A volume-of-fluid interfacial flow solver with advected normals

Mehdi Raessi*,1, Javad Mostaghimi and Markus Bussmann
Department of Mechanical and Industrial Engineering, University of Toronto, Canada

Abstract

We presented a new approach to calculating normal vectors to fluid interfaces in [JCP, 226:774–797, 2007], by advecting unit normals along with an interface. In this paper, we introduce an implementation of the method in an interfacial flow solver. The advected normals are used to compute the interface curvature for calculating the surface tension force, and for reconstructing the interface in a volume-conserving volume-of-fluid (VOF) method. To improve the performance of the method in under-resolved regions of the flow, where normals vary sharply, a curvature-based criterion is used to detect and correct poorly defined normals. We present two-dimensional results of advection as well as actual flow problems and demonstrate that the new method is well suited for problems that involve large interface deformation and breakup (i.e. problems that involve substantial interface movement).

Key words:
Curvature, Surface tension, Interfacial flow, Two-phase flow, Spurious currents, Volume-of-fluid, Level set

*Corresponding author
Email address: mraessi@mie.utoronto.ca (Mehdi Raessi)
1Present address: Center for Turbulence Research, Stanford University.
1. Introduction

Interfacial flows are an integral part of many technologies including liquid atomization, boilers, fuel cells, ink-jet printing, spray coating and casting processes. To design and improve such technologies, a thorough understanding of the flow fields is required. Numerical simulation is a useful tool for analyzing such flows, especially as experiments are often expensive and difficult to carry out. Analysis of interfacial flows requires solving for the flow field and predicting the shape of fluid interfaces. Since important characteristics of interfacial flows (e.g. interface breakup patterns) are essentially determined by the evolution of fluid interfaces, accurate modeling of interface kinematics is vital. In particular, it is important that the predicted interface geometries are not numerical artifacts arising from errors in modeling interface evolution.

As a result, accurate calculation of interface quantities such as curvature and normal vectors is important, as these quantities are used to evaluate surface tension. Errors in the calculated surface tension force will induce non-physical velocities, commonly known as spurious or parasitic currents. These velocities can grow with time and dominate important physical effects such as buoyancy, and so significantly degrade simulation results. The accuracy of interface curvature is most critical when surface tension is a dominant force (i.e. for low capillary, Weber, or Bond number flows), as occurs, for example, when the characteristic length scales of a phenomenon are small, as in microscale interfacial flows encountered in microfluidics. The accuracy of normal vectors is important in modeling interface kinematics too; for exam-
ple, in the volume-of-fluid (VOF) method [1, 2] the normal vectors are used to reconstruct the interface and advect fluid volume.

There are two main approaches to modeling interface kinematics [3, 4]: interface tracking (Lagrangian) methods (e.g. front tracking [5] and surface fitted methods [6]) and interface capturing (Eulerian) methods (e.g. the level set (LS) [7–9] and the VOF methods). In interface tracking methods, the interface location is explicitly known, and so the incorporation of surface tension and interfacial boundary conditions is relatively simple and accurate. The disadvantage of such methods is that as the interface topology changes, “surgical” processes are needed to smooth and distribute the interfacial elements utilized to represent the interface. These processes can become complicated and arbitrary when interfaces merge or rupture, especially in 3D [4].

In interface capturing methods, the location of an interface is implicitly represented by an indicator (scalar) function on an Eulerian mesh [4]. This complicates the calculation of surface tension, but these methods more easily allow interfaces to merge or rupture. In interface capturing methods, the interface is evolved on a fixed numerical mesh by solving the following advection equation

$$\frac{\partial \chi}{\partial t} + \vec{U} \cdot \nabla \chi = 0$$

(1)

where $\chi$ denotes the indicator scalar function (for example, the LS function or the VOF function), and $\vec{U}$ is the velocity. At each timestep, the interface unit normal vector $\hat{n}$ and curvature $\kappa$ can be calculated from the spatial derivatives of $\chi$ via

$$\hat{n} = \frac{\nabla \chi}{|\nabla \chi|}$$

(2)
and

\[ \kappa = -\nabla \cdot \hat{n} \]  

(3)

In the VOF method, the indicator function, denoted by \( f \), is discontinuous: for an interfacial flow consisting of fluids 1 and 2, \( f \) is, for example, set equal to one in fluid 1, and zero in fluid 2. The discretized form of \( f \) is the “volume fraction” of each numerical cell occupied by fluid 1: one in a full cell, zero in an empty cell, and between zero and one in a cell that contains a portion of the interface.

Calculating accurate \( \hat{n} \) and \( \kappa \) from volume fractions is challenging because \( f \) is discontinuous. Using Eqs. (2) and (3) directly to calculate \( \hat{n} \) and \( \kappa \) from \( f \) is straightforward, but offers no convergence [10, 11]. More sophisticated methods (e.g. the height functions [12–14] and curve fitting methods [15, 16]) are accurate but become complicated, especially for complex interface geometries and in 3D. What these methods all have in common is that they rely solely on the instantaneous volume fraction field. An alternative is to couple VOF to an altogether different method; for example, the “coupled level set and VOF” (CLSVOF) method [17–20] exploits the strengths of each method: accurate \( \hat{n} \) and \( \kappa \) from the LS function (via Eqs. (2) and (3)) and the volume conservation of the VOF method.

In [10], we introduced a different approach. Instead of computing \( \hat{n} \) from the instantaneous \( f \) field at each timestep (and then discarding those values and calculating new \( \hat{n} \) at the next timestep), we advect both the interface and the interface normals. We denote the new methodology the “advecting \( \vec{N} \)” method, where \( \vec{N} \) are the advected normals. We couple this method to a volume-conserving VOF scheme (Youngs [21]), but other methods could also
be used to model interface kinematics. In this method $f$ and $\vec{N}$ are advected independently, although they are coupled. The advected normals are used to reconstruct the interface to calculate volume fluxes, and to compute $\kappa$ from Eq. (3) to obtain the surface tension force.

In [10], we assessed the advecting $\vec{N}$ method with prescribed velocity fields; the method was not coupled to a flow solver. In this paper, we present an implementation of the method into a two-fluid flow solver. The advantage of the advecting $\vec{N}$ method is more accurate $\hat{n}$ and $\kappa$, certainly than those calculated directly from $f$. The results presented in [10] showed that the method yields second-order accurate $\hat{n}$ in space, and $\kappa$ that are at least first-order accurate. On the other hand, the results reveal two drawbacks: (i) the method does not perform well in under-resolved regions of a flow, where normal vectors vary sharply in space, and (ii) the method is better suited to flows characterized by substantial interface movement than flows near equilibrium, when the actual velocity field can be on the order of interfacial spurious currents. In this paper, we briefly review the method of advecting $\vec{N}$, describe the flow solver, introduce an approach to identifying and dealing with under-resolved interfaces, and then present a set of results that, in order: (i) demonstrate the efficacy of the under-resolvedness correction, (ii) demonstrate the behavior of the method for a near-equilibrium flow, and (iii) illustrate how well the method works for more interesting flows with substantial interface deformation.
2. Mathematical Formulations

2.1. Flow equations

Consider a two-phase flow of immiscible, incompressible, and Newtonian fluids. The governing equations for the flow are conservation of mass and momentum

\[ \nabla \cdot \vec{U} = 0 \]  \hspace{1cm} (4)

\[ \frac{\partial (\rho \vec{U})}{\partial t} + \nabla \cdot (\rho \vec{U} \vec{U}) = -\nabla p + \nabla \cdot \tau + \vec{F}_B + \vec{F}_{ST} \] \hspace{1cm} (5)

where \( p \) denotes pressure, \( \vec{F}_B \) any body forces such as gravity, \( \vec{F}_{ST} \) the surface tension force, and \( \tau = \mu (\nabla \vec{U} + (\nabla \vec{U})^T) \) the shear stress tensor. A single set of Eqs. (4) and (5) is solved for the flow field in both fluids. The fluid properties at any point are calculated from \( f \), which represents fluid 1

\[ \rho(f) = f \rho_1 + (1 - f) \rho_2 \] \hspace{1cm} (6)

and

\[ \mu(f) = f \mu_1 + (1 - f) \mu_2 \] \hspace{1cm} (7)

where subscripts 1 and 2 denote fluids 1 and 2. The advection equation for \( f \) is also solved

\[ \frac{\partial f}{\partial t} + \vec{U} \cdot \nabla f = 0 \] \hspace{1cm} (8)

The following boundary conditions apply at fluid interfaces. The normal components of velocity in each fluid are equal (provided that there is no mass interchange between two different phases of a single substance) [22]

\[ [\vec{U} \cdot \hat{n}] \equiv \vec{U}_1 \cdot \hat{n} - \vec{U}_2 \cdot \hat{n} = 0 \] \hspace{1cm} (9)
If viscous effects are present, the tangential components of the velocity are also equal

$$[\vec{U}] = 0 \quad (10)$$

Finally, the following is a jump condition for stress

$$[(-pI + \tau) \cdot \hat{n}] = \sigma \kappa \hat{n} \quad (11)$$

where $\sigma$ is the coefficient of surface tension (assumed constant), and $I$ is the unit tensor.

2.2. Advecting normals

The details of the advecting $\vec{N}$ method are given in [10]; the following is a brief overview. As mentioned previously, we use the VOF method to advect fluid interfaces, but this is simply our choice, and other methods could also be used. To derive an advection equation for normals, however, we exploit the properties of the LS function, which is denoted by $\phi$ and defined as a signed distance function to an interface. Consider the advection equation for $\phi$,

$$\frac{\partial \phi}{\partial t} + \vec{U} \cdot \nabla \phi = 0 \quad (12)$$

Defining $\vec{N} = \nabla \phi$ as the vector normal to the contours of $\phi$, the above equation can be rewritten as

$$\frac{\partial \phi}{\partial t} + \vec{U} \cdot \vec{N} = 0 \quad (13)$$

Taking the gradient of Eq. (13), we obtain an advection equation for normals

$$\frac{\partial \vec{N}}{\partial t} + \nabla \left( \vec{U} \cdot \vec{N} \right) = 0 \quad (14)$$
From [23], when $\phi$ is advected, it remains a distance function if and only if $\nabla U_n \cdot \nabla \phi = 0$, where $U_n = \vec{U} \cdot \nabla \phi$. This condition can also be expressed as

$$\nabla (\vec{U} \cdot \vec{N}) \cdot \vec{N} = 0$$  \hspace{1cm} (15)

Note that $|\nabla \phi| = |\vec{N}| = 1$ (although $\vec{N}$ is a unit vector, we denote it in this way to distinguish advected normals from the $\hat{n}$ calculated via Eq. (2)). From Eq. (14) we obtain

$$\frac{\partial \vec{N}}{\partial t} \cdot \vec{N} + \nabla \left( \vec{U} \cdot \vec{N} \right) \cdot \vec{N} = 0$$  \hspace{1cm} (16)

or

$$\frac{1}{2} \frac{\partial}{\partial t} \left( |\vec{N}|^2 \right) + \nabla \left( \vec{U} \cdot \vec{N} \right) \cdot \vec{N} = 0$$  \hspace{1cm} (17)

If Eq. (15) is to be satisfied, then we have

$$\frac{\partial}{\partial t} \left( |\vec{N}|^2 \right) = 0$$  \hspace{1cm} (18)

and since initially $|\nabla \phi| = |\vec{N}| = 1$, then $\vec{N}$ remains a unit vector. In other words, if $\vec{N}$ is initially a unit vector and we solve Eq. (14) to advect $\vec{N}$ while satisfying Eq. (15), then $\vec{N}$ remains a unit vector (in a sense, satisfying Eq. (15) is similar to LS reinitialization).

The interface curvature at any point is then obtained directly from the advected $\vec{N}$

$$\kappa = -\nabla \cdot \vec{N}$$  \hspace{1cm} (19)

3. Numerical Methodology

3.1. Flow solver

We employ a two-step projection method in the finite volume context to discretize Eqs. (4) and (5) on a 2D Cartesian co-located grid. Using a
first-order scheme to discretize the temporal derivative, we solve Eq. (5) by
splitting it into predictor and corrector steps

\[
\frac{\rho^{n+1}\vec{U}^* - \rho^n\vec{U}^n}{\Delta t} = -\nabla \cdot \left( \rho\vec{U}\vec{U} \right)^n + \nabla \cdot \left( \mu \left( \nabla\vec{U} + \nabla^T\vec{U} \right) \right)^n + F_B^n \tag{20}
\]

\[
\frac{\rho^{n+1}\vec{U}^{n+1} - \rho^{n+1}\vec{U}^*}{\Delta t} = -\nabla p^{n+1} + F_{ST}^{n+1} \tag{21}
\]

where superscripts \( n \) and \( n + 1 \) denote the current and next time levels, and
* an interim level.

Mass (volume) is advected first via the conservative VOF method of Youngs [21] in order to obtain the \( f^{n+1} \) field. Interfaces are reconstructed at
each timestep using the known volume fractions and \( \vec{N} \) at time \( n \). Volume
fluxes across control volume (CV) faces are then evaluated geometrically by
calculating areas like \( A_1 \) and \( A_2 \) shown in Figure 1.

Solution of Eq. (20) begins by advecting momentum in a way that is con-
sistent with the mass conservation, by evaluating momentum fluxes based
on the density of the already-calculated volume fluxes [24]. For example,
consider Figure 1: the density that is used to compute the momentum flux
from cell \( i \) to \( i + 1 \) is \( \left( \rho_1 A_1 + \rho_2 A_2 \right) / (A_1 + A_2) \), which is given by the cor-
responding volume flux. This scheme yields a tight coupling between mass
and momentum transport, which is required for accurate modeling of large
density ratio interfacial flows. To calculate the momentum flux, we also need
the “advected” velocities associated with flux volume. Bussmann et al. [24]
used an upwinding scheme to approximate these velocities everywhere in a
flow field. Instead, we employ a Taylor series (TS) expansion to calculate
these velocities everywhere except across an interface; there we use an upwind
value. We use the second-order van Leer slope limiter [25] to approximate
only the first derivative of the TS, and then construct the TS from the donor cell. Returning to Eq. (20), the remainder of the RHS, the viscous and body force terms, are evaluated in a straightforward manner, in order to obtain the interim velocity $\vec{U}^*$. 

Equation (21) requires the surface tension $\vec{F}_{ST}$ calculated from the $f^{n+1}$ field; we use the balanced force algorithm of Francois et al. [12] that imposes an exact balance between pressure and surface tension. $\nabla p$ and $\vec{F}_{ST}$ are treated consistently, and thus no spurious currents are induced in a flow when the correct interface curvature is prescribed. We use the continuous approach, which is based on the well-known CSF model [26], where the surface tension force is defined as

$$\vec{F}_{ST} = \sigma \kappa \hat{n} \delta$$

where $\delta$ is the Dirac delta function used to represent the interface. In [12], $\hat{n} \delta$ is approximated as $\nabla f$

$$\vec{F}_{ST} = \sigma \kappa \nabla f$$

where $f$ is not smoothed prior to calculating $\nabla f$. It follows that $\vec{F}_{ST}$ is non-zero where $\nabla f$ is non-zero.

Finally, applying Eq. (4) to Eq. (21) yields an implicit equation for pressure

$$-\nabla^2 p^{n+1} = \frac{1}{\Delta t} \nabla \cdot \left( -\rho^{n+1} \vec{U}^* \right) - \nabla \cdot \vec{F}_{ST}^{n+1}$$

(22)

$p^{n+1}$ is obtained by solving Eq. (22) iteratively using an Incomplete Cholesky Conjugate Gradient (ICCG) solver. $\vec{U}^{n+1}$ is then evaluated via Eq. (21).
3.2. Advecting normals

Equation (14) represents an initial value problem, with the initial value of $\vec{N}$ determined from the initial geometry of an interface. To solve Eq. (14), the temporal derivative is discretized using a third-order total variation diminishing (TVD) Runge-Kutta method [27] for higher numerical accuracy. The spatial derivatives are discretized using a weighted essentially non-oscillatory (WENO) scheme [4]. However, before evaluating the spatial derivatives, Eq. (15) is satisfied via an “extension” algorithm, detailed in [10].

In our implementation, $\vec{N}$ are located at cell vertices (see Figure 2); cell-centered normal vectors, which are required for interface reconstruction, are obtained by bilinear interpolation. As shown in Figure 2, we define $\vec{N}$ in a band around the interface defined by the VOF function. The interface reconstruction only requires $\vec{N}$ at vertices of interfacial cells, but to calculate surface tension we require $\kappa$ in neighboring cells as well; thus, the band spans $3\Delta x$.

3.2.1. Calculating normals in under-resolved regions

In under-resolved regions of a flow, normals associated with different interfaces will approach one another, as for example at a corner. The normals will vary sharply and so yield inaccurate spatial derivatives for Eq. (14), and curvatures with large errors [10]. Here, we address this issue.

Consider the idealized example illustrated in Figure 3, where two adjacent interfaces intersect at a corner. This configuration is under-resolved because two interfaces are too close to each other and so the bands of $\vec{N}$ overlap. The solid circles in Figure 3 illustrate points at which $\vec{N}$ are required for both interfaces, but will only be available for one, and will thus distort the
calculation of \( \vec{N} \) for the other. We remedy the issue of under-resolvedness by using the VOF function to calculate normals via Eq. (2) in such under-resolved regions, instead of the advected \( \vec{N} \). Once some minimum resolution is restored to such regions, we then switch back to the advected \( \vec{N} \), using \( \hat{n} \) calculated from the VOF function as the initial condition for Eq. (14). Note that for coarsely-defined interfaces the VOF function yields fairly accurate \( \hat{n} \) via Eq. (2) (see Table 7 in [10]). We refer to this as an “under-resolvedness” correction.

The criterion for assessing under-resolvedness is based on curvature [28]. Consider the fluid ligament (gray) shown in Figure 4. At points depicted by the solid circles, \( \vec{N} \) is undefined. These points appear when the distance between two interfaces is \( 4\Delta x \) or less (recall that \( \vec{N} \) bands span at least \( 3\Delta x \)). Consider point P in Figure 4; one can fit a circle (dashed line) centered at P that is tangent to the interfaces. Such a circle, for which \( \kappa \cdot \Delta x \geq 0.5 \), will contain at least one point at which \( \vec{N} \) is undefined. Thus, we use \( \kappa \cdot \Delta x \) to establish a criterion for assessing under-resolvedness. When interfaces are \( 5\Delta x \) apart (which corresponds to \( \kappa \cdot \Delta x = 0.4 \)) \( \vec{N} \)-bands will not overlap, but as a safety factor, we set the criterion to

\[
\kappa \cdot \Delta x \geq \frac{1}{3}
\]  

(23)

The under-resolvedness correction is implemented as follows: we first use \( \vec{N} \) to calculate \( \kappa \) in a cell \((i, j)\). If condition (23) is satisfied, then the interface is flagged as under-resolved at that cell, and we revert to the VOF function to calculate (and overwrite) \( \vec{N} \) on a stencil centered at \( \vec{N}_{(i,j)} \) (for the cell \((i, j)\) shown in Figure 4, \( \vec{N}_{(i,j)} \) is located at P). The stencil size is determined by the components of \( \vec{N}_{(i,j)} = (N_x, N_y) \). It is \( 5 \times 3 \) if \( |N_x| > |N_y| \) (i.e. the
stencil, centered at $\vec{N}_{(i,j)}$, spans 5 cell vertices in the $x$-direction and 3 cell vertices in the $y$-direction), $3 \times 5$ if $|N_y| > |N_x|$, or $5 \times 5$ if $|N_x| = |N_y|$. For example, assuming $|N_y| > |N_x|$ at P in Figure 4, the stencil is ABCD.

4. Results

We now present various results of the advected $\vec{N}$ method and compare them to those obtained by calculating $\hat{n}$ and $\kappa$ via Eqs. (2) and (3) from either the LS function in a CLSVOF method, or from the VOF function itself (see [10] for details). All tests were run on uniform meshes. First, we assess the under-resolvedness correction by revisiting an advection test presented earlier [10]. Then, the performance of the advecting $\vec{N}$ method coupled to an interfacial flow solver is assessed. Fluid volumes are exactly conserved in all test cases.

4.1. Prescribed velocity field: vortex test

In [10], we showed that the advecting $\vec{N}$ method (without the under-resolvedness correction) only performed well when interfaces were well-resolved. We revisit one of the tests here, and assess the performance of the advecting $\vec{N}$ method with the under-resolvedness correction.

Consider a circle of radius 0.15 initially placed at $(0.5,0.75)$ in a $1 \times 1$ domain. The following velocity field is specified

$$u = \sin^2(\pi x) \sin(2\pi y)$$  \hspace{1cm} (24)

$$v = -\sin^2(\pi y) \sin(2\pi x)$$  \hspace{1cm} (25)

The circle is advected to $t = 1$; then the sign of the velocity field is reversed and advection continues to $t = 2$, at which point the circle should have
returned to its initial configuration. We used the advecting $\vec{N}$ method with under-resolvedness correction to calculate $\hat{n}$. Tests were performed at two mesh resolutions: $\Delta x = 1/100$ and $1/200$, at a maximum Courant number of 0.1. The results are illustrated in Figure 5. At $\Delta x = 1/100$, the under-resolvedness correction was applied to an average of 8.3% of $\vec{N}$; the maximum number at any timestep was 30%, at $t = 0.96$. The corresponding values at $\Delta x = 1/200$ are 1.6% (average) and 5.7% (maximum, at $t = 0.965$).

Table 1 presents $l_\infty$ and $l_1$ errors of the volume fraction field at $t = 2$, defined as

$$ l_\infty = \max_j |(f - f_{\text{exact}})_j| $$

and

$$ l_1 = \frac{1}{N} \sum_{j=1}^{N} |(f - f_{\text{exact}})_j| $$

where $N$ includes all cells in the computational domain (i.e. not just the interface cells). The results reported in [10] are reproduced in Table 1. As demonstrated, the under-resolvedness correction significantly improves the results, as the $l_\infty$ and $l_1$ errors decrease considerably. Note too that the results of the advecting $\vec{N}$ method with the correction are more accurate than those of the VOF function, and only slightly less accurate than the results of the LS function. These results clearly demonstrate that the under-resolvedness correction significantly improves the performance of the advecting $\vec{N}$ method.

4.2. Solving for flow

4.2.1. Static drop in the absence of gravity

Consider a 2D drop of fluid 1 of radius $R = 0.25$ centered in a 1×1 domain otherwise filled with fluid 2, in zero gravity. Both fluids are initially
quiescent. \( \rho_1 = \rho_2 = 10^3, \mu_1 = \mu_2 = 5 \times 10^{-2}, \sigma = 0.1 \). This corresponds to an Ohnesorge number, which shows the ratio of viscous forces to inertia and surface tension forces, \( \text{Oh} = \mu/\sqrt{2R \rho \sigma} = 7 \times 10^{-3} \). For drops or bubbles, in particular, the Ohnesorge number shows the ratio of the viscous damping rate to the natural frequency of a drop or bubble. \( \Delta t = 1/256 \) and \( \Delta x = 1/128 \).

There are only pressure and surface tension forces present. The force balance relates pressure across the drop surface to surface tension via

\[
\Delta p = p_1 - p_2 = \sigma \kappa
\]

(28)

The exact velocity field is zero. Numerically, however, because of errors in computing \( \kappa \), spurious currents develop that distort the \( \vec{N} \) field.

We used the advected \( \vec{N} \) as well as the VOF and LS functions to calculate \( \hat{n} \) and \( \kappa \), and examined the magnitudes of dimensionless spurious currents \( \vec{U} \mu/\sigma \), and the pressure jump across the interface evaluated in three ways: \( \Delta p_{\text{total}} \) denotes the difference between average pressures in the \( r \leq R \) and \( r > R \) regions; \( \Delta p_{\text{partial}} \) represents the difference between average pressures in the \( r \leq R/2 \) and \( r > 3R/2 \) regions (to avoid the transition region near the interface); \( \Delta p_{\text{max}} \) is the difference between the maximum and minimum pressures in the domain.

We first consider the results after \( 10^3 \) timesteps. Figure 6 shows spurious currents, and Table 2 presents the magnitude of spurious currents, the pressure jumps, and the total kinetic energy (TKE) summed over the whole domain. The advecting \( \vec{N} \) method yields the best results: the lowest spurious currents, the smallest TKE, and the most accurate pressure jumps (the exact pressure jump is 0.4).

Table 2 also presents the same data after \( 10^4 \) timesteps. The VOF results
are almost the same at $10^3$ and $10^4$ timesteps, and reflect some sort of steady state. They are still the least accurate of all, due to large errors in $\kappa$. The LS and advected $\vec{N}$ spurious currents, on the other hand, grow during that time, and in fact the advected $\vec{N}$ currents are slightly larger than the LS currents. This is because the spurious currents are “noise” to the velocity field. They gradually distort the $\vec{N}$ via Eq. (14), and lead to an increase of errors in $\kappa$. This, in turn, exacerbates the spurious currents and magnifies the errors in $\vec{N}$ and $\kappa$. The advected $\vec{N}$ are more sensitive to the spurious currents than the LS function because the advecting $\vec{N}$ method solves for both the evolution of the indicator scalar function as well as its gradient; slight errors in the velocity field are thus magnified. Although this test appears to indicate that the advected $\vec{N}$ are more sensitive to spurious currents, hence the method is perhaps better suited to flows that involve more substantial interface movement, it is also important to note that the static drop test is an extreme case. In spite of that, the advecting $\vec{N}$ method was most accurate for an extended period and degraded only when exposed to a static interface (hardly an interesting flow) for a very long time.

4.2.2. Buoyancy-driven flow

Next, consider a bubble of fluid 1, of radius $R = 1/3$, positioned at (1,1) in a $2 \times 3$ container otherwise filled with fluid 2. $\rho_1 = 1$, $\rho_2 = 10^3$, $\mu_1 = 1.78 \times 10^{-5}$, $\mu_2 = 1.137 \times 10^{-3}$, $\sigma = 73$, $g = 9.81$. This corresponds to $\text{Oh}_1 = 2.55 \times 10^{-6}$, $\text{Oh}_2 = 5.15 \times 10^{-6}$ and Bond numbers $\text{Bo}_1 = 4\rho_1 g R^2 / \sigma = 0.06$ and $\text{Bo}_2 = 60$. The same problem was studied by Kang et al. [29] using the LS method, and by Francois et al. [12] using a VOF-based height function method.
A no-slip boundary condition was applied at all boundaries. Using the advecting $\vec{N}$ method, we ran the simulations at two mesh sizes $\Delta x = 1/20$ and $1/40$, at $\Delta t = 1/256$ for 128 timesteps. Figure 7 shows the bubble shape and the corresponding normal vectors at $t = 0.2$, $0.35$, and $0.5$. Despite major interface deformation, the advecting $\vec{N}$ method works very well. As they follow the velocity field, the normal vectors separate at the bottom of the bubble, properly pointing to interfaces on either side of the vertical plane of symmetry.

We then decreased $\sigma$ to 0.073 ($\text{Oh}_1 = 8.07 \times 10^{-5}$ and $\text{Bo}_1 = 60$), which corresponds to a large bubble that is known to be unstable [29]. The timestep and mesh sizes were the same as for the previous case. The bubble shapes are depicted in Figure 8 at $t = 0.2$, $0.35$, and $0.5$, and agree very well with those presented in [29]. At the higher resolution (Figure 8 (b)) the bubble ruptures due to the Rayleigh-Taylor and Kelvin-Helmholtz instabilities, which reflect density and velocity differences respectively [29]. Since the surface tension forces are small, these instabilities grow and dominate the flow. This test demonstrates that the advecting $\vec{N}$ method is well capable of modeling flows that involve interface breakup.

4.2.3. Rayleigh-Taylor instability test

We conclude with a Rayleigh-Taylor instability test also performed by, among others, Gómez et al. [30]. Consider fluid 1 ($\rho_1 = 1.225$) above fluid 2 ($\rho_2 = 0.1694$) in a $1 \times 4$ domain, depicted in Figure 9. The volumes of fluids 1 and 2 are equal. $\mu_1 = \mu_2 = 3.13 \times 10^{-3}$, $\sigma = 0.1337$, and $g = 9.81$. At
\( t = 0 \), the interface shape is given by the following position vector \( \vec{R} \)

\[
\vec{R}(x) = x \hat{i} + [2 + 0.05 \cos(2\pi x)] \hat{j}
\] (29)

Due to the symmetry of the problem, only half of the domain was solved. Free-slip boundary conditions were imposed at all boundaries. We employed the advecting \( \vec{N} \) method with the under-resolvedness correction and used three mesh sizes: \( \Delta x = 1/64, 1/128, \) and \( 1/256 \), with timesteps \( \Delta t = 1/1024, 1/2048, \) and \( 1/8192 \), respectively. Figure 10 illustrates the results for \( \Delta x = 1/256 \). Our results compare well with those obtained by Gómez et al. (Figure 17 in [30]) who used a locally refined LS method, with a main grid of \( \Delta x = 1/64 \) that was divided into \( 4 \times 4 \) subcells near the interface.

To demonstrate the convergence of the results, interface shapes at \( t = 1.3 \) are superimposed in Figure 11; gray represents results at \( \Delta x = 1/256 \), and black depicts those at (a) \( \Delta x = 1/64 \) and (b) \( 1/128 \). For \( \Delta x = 1/64 \), the shape and location of the interface at the leading edge are quite different from those at \( \Delta x = 1/256 \); in other regions of the flow, the results are fairly similar. At \( \Delta x = 1/128 \), the model captures more details of the flow (see Figure 11 (b)), and the shape and location of the interface are very similar to those at \( \Delta x = 1/256 \) almost everywhere. Again, this test shows that the advecting \( \vec{N} \) method handles complex deformations very well.

5. Summary

The advecting normals method was presented in [10] as a new approach to computing interface normals and curvatures. In this method, normal vectors are advected along with the interface, and are employed for interface
reconstruction and for calculating curvature. Here, we presented an implementation of the method into a two-dimensional volume-conserving VOF flow solver, and introduced an approach to handling locally under-resolved interfaces.

We assessed the performance of the advecting normals method via advection as well as actual flow problems, and demonstrated that the method capably handles flows that involve large interface deformation and breakup. The method is better suited to such problems than those that, for example, asymptotes to a steady quiescent flow, because the advected $\vec{N}$ are more sensitive to accumulated errors introduced by spurious currents.

The extension of the advecting normals method to three-dimensions is likely to be straightforward, because the governing equations of the method are solved algebraically without any geometrical treatments that would become complicated in three-dimensional cases, especially in flows with complex interface geometries.

References


Figure 1: Advection of mass and momentum to/from interface cells. $A_1$ and $A_2$ are volume fluxes of fluid 1 (white) and 2 (gray), respectively, from cell $i$ to $i+1$ due to velocity $u$. The same volume fluxes are used to calculate momentum fluxes.
Figure 2: Normal vectors are defined at cell vertices in a $3\Delta x$ band around an interface. The $6 \times 6$ stencil ABCD is used to extend $\vec{U} \cdot \vec{N}$ around point O.
Figure 3: An under-resolved region in a flow, where normals vary sharply or are undefined. The • indicate vertices at which $\vec{N}$ are undefined.
Figure 4: A fluid ligament (gray) where bands of \( \vec{N} \) overlap. The ● indicate vertices at which \( \vec{N} \) are undefined. Assuming \(|N_y| > |N_x|\) at point P, the under-resolvedness correction will apply to \( \vec{N} \) on stencil ABCD, centered at P.
Figure 5: The evolution of a circle in a vortex field using the advecting $\vec{N}$ method with the under-resolvedness correction, with (a) $\Delta x = 1/100$ and (b) $1/200$.

Figure 6: Spurious currents for a static drop in zero gravity after $10^3$ timesteps; interface curvature calculated from (a) the advected $\vec{N}$, (b) the LS function, and (c) the VOF function (velocities reduced five times). $\Delta t = 1/256$, $\Delta x = \Delta y = 1/128$, $\rho_1 = \rho_2 = 10^3$, $\mu_1 = \mu_2 = 5 \times 10^{-2}$, $\sigma = 0.1$, Oh = $7 \times 10^{-3}$. 
Figure 7: Interface shape and normal vectors as a bubble (fluid 1) rises in fluid 2 due to buoyancy. (a) $\Delta x = 1/20$ and (b) $\Delta x = 1/40$. $\rho_1 = 1$, $\rho_2 = 1000$, $\mu_1 = 1.78 \times 10^{-5}$, $\mu_2 = 1.137 \times 10^{-3}$, $\sigma = 73$, $g = 9.81$, $\Delta t = 1/256$. 
Figure 8: Interface shape as a bubble (fluid 1) rises in fluid 2 due to buoyancy. (a) \( \Delta x = \Delta y = 1/20 \) and (b) \( \Delta x = \Delta y = 1/40 \). \( \rho_1 = 1, \rho_2 = 1000, \mu_1 = 1.78 \times 10^{-5} \), \( \mu_2 = 1.137 \times 10^{-3} \), \( \sigma = 0.073 \), \( g = 9.81 \), \( \Delta t = 1/256 \).
Figure 9: Initial configuration for the Rayleigh-Taylor instability test. $\rho_1 = 1.225$, $\rho_2 = 0.1694$, $\mu_1 = \mu_2 = 3.13 \times 10^{-3}$, $\sigma = 0.1337$, $g = 9.81$. 

$\rho_1 = 1.225$, $\rho_2 = 0.1694$, $\mu_1 = \mu_2 = 3.13 \times 10^{-3}$, $\sigma = 0.1337$, $g = 9.81$. 

31
Figure 10: Interface evolution in the Rayleigh-Taylor instability test using the advecting $\vec{N}$ method with the under-resolvedness correction. $\Delta x = \Delta y = 1/256$. 

$t = 0.5$  $t = 0.7$  $t = 0.9$  $t = 1.0$  $t = 1.1$  $t = 1.2$  $t = 1.3$
Figure 11: Interface topology at $t = 1.3$ s in the Rayleigh-Taylor instability test using the advecting $\vec{N}$ method with the under-resolvedness correction, at different mesh resolutions. Gray represents results with $\Delta x = 1/256$, black shows results with (a) $\Delta x = 1/64$ and (b) $\Delta x = 1/128$. 
Table 1: Errors associated with volume fractions for the vortex test at $t = 2$, when $\hat{n}$ are calculated from the advected $\vec{N}$ with and without the under-resolvedness correction, and from the VOF and the LS functions.

<table>
<thead>
<tr>
<th>$\Delta x$</th>
<th>$l_\infty$</th>
<th>$l_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advected $\vec{N}$ with correction</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/100</td>
<td>0.305</td>
<td>$1.41 \times 10^{-3}$</td>
</tr>
<tr>
<td>1/200</td>
<td>0.298</td>
<td>$1.30 \times 10^{-4}$</td>
</tr>
<tr>
<td>Advected $\vec{N}$ without correction</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/100</td>
<td>1.000</td>
<td>$3.99 \times 10^{-3}$</td>
</tr>
<tr>
<td>1/200</td>
<td>0.971</td>
<td>$1.57 \times 10^{-3}$</td>
</tr>
<tr>
<td>VOF function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/100</td>
<td>0.433</td>
<td>$1.70 \times 10^{-3}$</td>
</tr>
<tr>
<td>1/200</td>
<td>0.416</td>
<td>$3.18 \times 10^{-4}$</td>
</tr>
<tr>
<td>LS function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/100</td>
<td>0.302</td>
<td>$1.31 \times 10^{-3}$</td>
</tr>
<tr>
<td>1/200</td>
<td>0.259</td>
<td>$1.00 \times 10^{-4}$</td>
</tr>
</tbody>
</table>
Table 2: Spurious currents, $\Delta p$, and the total kinetic energy (TKE) of a static drop in zero gravity after $10^3$ and $10^4$ timesteps, for different methods of calculating $\kappa$. $\Delta t = 1/256$, $\Delta x = \Delta y = 1/128$, $\rho_1 = \rho_2 = 10^3$, $\mu_1 = \mu_2 = 5 \times 10^{-2}$, $\sigma = 0.1$, Oh = $7 \times 10^{-3}$, $\Delta p_{\text{exact}} = 0.4$.

| $10^3$ timesteps | $|\vec{U} \mu/\sigma|_{\text{max}}$ | $|\vec{U} \mu/\sigma|_{\text{ave.}}$ | $\Delta p_{\text{partial}}$ | $\Delta p_{\text{total}}$ | $\Delta p_{\text{max}}$ | TKE          |
|------------------|----------------------------------|---------------------------------|----------------------------|----------------|----------------|-------------|
| Advected $\vec{N}$ | $6.01 \times 10^{-7}$            | $3.27 \times 10^{-8}$           | 0.3999                    | 0.3999        | 0.4002        | $9.38 \times 10^{-12}$ |
| LS function      | $2.32 \times 10^{-5}$            | $2.02 \times 10^{-6}$           | 0.3983                    | 0.3945        | 0.4078        | $2.30 \times 10^{-8}$  |
| VOF function     | $8.30 \times 10^{-3}$            | $5.85 \times 10^{-5}$           | 0.4121                    | 0.4054        | 0.4613        | $2.25 \times 10^{-5}$  |

| $10^4$ timesteps | $|\vec{U} \mu/\sigma|_{\text{max}}$ | $|\vec{U} \mu/\sigma|_{\text{ave.}}$ | $\Delta p_{\text{partial}}$ | $\Delta p_{\text{total}}$ | $\Delta p_{\text{max}}$ | TKE          |
|------------------|----------------------------------|---------------------------------|----------------------------|----------------|----------------|-------------|
| Advected $\vec{N}$ | $1.63 \times 10^{-4}$            | $1.13 \times 10^{-5}$           | 0.3994                    | 0.3984        | 0.4205        | $7.58 \times 10^{-7}$ |
| LS function      | $9.85 \times 10^{-5}$            | $4.09 \times 10^{-6}$           | 0.4016                    | 0.3970        | 0.4451        | $1.36 \times 10^{-7}$  |
| VOF function     | $6.90 \times 10^{-3}$            | $5.40 \times 10^{-5}$           | 0.4310                    | 0.4254        | 0.5740        | $1.87 \times 10^{-5}$  |