^1,$$  

(10)

where $\phi_h^n$ is the result of the pre-stage given by (9). Note that we discretize the consistency terms in (3a) via a second order Crank-Nicolson discretization in time and use a first-order, implicit-explicit discretization for the regularization and penalization terms respectively. This approach keeps the method simple and efficient (compared to the two stage method proposed in [39]). Indeed, by doing a linearization of (3a) around the interface, the regularization and penalization terms are expected to be $O(h^2)$, see [39, Remark 3.2.2]. Therefore, one can expect no harm from being permissive with respect to the order of approximation of such
terms. We solve (10) via Newton’s method. The Jacobian corresponding to \( R(\phi_h^{n+1}, w_i) \) at the \( k \)-th Newton iteration is given by

\[
\frac{\partial R(\phi_h^k, w_i)}{\partial \phi} = J_h^k = \int_\Omega \nabla \phi_h^k \left( \frac{1}{\Delta t} w_i w_j - \frac{1}{2} \nabla w_i \cdot (v^{n+1} w_j) \right) dx + \int_\Omega \lambda \nabla w_i \cdot \nabla w_j dx + \frac{1}{2} \int_{\partial \Omega} \nabla \phi_h^k w_i w_j v^{n+1} \cdot n ds.
\]

**Remark 4.2.1** (About the mass conservation property). The discrete model (10) is conservative in a regularized sense; i.e.,

\[
\int_\Omega H_\epsilon(\phi^{n+1}) dx = \int_\Omega H_\epsilon(\phi^n) dx = \cdots = \int_\Omega H_\epsilon(\phi^0) dx,
\]

up to the tolerance of the nonlinear solver. One could conserve the exact mass by considering (and integrating) sharp Heaviside functions in (10). To do this we would need to employ more advanced quadrature techniques and/or cut cell methodologies, see for example [13, 45, 46]. We do not pursue this approach in this work.

We close this section by solving a benchmark in the literature of level sets. The problem is known as periodic vortex, see [40]. The domain is given by \( \Omega = (0, 1)^2 \). The initial condition and velocity field are given by

\[
\phi(x,0) = \pm \text{dist}(x, \Gamma_0),
\]

\[
v(x,y,t) = \left[ -\sin(\pi x)^2 \sin(2\pi y) \sin(2\pi t/8) \right],
\]

where \( \Gamma_0 := \{(x,y) \in \Omega \mid (x-x_c)^2 + (y-y_c)^2 = r^2\} \) is a circle of radius \( r = 0.15 \) centered at \( (x_c, y_c) = (0.5, 0.75) \). We select the positive distance in (11a) if \( x = (x,y) \) is inside the circle \( \Gamma_0 \) and the negative distance otherwise. In figure 2 we show the solution at different times, and in table 2 we show the errors and convergence rates in the \( L^1 \)- and \( L^2 \)-norms for different refinements. Additionally, we compute some of the metrics in [39, §5] given by

\[
I_{\text{err}}(\phi_h) = \frac{1}{L} \| H_\epsilon(\phi_h(x)) - H_\epsilon(\phi_h(x, t^{n+1})) \|_{L^1(\Omega)},
\]

\[
V_{\text{err}}(\phi_h) = \frac{1}{\int_\Omega H(\phi_h(x,0)) dx} \left( \int_\Omega [H(\phi_h(x,0)) - H(\phi_h(x,t))] dx \right),
\]

\[
V_{\epsilon,\text{err}}(\phi_h) = \frac{1}{\int_\Omega H(\phi_h(x,0)) dx} \left( \int_\Omega [H_\epsilon(\phi_h(x,0)) - H_\epsilon(\phi_h(x,t))] dx \right),
\]

\[
D_{\text{err}}(\phi_h) = \frac{1}{2} \int_\Omega (|\nabla \phi_h| - 1)^2 dx,
\]

where \( L \) is the \((d-1)\)-dimensional measure of the zero level set \( \Gamma(0) \), and \( H(\cdot) \) is the sharp Heaviside function. The quantities \( I_{\text{err}}(\phi_h) \) and \( V_{\text{err}}(\phi_h) \) measure the extent of interface displacements and area/volume difference with respect to the exact SDF and exact Heaviside function, respectively; see [15, 37], while \( D_{\text{err}}(\phi_h) \) measures the deviation of \( \phi_h \) from a distance function. We present the results in table 3 for five levels of refinement.

<table>
<thead>
<tr>
<th>h</th>
<th>N-DOFs</th>
<th>|\phi_h - \phi^{exact}|_{L^1(\Omega)}</th>
<th>Rate</th>
<th>|\phi_h - \phi^{exact}|_{L^2(\Omega)}</th>
<th>Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.50E-2</td>
<td>1,681</td>
<td>5.77E-2</td>
<td>-</td>
<td>7.52E-2</td>
<td>-</td>
</tr>
<tr>
<td>1.25E-2</td>
<td>6,561</td>
<td>1.87E-2</td>
<td>1.62</td>
<td>2.10E-2</td>
<td>1.84</td>
</tr>
<tr>
<td>6.25E-3</td>
<td>25,921</td>
<td>4.51E-3</td>
<td>2.05</td>
<td>4.78E-3</td>
<td>2.13</td>
</tr>
<tr>
<td>3.12E-3</td>
<td>103,041</td>
<td>1.33E-3</td>
<td>1.75</td>
<td>1.40E-3</td>
<td>1.77</td>
</tr>
<tr>
<td>1.56E-3</td>
<td>410,881</td>
<td>3.82E-4</td>
<td>1.80</td>
<td>3.53E-4</td>
<td>1.98</td>
</tr>
</tbody>
</table>

Table 2: Convergence of two dimensional periodic vortex problem.
Figure 2: Two dimensional periodic vortex problem. We show the solution \( \phi \) at \( t = 0, 2, 4, 6, \) and 8. In all figures we plot the zero contour plot in solid white. The mesh, partially displayed in the lower right corner, is composed of structured triangles with mesh size given by \( h = \frac{1}{160} = 6.25 \times 10^{-3} \).

<table>
<thead>
<tr>
<th>( h )</th>
<th>N-DOFs</th>
<th>( I_{\text{err}} )</th>
<th>( V_{\text{err}} )</th>
<th>( V_{\epsilon, \text{err}} )</th>
<th>( D_{\text{err}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.50E-2</td>
<td>1,681</td>
<td>6.08E-2</td>
<td>2.26E-2</td>
<td>2.30E-11</td>
<td>2.48E-3</td>
</tr>
<tr>
<td>1.25E-2</td>
<td>6,561</td>
<td>1.82E-2</td>
<td>7.68E-3</td>
<td>6.42E-12</td>
<td>4.10E-4</td>
</tr>
<tr>
<td>6.25E-3</td>
<td>25,921</td>
<td>4.51E-3</td>
<td>3.76E-3</td>
<td>2.07E-12</td>
<td>7.12E-5</td>
</tr>
<tr>
<td>3.12E-3</td>
<td>103,041</td>
<td>1.30E-3</td>
<td>1.39E-3</td>
<td>3.56E-12</td>
<td>3.95E-5</td>
</tr>
<tr>
<td>1.56E-3</td>
<td>410,887</td>
<td>3.60E-4</td>
<td>1.74E-3</td>
<td>1.22E-12</td>
<td>8.42E-6</td>
</tr>
</tbody>
</table>

Table 3: Error metrics (12) for the two dimensional periodic vortex problem. The initial length of the interface is \( L \approx 9.42 \times 10^{-1} \).

5 Incompressible Navier-Stokes solver

The incompressible, non-conservative form of the Navier-Stokes equations with variable material parameters is given by

\[
\begin{align*}
\rho (\partial_t u + (u \cdot \nabla)u) - \nabla \cdot (2\mu \varepsilon(u)) + \nabla p &= f, \quad \forall x \in \Omega \quad (13a) \\
\nabla \cdot u &= 0, \quad \forall x \in \Omega \quad (13b)
\end{align*}
\]

where \( \varepsilon(u) = \frac{1}{2} (\nabla u + \nabla u^T) \) is the symmetric gradient, \( \rho \) and \( \mu \) are the density and viscosity respectively and \( u, p \) and \( f \) are the velocity, pressure and force fields respectively.

5.1 Time discretization via a projection scheme

We consider the second order projection scheme for the incompressible Navier-Stokes equations with variable density by [18] and adapt it for variable time step sizes. Let \( r := \frac{\Delta t^n}{\Delta t^m} \). Upon defining

\[
\begin{align*}
\text{BDF2}(u^{n+1}) &:= \frac{(1 + 2r)u^{n+1} - (1 + r)^2u^n + r^2u^{n-1}}{(1 + r)\Delta t^n}, \quad (14a) \\
u^* &:= (1 + r)u^n - ru^{n-1}, \quad (14b) \\
p^* &:= (1 + r)p^n - rp^{n-1}, \quad (14c)
\end{align*}
\]

the projection method is given as follows:

\[
\begin{align*}
\rho^{n+1} \left[ \text{BDF2}(u^{n+1}) + (u^* \cdot \nabla)u^{n+1} \right] - \nabla \cdot (2\mu^{n+1}\varepsilon(u^{n+1})) + \nabla p^* &= f^{n+1}, \quad \forall x \in \Omega, \quad (14d) \\
u &= u^{BC}, \quad \forall x \in \partial\Omega_1, \quad (14e) \\
\varepsilon(u^{n+1}) n &= n_{\partial_n u^{BC}}, \quad \forall x \in \partial\Omega_2, \quad (14f) \\
u \cdot n &= 0, \quad \left[\varepsilon(u^{n+1}) n \right]_\tau = 0, \quad \forall x \in \partial\Omega_3, \quad (14g)
\end{align*}
\]

where \( \partial\Omega_1 \subseteq \partial\Omega \) and \( \partial\Omega_2 \subseteq \partial\Omega \) are sections of the boundary where Dirichlet and Neumann boundary conditions for the velocity field \( u \) are applied, respectively, and \( \partial\Omega_3 \subseteq \partial\Omega \) corresponds to a slip boundary.
Here \((\cdot)\) denotes the tangential component of a given vector field. The pressure is updated via

\[
-\Delta \psi^{n+1} = -\frac{(1 + 2r) \min_x (\rho(x, t = 0))}{(1 + r)\Delta t^n} \nabla \cdot \mathbf{u}^{n+1}, \quad \forall \mathbf{x} \in \Omega, \tag{14h}
\]
\[
\delta \psi^{n+1} = p^{BC} - p^n, \quad \forall \mathbf{x} \in \partial \Omega_4, \tag{14i}
\]
\[
\nabla \delta \psi^{n+1} \cdot \mathbf{n}|_{\partial \Omega} = 0, \quad \forall \mathbf{x} \in \partial \Omega \setminus \partial \Omega_4, \tag{14j}
\]
\[
p^{n+1} = p^n + \delta \psi^{n+1}, \quad \forall \mathbf{x} \in \Omega, \tag{14k}
\]

where \(\partial \Omega_4 \subseteq \partial \Omega\) is the section of the boundary where the Dirichlet boundary condition \(p = p^{BC}\) is applied.

We consider the following initial conditions:

\[
\mathbf{u}(\mathbf{x}, t = 0) = \mathbf{u}_0(\mathbf{x}), \tag{14l}
\]
\[
\delta \psi(\mathbf{x}, t = 0) = 0, \tag{14m}
\]
\[
p(\mathbf{x}, t = 0) = p_0(\mathbf{x}). \tag{14n}
\]

See \(\S 6\) for details about common initial and boundary conditions.

**Remark 5.1.1** (About the linearization of the momentum equations). Note that we follow [18] and use \(\mathbf{u}^\ast\), a second order extrapolation of the velocity field, to linearize the momentum equations. Moreover, we use \(\mathbf{u}^\ast\) in the next section to incorporate artificial viscosity to stabilize the advective term. By doing this, the equation remains linear with respect to the solution at time \(t^{n+1}\).

**Remark 5.1.2** (About accuracy). The expected accuracy of this projection scheme is second order in the \(L^2\)- and \(H^1\)-norms. We refer the reader to [18] for a convergence study.

### 5.2 Artificial viscosity

In this section, we add artificial viscosity to stabilize the advective term in the momentum equations. To do this we consider [1, 5, 17, 29], where artificial dissipative operators are introduced to enforce Discrete Maximum Principles (DMP). We do not have a DMP requirement; instead, we are interested in adding enough artificial viscosity to have a robust and well-behaved discretization. This extra viscosity must vanish as the mesh is refined, must behave properly for unstructured meshes and for all refinements and should not add significant computational cost to the method. In particular, we are interested in keeping equation (14d) linear. Finally, we aim to have no tunable parameters and to preserve, to the best of our ability, the accuracy properties of the underlying method.

**Componentwise smoothness indicator**

We concentrate first on one component of the velocity vector; e.g., in the \(x\)-component \(u_x\), whose degrees of freedom are denoted by \(U\), see \(\S 2\). The first step is to neglect the force, pressure and viscosity terms in (14d); as a result, we get a hyperbolic system, which is more suitable for the theory developed in the references above. In the rest of this section we consider backward Euler time stepping; nevertheless, we apply these results to the second order method in \(\S 5.1\). By doing this the full discretization becomes

\[
M \left( \frac{U^{n+1} - U^n}{\Delta t} \right) + K(\mathbf{u}_h^n)U^{n+1} = 0, \tag{15}
\]

where \(M\) is the mass matrix and \(K\) is the linearized advective matrix, whose components are given by \(M_{ij} = \int_{\Omega} w_i(\mathbf{x})w_j(\mathbf{x})d\mathbf{x}\) and \(K_{ij} = \int_{\Omega} \mathbf{u}_h^\ast \cdot \nabla w_i(\mathbf{x})w_j(\mathbf{x})d\mathbf{x}\) respectively. Now we introduce an artificial dissipative matrix, see e.g. [29] and references therein, whose components are given by

\[
D_{ij}^{\text{Low}} = \begin{cases} 
\max \left[ K_{ij}(\mathbf{u}_h^\ast), \, 0, \, K_{ji}(\mathbf{u}_h^\ast) \right], & \text{if } i \neq j, \\
-\sum_{k \neq i} D_{ik}^{\text{Low}}, & \text{if } i = j.
\end{cases} \tag{16}
\]

The objective behind this idea is to add sufficient conditions to construct \(U^{n+1}\) as a convex combination of \(U^n\). This, along with using a positive lumped mass matrix, guarantees the DMP property. By doing this,
however, the accuracy of the method is reduced to first order. To recover the second order accuracy expected from the projection scheme we use a smoothness based indicator, see [1, 5, 17], given by

$$
\beta_i^x := 1 - \left[ \frac{|\sum_j U_j^x - U_i^x|}{\sum_j |U_j^x - U_i^x| + 10^{-10}} \right]^2. \tag{17}
$$

The indicator $0 \leq \beta_i^x \leq 1$, which is associated with the $x$-component of the velocity field, is (close to) one when the solution is smooth and (close to) zero when the solution is non-smooth and at local extrema.

Isotropic artificial dissipation

The smoothness indicators for the other velocity components are computed similarly. Assume the problem is three dimensional. Then, we define

$$
\beta_i = \min(\beta_i^x, \beta_i^y, \beta_i^z),
$$

which is subsequently used to reduce the amount of artificial dissipation from (16). This can be done in different ways. In this work, we follow [1] and define

$$
D_{ij} := \begin{cases} 
\max\left[ (1 - \beta_i)K_{ij}(u_h^i), 0, (1 - \beta_j)K_{ji}(u_h^j) \right], & \text{if } i \neq j \\
-\sum_{k \neq i} D_{ik}, & \text{if } i = j.
\end{cases}
$$

Finally, we apply this discrete operator to each component of the velocity and reincorporate the pressure, force and viscosity terms. The full discretization for the $x$-th component is given by

$$
M \left( \frac{U^{n+1} - U^n}{\Delta t} \right) + [K(u_h^i) - D + L]U^{n+1} + F^x = 0,
$$

and similarly for the other components of the velocity. Here $L$ is the stiffness matrix and $F^x$ is a vector that accounts for the pressure and force terms. Let $f^x$ and $n^x$ denote the $x$-th component of the force field and the $x$-th component of the normal unit vector to the boundary respectively. Then, the entries of $L$ and $F^x$ are given by

$$
L_{ij} = \int_{\Omega} 2\mu^n \varepsilon(w_j) : \nabla w_i \, dx,
$$

$$
F_i^x = -\int_{\Omega} p_h \partial_x w_i(x) \, dx + \int_{\partial \Omega} p n^x w_i(s) \, ds - \int_{\Omega} f^x w_i(x) \, dx.
$$

Note that we integrate by parts for the pressure term.

**Remark 5.2.1** (About the DMP property). When the smoothness indicator $\beta^x$ is constructed based on $U^{n+1}$, the mass matrix is lumped such that $m_i = \int_{\Omega} w_i \, dx > 0$ and the corresponding $D_{ij}$ matrix is applied to (15), the DMP property is guaranteed, see e.g. [11]. By using $U^*$ instead, the DMP property is likely lost. In addition, the viscous, pressure and force terms require extra care. Also note that, to avoid loss of accuracy, we do not lump the mass matrix. Nevertheless, as mentioned before, we do not aim to satisfy a DMP. Instead, our objective is to add robust and reliable artificial viscosity to dissipate small spatial scales not resolved by the computational mesh in order to obtain a well-behaved method for any mesh and any refinement level while nevertheless converging rapidly.

**Remark 5.2.2** (About accuracy). It is known that the discrete operator $D_{ij}^{\text{Low}}$ decreases the order of convergence to first order even for smooth solutions, see for instance [29]. Second order convergence is recovered except around local extrema when the smoothness indicator $\beta$ is used. The degeneracy of accuracy around local extrema occurs since $\beta = 0$ at local extrema. To overcome this problem, one can consider an extra smoothness sensor based on second derivatives, which is suitable for the velocity space $X_h^2$, see [32]. The objective is to identify smooth local extrema to deactivate the artificial dissipation. In [30] this idea is employed for different scalar equations. By doing this, the authors demonstrate that the high-order accuracy of the underlying method is recovered. We do not explore this idea further in this work.
5.3 Surface tension

In this section, we summarize the work by [22] to incorporate surface tension effects. Given \( \Gamma \subset \Omega \) to be the interface between the two fluids, the goal is to impose \( |u| = 0 \) and \( -pI + 2\mu \varepsilon(u)|_\Gamma \cdot \mathbf{n}^\Gamma = \sigma \kappa \mathbf{n}^\Gamma \), where \( \mathbf{n}^\Gamma \) is the unit normal vector to the interface, \( \sigma \) is the surface tension coefficient and \( \kappa \) is the curvature of the interface. These conditions can be imposed by incorporating

\[
f_{st} = -\sigma \kappa \mathbf{n}^\Gamma|_\Gamma \tag{18}\]

into the left hand side of (13). Let \( \nabla z(x) = \nabla z(x) - [\mathbf{n}^\Gamma \cdot \nabla z(x)] \mathbf{n}^\Gamma \), \( \forall x \in \Gamma \) be the tangential gradient with respect to \( \Gamma \) and \( \Delta z(x) = \nabla \cdot \nabla z(x) \), \( \forall x \in \Gamma \) be the Laplace-Beltrami operator. Using the fact that \( \kappa \mathbf{n}^\Gamma = \Delta x|_\Gamma \), see for instance [16]; the weak form of (18), after multiplying \( f_{st} \) by a test function \( \mathbf{w}(x) \) and integrating by parts, is given by

\[
f_{st} = \int_{\Gamma} \sigma \nabla \mathbf{n}^\Gamma(x) \cdot \nabla \mathbf{w}(x) dS, \tag{19}\]

where \( \mathbf{n}^\Gamma(x) \) is the identity map on \( \Gamma \). The boundary integral vanishes for the cases of interest in this work. Once the force integral related to the surface tension is defined, the authors in [22] propose to treat (19) semi-implicitly based on \( \mathbf{n}^{\Gamma+1} \approx \mathbf{n}^\Gamma + \Delta t^{n+1} \mathbf{n}^{n+1} \) and to approximate the surface integrals by volume integrals via the regularized delta function of the level set. Doing this yields

\[
f_{st}^{n+1} \approx \int_{\Omega} \sigma [\nabla \mathbf{n}^{\Gamma n} \cdot \nabla \mathbf{w}(x)] \delta_{s}(\phi^\Gamma_n) dx + \Delta t^{n+1} \int_{\Omega} \sigma [\nabla \mathbf{u}^{n+1} \cdot \nabla \mathbf{w}(x)] \delta_{s}(\phi^\Gamma_n) dx. \tag{20}\]

Assuming the acute angle condition is satisfied, see for instance [9], it is remarked in [22] that since the second term in (20) is positive, it contributes to the stability properties of the momentum equations. In other words, it adds viscosity to the velocity at the interface \( \Gamma \). See more details about this discretization in the next section.

5.4 Full discretization

Let \( \mathbf{u}_h, \mathbf{w} \in [X^2_h]^d \) and \( p_h, \delta \psi_h, \theta \in X_h^1 \). The full discretization of (14) is given by

\[
\int_{\Omega} \rho^{n+1} \left[ 1 + \frac{(1 + r) \Delta t^n}{1 + 2r} (\mathbf{u}^n_h \cdot \nabla) \right] \mathbf{u}^{n+1}_h \cdot \mathbf{w} \, dx + \frac{(1 + r) \Delta t^n}{1 + 2r} \int_{\Omega} 2\mu^{n+1} \varepsilon(\mathbf{u}^{n+1}_h) : \varepsilon(\mathbf{w}) \, dx + \text{stab.} \tag{21a}\]

\[
+ \frac{(1 + r) \Delta t^n}{1 + 2r} \Delta t^{n+1} \int_{\Omega} \sigma (\nabla \mathbf{u}^{n+1}_h \cdot \nabla \mathbf{w}) \delta_s(\phi^\Gamma_n) dx = \mathbf{b}^{n+1}(w), \]

where ‘stab.’ refers to the artificial viscosity from §5.2 and \( \mathbf{b}^{n+1} \) is given by

\[
\frac{1 + 2r}{(1 + r) \Delta t^n} \mathbf{b}^{n+1}(w) = \int_{\Omega} \rho^{n+1} \left[ (1 + r)^2 \mathbf{u}^n_h - (1 + r) \mathbf{u}^{n-1}_h \right] \cdot \mathbf{w} \, dx \int_{\Omega} p_h^{n} \, \text{div}(\mathbf{w}) \, dx
\]

\[
- \int_{\partial \Omega} p^{n}_h \, \mathbf{n} \cdot \mathbf{w} \, ds - \sigma \int_{\partial \Omega} (\nabla \mathbf{n}^{\Gamma n} \cdot \nabla \mathbf{w}) \delta_s(\phi^\Gamma_n) \, ds + \int_{\partial \Omega} 2\mu^{n+1} \partial_n \mathbf{u}^{BC} \cdot \mathbf{w} \, ds. \tag{21b}\]

Upon defining \( a := \frac{(1+2r) \min(\rho(x,t=0))}{(1+r)\Delta t} \), the full discretization of the pressure update is given by

\[
\int_{\Omega} \nabla \delta \phi^{n+1}_h \cdot \nabla \theta \, dx = a \left( \int_{\Omega} \mathbf{u}^{n+1}_h \cdot \nabla \theta \, dx - \int_{\partial \Omega} (\mathbf{u}^{n+1}_h \cdot \mathbf{n}) \theta \, ds \right) \tag{21c}\]

\[
p^{n+1}_h = p^n_h + \delta \psi^{n+1}_h. \tag{21d}\]

Remark 5.4.1 (Velocity correction). After the pressure update, it is possible to correct the velocity field to obtain a weakly divergence free velocity field. To do this we can rearrange the terms in (21c) to obtain

\[
\int_{\Omega} \left( \mathbf{u}^{n+1}_h - \frac{1}{a} \nabla \delta \psi^{n+1}_h \right) \cdot \nabla \theta \, dx - \int_{\partial \Omega} \mathbf{u}^{n+1}_h \cdot \mathbf{n} \theta \, ds = 0, \]
and redefine the velocity such that \( u_h^{n+1} \rightarrow u_h^{n+1} - \frac{1}{a} \nabla \psi_h^{n+1} \) in the interior of the domain. By doing this it is clear that

\[
\int_{\Omega} u_h^{n+1} \cdot \nabla \theta dx - \int_{\partial \Omega} (u_h^{n+1} \cdot \mathbf{n}) \theta ds = 0 \implies - \int_{\Omega} \nabla \cdot (u_h^{n+1}) \theta dx = 0.
\]  

(22)

It is remarked in [19, §2.5], and references therein, that not doing this correction does not affect the accuracy properties of the method. However, as explained in §4, the level set model assumes the velocity to be (at least weakly) divergence free. Therefore, we correct the velocity field.

### 6 Numerical experiments

A total of eight test problems were selected to demonstrate the behavior of the method we propose. The results were compared qualitatively and quantitatively versus other results in the literature and versus experimental measurements. These two-phase flow experiments represent a wide range of applications ranging from high-viscosity buckling fluids to free surface flows around obstacles. Our main objective is to test the robustness of the method for a wide range of physical parameters, surface tension, different types of boundary conditions, external forces and arbitrary mesh refinements. In particular, we set the numerical parameters to be the same for all problems; this demonstrates the robustness of the numerical method and suitability of the computational model for engineering applications.

#### Parameters

The physical parameters are part of the definition of the problem. They are given by the density (\( \rho \)) and viscosity (\( \mu \)) of each phase, the magnitude of the gravity (\( g \)) and the surface tension coefficient (\( \sigma \)). Unless otherwise stated, we consider the metric system and set

\[
\rho_W = 998.2 \left( \frac{\text{kg}}{\text{m}^3} \right), \quad \rho_A = 1.205 \left( \frac{\text{kg}}{\text{m}^3} \right), \quad \mu_W / \rho_W = 1.004 \times 10^{-6} \left( \frac{\text{m}^2}{\text{s}} \right),
\]

\[
\mu_A / \rho_A = 1.5 \times 10^{-5} \left( \frac{\text{m}^2}{\text{s}} \right), \quad g = 9.8 \left( \frac{\text{m}}{\text{s}^2} \right), \quad \sigma = 72.8 \times 10^{-3} \left( \frac{\text{kg}}{\text{s}^2} \right),
\]

where \( W \) and \( A \) refer to water and air, respectively. In the rest of this work we omit the units. The parameter \( \lambda \) in the conservative level set method is the main numerical parameter, see remark 4.0.4. In all simulations we use \( \lambda = 10 \). A less important parameter is \( \epsilon \), which is used to regularize Heaviside and Dirac delta functions, see §4. During the redistancing process we set \( \epsilon = \frac{1}{4} \) and elsewhere we use \( \epsilon = \frac{3}{4} \). In all problems we let \( h_e \) denote a characteristic element size (of the unstructured mesh) and report the number of elements.

#### Commonly used initial and boundary conditions

In different problems we start ‘at rest’; i.e., the initial velocity and pressure fields are set to zero. Likewise, we use similar and common boundary conditions in different problems. We refer to them as slip, non-slip, open top and inflow boundary conditions. These common boundary conditions for the velocity are shown in table 4, where \( \langle \cdot \rangle_t \) denotes the tangential component of a given vector field. The level set does not require boundary conditions when the boundary is set to slip or non-slip. When the boundary is open and \( \mathbf{u} \cdot \mathbf{n} < 0 \) we set \( S_t(\phi) = 1 \); i.e., we let only air through the open boundary. Finally, when the boundary is of inflow type we set either \( S_t(\phi) = 0 \) or \( S_t(\phi) = 1 \) to let water or air respectively into the domain.

<table>
<thead>
<tr>
<th></th>
<th>Slip</th>
<th>Non-slip</th>
<th>Open top</th>
<th>Inflow</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \mathbf{u} \cdot \mathbf{n} = 0 )</td>
<td>( \epsilon(\mathbf{u}) \cdot \mathbf{n} = 0 )</td>
<td>( \mathbf{u} = 0 )</td>
<td>( \epsilon(\mathbf{u}) \cdot \mathbf{n} = 0 ), ( p = 0 )</td>
<td>( \mathbf{u} = \mathbf{u}^{\text{inlet}}, \text{s.t. } \mathbf{u} \cdot \mathbf{n} &lt; 0 )</td>
</tr>
</tbody>
</table>

Table 4: Commonly used boundary conditions for the velocity.

#### Computational framework

This work was developed using PROTEUS (https://proteustoolkit.org), a toolkit for computational methods and simulation released under the MIT open source license with source available at https://github.com/
erdc/proteus. The numerical linear algebra is handled by PETSc, see [2, 3, 4], through petsc4py [12]. Inside PROTEUS 1.5.1, we created a set of files to facilitate the definition of the different problems. This framework is not part of PROTEUS; nevertheless, to facilitate its use, we provide a release with this framework incorporated, see release 1.5.1-mp-r1 (https://github.com/erdc/proteus/releases/tag/1.5.1-mp-r1).

Our aim is to provide a solid, robust and easy to use open-source computational framework for users interested in two-phase flows. We do not assume the users have any knowledge on finite elements nor expect the need to adjust the numerical parameters, except for potentially λ̂. Instead, we expect the user to provide information only about the problem; in particular, the domain, initial and boundary conditions and physical parameters. We encourage the interested reader to install the PROTEUS release and run different problems.

6.1 Two-dimensional experiments

6.1.1 Rising bubbles

We consider first a bubble of a light fluid at rest immersed in a heavier fluid. Due to the action of gravity, the bubble rises through the heavier fluid. In this problem, the surface tension is critical for maintaining the correct shape of the bubble and obtaining its correct position and speed. We consider the two common test cases in [23]. The domain of interest is Ω = (0, 1) × (0, 2). We impose non-slip boundary conditions on the bottom and top boundaries and slip boundary conditions on the left and right boundaries. The initial condition for level set is initialized as explained in §4.1 considering an interface given by Γ = {(x, t) ∈ Ω | r² = (x − 0.5)² + (y − 0.5)², r = 0.25}. The material parameters for the different test cases are shown in table 5. We consider structured meshes with refinement levels h_e = 1/80, 1/40 and h_e = 1/160, which corresponds to 6480, 25760 and 102720 elements respectively. The zero contour plot of the level set and the evolution of the center of mass c_y = 1/|ΩB| ∫ΩB y dΩ and rise velocity r_v = 1/|ΩB| ∫ΩB u_y dΩ are shown in figure 3. Here ΩB = {(x, y) ∈ Ω | φ(x, t) ≥ 0} denotes the domain occupied by the bubble.

<table>
<thead>
<tr>
<th>Test case</th>
<th>ρ_W</th>
<th>ρ_A</th>
<th>μ_W</th>
<th>μ_A</th>
<th>g</th>
<th>σ</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1000</td>
<td>100</td>
<td>10</td>
<td>1</td>
<td>0.98</td>
<td>24.5</td>
</tr>
<tr>
<td>2</td>
<td>1000</td>
<td>1</td>
<td>10</td>
<td>0.1</td>
<td>0.98</td>
<td>1.96</td>
</tr>
</tbody>
</table>

Table 5: Material parameters for the rising bubble problems. We consider two common test cases.

6.1.2 Dam break problem

Here we consider a common two-dimensional dam break problem. In [35], the authors considered a similar problem and performed a series of experiments placing pressure and water height gauges at different locations. Based on these results, different numerical studies have been performed to validate numerical methods and codes. In this work, we consider the setup by [11] and reproduce some of their results. The domain of interest is Ω = (0, 3.22) × (0, 1.8). An initial column of water at rest is located in W = {x ∈ Ω | x ≤ 1.2, y ≤ 0.6}. The rest of the domain (A = Ω \ W) is filled with air. At time t = 0 the column of water starts to fall because of the action of gravity. For this problem, we follow [11] and consider non-slip boundary conditions everywhere except for the top boundary, which is left open. We report the results of only one pressure gauge located at P_1 = (3.22, 0.12) and two water height gauges located at H_1 = (2.228, y) and H_2 = (2.724, y), ∀y ∈ [0, 1.8]. Let d = 3.22 denote the length of the domain. We consider three unstructured meshes with h_e = 0.01d, 0.00625d and h_e = 0.003125d, which correspond to 17752, 45188 and 181033 triangular elements respectively. In figure 4 we compare the numerical results (for the different refinements) versus the experimental measurements for all the gauges. Finally, in figure 5 we consider the same times as [11, figure 17] and show the water-air interface. Note that only in this figure, to facilitate comparisons, we follow the reference and scale the x-axis as x̂ = x/H and the time as t = t√|g/H|, where H = 0.6 is the initial height of the column of water.

6.1.3 Buckling flow

The fluid buckling phenomenon occurs in situations involving thin streams of highly viscous flow that encounter a boundary or plate. Depending on the Reynolds number and the cross-sectional geometry of the viscous jet, the fluid may fold, coil, or oscillate antiperiodically. Here, the method’s ability to simulate buckling flow is evaluated against results from previous literature, see e.g., [7, 8, 44, 47]. The domain Ω = (0, 1)^2
Figure 3: Rising bubble problems in [23]. We show (in the left panel) the zero contour plot of the level set at \( t = 3 \), (in the middle panel) the evolution of the center of mass and (in the right panel) the evolution of the \( y \)-th component of the velocity. We consider three refinement levels with (black) \( h_e = 1/80 \), (red) \( h_e = 1/160 \) and (green) \( h_e = 1/320 \).

is initially occupied by the air phase, which is at rest. A small inlet on the top boundary of the box, defined by \( I = \{ x \in \bar{\Omega} \mid |x - 0.5| < 0.05 \} \), is the inflow boundary, where the velocity is strongly set to \( u = (0, -1) \). The rest of the top boundary is left open. The non-slip boundary condition is applied in the bottom, left and right boundaries. The material properties are shown in table 6. We use an unstructured mesh with a maximum element size of \( h_e = 5 \times 10^{-3} \), which correspond to 126,646 triangular elements. In figure 6 we plot the heavy fluid phase for different times.

<table>
<thead>
<tr>
<th>( \rho_W )</th>
<th>( \rho_A )</th>
<th>( \mu_W )</th>
<th>( \mu_A )</th>
<th>( g )</th>
<th>( \sigma )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1800</td>
<td>1</td>
<td>500</td>
<td>( 2 \times 10^{-2} )</td>
<td>9.8</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 6: Material parameters for the buckling flow problem.

6.1.4 Filling tank

In this section we consider the problem of filling an empty tank with a water-like fluid. For this problem, the domain of interest is given by \( \Omega = (0, 0.4)^2 \). The initial data consists of water in \( W = \{ (x, y) \in \Omega \mid x < 0.01, \ |y - 0.325| < 0.025 \} \) and air in \( A = \Omega \setminus W \). Both phases start at rest. The boundary \( I = \{ (x, y) \in \Omega \mid x = 0, \ |y - 0.325| < 0.025 \} \) is set to inflow boundary. At \( I \) we set strongly \( u = (0.25, 0) \). The rest of the left boundary is defined as slip boundary, the bottom and the right boundaries are non-slip and the top is left open. We consider two sets of material parameters. First we follow [20, §11.3] and reproduce qualitatively their results. In this case the material parameters are

\[
\rho_W = 1000, \quad \rho_A = 1, \quad \mu_W = 1, \quad \mu_A = 1.8 \times 10^{-2}, \quad g = 1, \quad \sigma = 0.
\]

Then we consider more realistic parameters given by (23a), without surface tension effects; i.e., \( \sigma = 0 \). We use an unstructured mesh with maximum element size of \( h_e = 2 \times 10^{-3} \), which corresponds to 126,888 triangular elements. The water phase at different times are shown in figure 7. Note the difference in the behavior of the solution due to different gravity and viscosity parameters.
6.2 Three-dimensional experiments

6.2.1 Dam break problem with obstacle

The first three-dimensional problem that we consider is a dam break with an obstacle located downstream. The computational domain is \( \Omega = (0, 3.22) \times (0, 1) \times (0, 1) \setminus O_1 \), where \( O_1 = [2.39, 2.55] \times [0.3, 0.7] \times [0, 0.16] \) represents the obstacle. An initial column of water at rest is located in the domain \( W = \{ x \in \Omega \mid z \leq 1.22, \, z \leq 0.55 \} \). The rest of the domain; i.e., \( A = \Omega \setminus W \), is filled with air. At \( t = 0 \) the column of water starts to fall because of the action of gravity. Non-slip boundary conditions are imposed everywhere except for the top, which is left open. Experiments for this problem were performed by the Maritime Research Institute Netherlands (MARIN), see [26, 27]. In particular, pressure and water height gauges were located in different points of interest. In this work we report the results for only four pressure gauges located at \( P_1 = (2.389, 0.526, 0.025), \, P_3 = (2.389, 0.526, 0.099), \, P_5 = (2.414, 0.474, 0.165) \) and \( P_7 = (2.487, 0.474, 0.165) \) and four water height gauges located at \( H_1 = (0.582, 0.5, z), \, H_2 = (1.732, 0.5, z), \, H_3 = (2.228, 0.5, z) \) and \( H_4 = (2.724, 0.5, z) \), \( \forall z \in [0, 1] \). See figure 8. We consider an unstructured mesh with characteristic size \( h_e = 0.025 \), which corresponds to 2,552,090 tetrahedral elements. In figure 9, we compare the numerical results versus the experimental measurements for the pressure and water height gauges and in figure 10 we show the water phase at different times.

6.2.2 Moses flow

In this simulation we consider the domain \( \Omega = (0, 3.22) \times (0, 1) \times (0, 1) \setminus O_1 \cup O_2 \), where \( O_1 = [0.6635, 0.8245] \times [0.2985, 0.7015] \times [0, 0.35] \) and \( O_2 = [2.3955, 2.5565] \times [0.2985, 0.7015] \times [0, 0.35] \) represent two obstacles located at both ends of the domain. Initially, the tank is partially filled with water and air at rest in the domains \( W = \{ x \in \Omega \mid z \leq 0.3 \} \) and \( A = \Omega \setminus W \), respectively. We apply an external force designed to split the water at the center of the tank. Our aim with this problem is to test the robustness of the method with respect to
Figure 6: Two-dimensional buckling flow. From top to bottom and left to right, we show the heavy fluid phase at \( t = 0.3, 0.6, 0.9, 1.2, 1.5, 1.8, 2.1, 2.4, 2.7 \) and 3.

external (and aggressive) forces. The force is

\[
F(x, t) = \begin{cases} 
-7.5\rho[1 + \tanh(t)](10 - 9z), & \text{if } 1.36 \leq x < 1.61, \\
7.5\rho[1 + \tanh(t)](10 - 9z), & \text{if } 1.61 \leq x \leq 1.86.
\end{cases}
\]

Note that the external force is scaled by the fluid density. All boundaries are considered to be non-slip except the top which is left open. The grid is unstructured with element size \( h_e = 0.025 \), which corresponds to 2,522,647 tetrahedral elements. In figure 11 we show the water phase at different times. We remark that the solution appears to tend to a steady state.
Figure 7: Two-dimensional filling of a tank. For both test cases we show the heavy fluid phase at $t = 0.3, 0.6, 0.9, 1.2, 1.5, 1.8, 2.1$ and $2.4$.

Figure 8: Water height and pressure gauges shown in blue and red, respectively.
Figure 9: Water height and pressure gauges. The experimental data is plotted in dashed black and the numerical solution in solid red.

Figure 10: Three-dimensional dam break problem with obstacle. From top to bottom and left to right, we show the water phase at $t = 0, 0.5, 0.7, 0.9, 1.1, 1.4, 4.5, 6.0$ and $7.5$. 
Figure 11: Three-dimensional Moses test. From top to bottom and left to right, we show the water phase at \( t = 0, 0.1, 0.4, 0.6, 0.9, 1.3, 2, 3, \) and 4.
7 Conclusions

In this work we presented a numerical model of incompressible, immiscible two-phase flow. The general algorithm is driven by an operator splitting scheme in which a conservative level set method is first solved for a given velocity field to evolve the interface. Afterwards, the interface (at the new location) is employed to obtain an updated velocity field.

We extended the conservative monolithic level set method presented in [39] to make it more robust and to allow simpler initial conditions. The original monolithic scheme contains a second-order term that penalizes deviations from the distance function, which was derived from parabolic and elliptic redistancing, see [6, 10]. In our experience with two-phase flow simulations using the method in [39], we found it is not always possible to force the redistancing to emanate from the interface, which results in incorrect solutions (of the viscous Eikonal equation [33]). As this was not an issue with the earlier multistage conservative level set method presented in [25] based on the classical Eikonal equation, we propose performing an extra pre-processing step to induce the redistancing to be propagated from the interface, effectively providing a better initial guess to the original monolithic scheme. This extra step is done by solving the Eikonal equation directly. It is important to remark that, thanks to the penalization embedded into the conservative level set, solving the Eikonal equation in our case is not a computationally expensive task. This is true since the initial guess in the nonlinear iterative process already contains most of the features of the solution. We see this pre-processing step as a small but necessary improvement for the initial condition at each time step and consider a more robust monolithic scheme as future work.

We use a second order projection scheme proposed for variable-density incompressible flows in [18] extended to solve the Navier-Stokes equations with variable density, viscosity, and time steps. For the applications of consideration in this work, extra artificial viscosity is commonly needed to stabilize the momentum equations at high Reynolds number. Special care is required when some of these techniques are used with unstructured meshes. In this work, we extend recent ideas on solving hyperbolic equations via Discrete Maximum Principle (DMP) preserving continuous Galerkin finite elements. These methods are algebraic and suitable for both coarse and highly refined unstructured meshes. We relax the DMP preserving features of these schemes, which allows simplification of the stabilization; however, in our numerical experiments we always obtained well-behaved solutions. We remark that this stabilization technique is free of tunable parameters and capable of high-order accuracy with suitable changes to the underlying approximation spaces.

Once the individual methods are described, we present an extensive set of numerical examples in two- and three-dimensions. In all these problems, we use the same numerical parameters to demonstrate the robustness of the method. Finally, we provide an open source and freely downloadable computational framework to reproduce the numerical examples presented in this work. The code uses a simple Python input file, which the interested reader can easily modify to solve new problems.

8 Acknowledgments

The work of Manuel Quezada de Luna was supported primarily by an appointment to the Postgraduate Research Participation Program at the U.S. Army Engineer Research and Development Center, Coastal and Hydraulics Laboratory (ERDC-CHL) administrated by the Oak Ridge Institute for Science and Education through an inter-agency agreement between the U.S. Department of Energy and ERDC. Haydel Collins and Chris Kees were supported by the ERDC University program and the ERDC Future Investment Fund. Permission was granted by the Chief of Engineers, US Army Corps of Engineers, to publish this information.

References


