Journal of Computational Physics 249 (2013) 275–302

Contents lists available at SciVerse ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

# A discontinuous Galerkin conservative level set scheme for interface capturing in multiphase flows

Mark Owkes *, Olivier Desjardins

*Sibley School of Mechanical and Aerospace Engineering, Cornell University, Ithaca, NY 14853, USA*

ARTICLE INFO

*Article history:*
Received 18 December 2012
Received in revised form 16 April 2013
Accepted 17 April 2013
Available online 10 May 2013

*Keywords:*
Multiphase flow
DNS
Discontinuous Galerkin
Conservative level set
Incompressible flow
Mass conservation

ABSTRACT

The accurate conservative level set (ACLS) method of Desjardins et al. [O. Desjardins, V. Moureau, H. Pitsch, An accurate conservative level set/ghost fluid method for simulating turbulent atomization, J. Comput. Phys. 227 (18) (2008) 8395–8416] is extended by using a discontinuous Galerkin (DG) discretization. DG allows for the scheme to have an arbitrarily high order of accuracy with the smallest possible computational stencil resulting in an accurate method with good parallel scaling. This work includes a DG implementation of the level set transport equation, which moves the level set with the flow field velocity, and a DG implementation of the reinitialization equation, which is used to maintain the shape of the level set profile to promote good mass conservation. A near second order converging interface curvature is obtained by following a height function methodology (common amongst volume of fluid schemes) in the context of the conservative level set. Various numerical experiments are conducted to test the properties of the method and show excellent results, even on coarse meshes. The tests include Zalesak's disk, two-dimensional deformation of a circle, time evolution of a standing wave, and a study of the Kelvin–Helmholtz instability. Finally, this novel methodology is employed to simulate the break-up of a turbulent liquid jet.



## 1. Introduction

In simulations of multiphase flows, changing fluid properties and surface tension at the interface lead to discontinuities that make discretizing the Navier–Stokes equations challenging. Consequently, numerical methods have been developed to handle these singularities, including the continuum surface force (CSF) approach [1] and the ghost fluid method (GFM) [2]. Both the CSF method and the GFM are based on the assumption that the interface location is known accurately. The discontinuous Galerkin conservative level set method, presented herein, provides an accurate interface location needed for the CSF method, the GFM, or other chosen methods.

Commonly, two classes of methods are used to locate the interface: interface tracking and interface capturing. Interface tracking schemes typically use either arbitrary Lagrangian–Eulerian (ALE) methods based on a mesh that deforms with the interface [3–6] or marker and cell (MAC) methods that advect Lagrangian particles that define a given fluid by their locations [7]. The main problem with interface tracking schemes occurs when the interface deforms substantially or when the interface disconnects and reconnects. Significant re-meshing or re-seeding of particles is needed to account for the large interface changes.

* Corresponding author.
*E-mail address:* mfc86@cornell.edu (M. Owkes).

0021-9991/$ - see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.jcp.2013.04.036

Interface capturing methods include volume of fluid (VOF) and level set methods. VOF methods capture the liquid volume fraction in each grid cell [8–10]. While VOF schemes have excellent mass conservation properties, they suffer from the challenge of reconstructing the interface location using only the cell volume fraction. Level set methods represent the interface as an iso-surface of a function called the level set function [11,12]. Level set methods alleviate the problem found with VOF methods of having to reconstruct an interface since the interface location, normal, and curvature are readily accessible from the level set function. Classically, the level set function is defined as a signed distance function, which lacks conservation properties. The conservation properties were improved with the development of the conservative level set [13–15]. Details of the signed distance and conservative level set methods are given in Section 2.

Spatial discretization of the conservative level set can be performed using finite difference operators, which was done in the accurate conservative level set (ACLS) method [15]; however, the discontinuous Galerkin (DG) discretization method was chosen in this work for its high accuracy and compact stencil [16,17]. High accuracy is obtained by projecting the solution onto high-order discontinuous polynomials, similar to finite element methods. Compactness is a result of the local nature of the polynomials. Since the polynomials are defined on each grid cell, updates do not need global information but rather only information from nearest grid cell neighbors. This small stencil results in minimal communication requirements and a highly parallelizable code. The small stencil also makes the discretization amenable to both structured and unstructured grids. Although a Cartesian structured grid is the focus of this work, the extension of the proposed framework to an unstructured grid is possible. The aforementioned scheme that discretizes the conservative level set using a DG formulation is refered to as the discontinuous Galerkin conservative level set (DG-CLS).

The conservative level set method includes a transport equation that describes the convection of the level set due to the fluid velocity and a reinitialization equation that maintains the shape of the level set. Cockburn and Shu [18] provide a DG discretization of the transport equation with an accurate temporal integration method and appropriate definition of fluxes. The DG discretization was applied to the signed distance level set by Marchandise et al. [19]. A quadrature-free implementation was used wherein all the integrals that appear in the weak form of the equations were precomputed to improve computational efficiency. Marchandise et al. [20] reinitialized the signed distance level set using a recursive contouring algorithm with a fast search tree method to find the smallest distance to the interface, which, for the signed distance level set method, is also the value of the level set function. However, when the conservative level set is used the level set is not a signed distance function and a different reinitialization method is employed. Following the steps of the ACLS method, a compressive-diffusion equation is solved to reinitialize the level set and maintain its profile. We propose to discretize the compressive-diffusion equation using DG in order to maintain a similar order of accuracy for both transport and reinitialization steps. Details of the DG implementation are given in Section 3 which includes background information on our DG formulation in Section 3.1 and the particulars of the spatial discretization of the transport and reinitialization equations in Sections 3.2 and 3.3, respectively.

In Section 4, the stability of the DG-CLS scheme is discussed. We begin the section with a discussion of how to improve the robustness of the scheme when the discretized level set function oscillates above or below the bounds due to the use of high order polynomials. Then, an investigation on the effect of a minimum/maximum preserving limiter [21] is provided. The limiter is designed to improve the boundedness of the conservative level set function. While the limiter is found to improve the boundedness, it significantly reduces the accuracy of the method.

The most straightforward method to integrate the transport and reinitialization equations in time is an explicit scheme such as a Runge–Kutta (RK) method. An explicit scheme does not require global communications, thereby maintains the highly parallelizable nature of the DG spatial discretization. Cockburn and Shu [18] provide a description of the RK methods and show many are stable when high order polynomials are used in the DG approximation of the solution. The total variation diminishing third order RK (TVD-RK3) method is used in this work; details are provided in Section 5.

The interface curvature and normal have direct effects on the solution; therefore, the methods used to calculate these interface properties should be accurate and converge under mesh refinement. In the context of their DG implementation of the signed distance level set method, Marchandise et al. [20] proposed a least squares approach for computing curvature. This method was later employed by Desjardins et al. [15] in the ACLS method, but only first order curvature convergence was observed. Obtaining convergence is difficult with the conservative level set because the level set profile is a relatively sharp approximation of a step function. In fact, sharper profiles lead to smaller conservation errors [15]. When the mesh is refined, one has to choose between sharpening the level set profile such that the number of cells across the profile remains constant, or keeping the profile unchanged thereby improving the resolution of the profile. Choosing the former leads to improved mass conservation, while the latter leads to lower curvature errors. To improve the convergence of the interface curvature while sharpening the level set profile, we applied the height function method commonly used in volume of fluid (VOF) methods [22] to the DG-CLS formulation resulting in a near second order converging curvature. Details of the implementation are described in Section 7.

In Sections 6 and 9, we discuss numerical experiments conducted using the DG-CLS method. Section 6 includes tests that focus on the DG-CLS method and Section 9 has applications that use the DG-CLS scheme coupled with the NGA flow solver [23]. Coupling between DG-CLS and the flow solver is presented in Section 8. The numerical tests include normal and curvature convergence, simulations that examine transport of the level set function, and several canonical multiphase flow problems.

## 2. Mathematical formulation

### 2.1. Signed distance level set

The classical level set method represents the interface as the zero iso-surface of a signed distance function, $G$, i.e.,

$$|G(\boldsymbol{x},t)| = |\boldsymbol{x} - \boldsymbol{x}_I|, \tag{1}$$

where $t$ is time and $\boldsymbol{x}_I$ is the location on the interface that is closest to point $\boldsymbol{x}$. The level set function is defined to be positive on one side of the interface and negative on the other side. In this paper we will consider that $G > 0$ corresponds to the liquid and $G < 0$ corresponds to the gas. By defining the level set function using a signed distance function, the interface is implicitly represented by the zero iso-surface, i.e., $I(t) = \{\boldsymbol{x} \in \Omega | G(\boldsymbol{x},t) = 0\}$, where $I$ is the interface and $\Omega$ is the computational domain formally defined in Section 3.1.

Motion of the interface is achieved by solving the transport equation

$$\frac{\partial G}{\partial t} + \boldsymbol{u} \cdot \nabla G = 0, \tag{2}$$

where $\boldsymbol{u}$ is the velocity vector, which is assumed to be known within $\Omega$. In addition to providing the interface location, the level set is also used to calculate the interface normal vector, $\boldsymbol{n}$, and curvature, $\kappa$, using

$$\boldsymbol{n} = \frac{\nabla G}{|\nabla G|} \tag{3}$$

and

$$\kappa = -\nabla \cdot \boldsymbol{n}. \tag{4}$$

Eqs. (3) and (4) provide an accurate result when the level set is smooth, such as a signed distance function. However, transporting the interface using Eq. (2) will alter the shape of the level set profile when the velocity field is not uniform; therefore, a reinitialization step is added to restore the level set to a signed distance function. A variety of methods can be used to reinitialize the level set to a signed distance function. Fast marching methods involve solving the Eikonal equation $|\nabla G| = 1$ from the interface outwards [12,24]. Closest point algorithms reinitialize the level set function by using a tree-based structure to determine the closest point that lies on the interface [12]. Another common reinitialization method solves the Hamilton–Jacobi equation [25],

$$\frac{\partial G}{\partial \tau} + S(G)(|\nabla G| - 1) = 0, \tag{5}$$

in pseudo-time, $\tau$, until steady state is achieved. In the previous equation, $S(G)$ is a modified sign function described for example by Sussman et al. [26].

The signed distance level set function does not represent any physical quantity; therefore, conservation of the signed distance function will not provide conservation of the volume delimited by the interface. When the signed distance level set is used, the volume of the fluid on either side of the interface can change leading to significant errors especially for applications with complex velocity fields and frequent interface topology changes.

### 2.2. Conservative level set

In an effort to add conservation properties to level set schemes, the signed distance function can be replaced with a modified hyperbolic tangent function [13–15],

$$\Psi(\boldsymbol{x},t) = \frac{1}{2}\left(\tanh\left(\frac{G}{2\varepsilon}\right) + 1\right), \tag{6}$$

where $\varepsilon$ sets the thickness of the profile. Note that the conservative level set function represents the interface using the $\Psi = 0.5$ iso-surface, i.e., $I(t) = \{\boldsymbol{x} \in \Omega | \Psi(\boldsymbol{x},t) = 0.5\}$.

The conservative and signed distance level set functions are shown in Fig. 1 along with the liquid volume fraction. From the figure it is evident that the conservative level set function mimics the liquid volume fraction. In the limit where $\varepsilon$ goes to zero, the conservative level set collapses on the liquid volume fraction and the integral of the conservative level set function is equal to the liquid volume, i.e.,

$$\lim_{\varepsilon \to 0} \int_\Omega \Psi(\boldsymbol{x},t)\partial V = \int_\Omega H(\Psi(\boldsymbol{x},t) - 0.5)\partial V, \tag{7}$$

where $H$ is the Heaviside function and $\partial V$ represent volume elements. A useful consequence of Eq. (7) is that conservation of the conservative level set $\Psi$ results in conservation of the liquid volume when $\varepsilon$ goes to zero. Therefore, $\varepsilon$ is chosen to be as small as possible in order to limit mass conservation errors, while maintaining reasonable resolution of the level set function to avoid numerical difficulties.

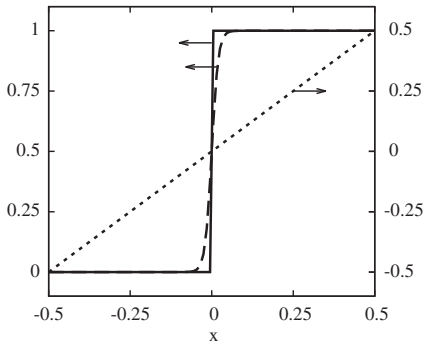



**Fig. 1.** Liquid volume fraction, conservative level set function, and signed distance level set function represented by solid, dashed, and dotted lines, respectively for a one-dimensional problem with the liquid–gas interface located at $x = 0$.

The conservative level set can be transported using Eq. (2) by replacing $G$ with $\Psi$. Furthermore, in the context of a solenoidal velocity field, i.e., $\nabla \cdot \boldsymbol{u} = 0$, the equation can be written in conservative form

$$\frac{\partial \Psi}{\partial t} + \nabla \cdot (\boldsymbol{u}\Psi) = 0, \tag{8}$$

which ensures that $\Psi$ is discretely conserved.

Reinitialization is performed by solving

$$\frac{\partial \Psi}{\partial \tau} + \nabla \cdot (\Psi(1-\Psi)\boldsymbol{n}) = \nabla \cdot (\varepsilon(\nabla\Psi \cdot \boldsymbol{n})\boldsymbol{n}). \tag{9}$$

This equation is developed by recognizing that the level set profile given by Eq. (6) should correspond to the steady-state solution of the reinitialization equation. In one-dimension, the steady-state solution of Eq. (9) verifies

$$\frac{\partial \Psi}{\partial n} = \frac{\Psi(1-\Psi)}{\varepsilon}, \tag{10}$$

which is compatible with Eq. (6). In addition to being written conservatively, Eq. (9) has the property that both the compressive term, shown on the left, and the diffusive term, shown on the right, only act in the direction normal to the interface, represented by $\boldsymbol{n}$. This prevents motion in the direction tangential to the interface. Note that the normal that appears in Eq. (9) is pre-calculated and remains constant throughout the reinitialization step. Similarly to solving the Hamilton–Jacobi equation introduced in Section 2.1, Eq. (9) is advanced in pseudo-time, $\tau$. Note that Eq. (9) is not the only multi-dimensional equation compatible with Eq. (6). For example, Shukla et al. [27] describe another form that does not lead to $\Psi$ being conserved.

## 3. Discontinuous Galerkin implementation

DG schemes offer a variety of desirable properties when applied to the conservative level set method. DG methods allow for arbitrarily high orders of accuracy without the necessity of a large stencil, which results in an accurate and highly scalable scheme. After a background section on DG, the spatial discretization of the level set transport equation is discussed followed by a section addressing the DG discretization of the reinitialization equation.

### 3.1. DG formulation

The computational domain is represented by $\Omega$ with closed boundary $\Gamma$. This domain is represented using a computational grid consisting of a collection of $Q$ non-overlapping grid cells referred to as $\omega_q$ where $q = 1, \ldots, Q$. The union of all cells is equal to the computational domain, i.e.,

$$\Omega = \omega_1 \cup \omega_2 \cup \cdots \cup \omega_Q. \tag{11}$$

Each grid cell $\omega_q$ has an associated closed boundary $\gamma_q$ such that the set of all cell boundary parts that are a member of exactly one cell boundary is equal to the closed physical domain boundary, i.e.,

$$\Gamma = (\gamma_1 \cup \gamma_2 \cup \cdots \cup \gamma_Q) \setminus (\gamma_1 \cap \gamma_2 \cap \cdots \cap \gamma_Q). \tag{12}$$

We should note that the computational grid used in all of the test cases and applications is regular Cartesian. However, the scheme is not limited to Cartesian grids and is applicable to a wide range a grids that meet the criteria outlined above.

DG is used to spatially discretize a function onto the computational grid. The main idea is to represent the function using a linear combination of $P$ basis functions within each grid cell to create a piecewise polynomial representation of the

function. The basis functions, $\boldsymbol{\phi} = \{\phi_1, \phi_2, \ldots, \phi_P\}^\mathsf{T}$, can be almost any set of basis functions; however, orthogonal polynomials are preferred. In this work, the basis functions are taken to be the Legendre polynomials. With these definitions, the level set function can be approximated within the $q$th grid cell using

$$\Psi(\boldsymbol{x}|\boldsymbol{x} \in \omega_q, t) \approx \Psi_{h,q}(\boldsymbol{x}, t) = \sum_{i=1}^{P} \psi_{i,q}(t)\phi_i(\boldsymbol{\chi}_q(\boldsymbol{x})) = \psi_{i,q}\phi_i, \tag{13}$$

where $\Psi_{h,q}$ is the DG approximation of $\Psi$ within the $q$th grid cell, $\psi_{i,q}$ is the weight associated with the $i$th basis function and $q$th grid cell, and $\boldsymbol{\chi}_q$ is a coordinate system local to $\omega_q$ and defined, for a basis formed using Legendre polynomials, such that $\boldsymbol{\chi}_q = [-1, 1]^d$ within $\omega_q$, where $d$ is the dimensionality of $\Omega$. By defining $\boldsymbol{\chi}_q$ this way, the orthogonality relations between the Legendre polynomials are maintained, i.e.,

$$\int_{\omega_q} \phi_i\phi_j \partial V = \delta_{ij} \int_{\omega_q} \phi_i^2 \partial V, \tag{14}$$

where $\delta_{ij}$ is the Kronecker delta. Note, Einstein's summation notation is used throughout this paper for any indices that appear twice in a term unless the index is explicitly defined.

With these definitions, the weights, $\boldsymbol{\psi}_q = \{\psi_{1,q}, \psi_{2,q}, \ldots, \psi_{P,q}\}^\mathsf{T}$, within the $q$th grid cell can be obtained directly from the conservative level set $\Psi$ using

$$\psi_{i,q} = \left(\int_{\omega_q} \Psi\phi_i \partial V\right)\left(\int_{\omega_q} \phi_i^2 \partial V\right)^{-1} \tag{15}$$

for $i = 1, \ldots, P$ and $q = 1, \ldots, Q$. This is useful for initialization routines, when $\Psi$ is known and the approximation $\Psi_h$ needs to be computed.

Legendre polynomials can be generated at order $i$ using

$$\phi_i = \frac{i!}{(2i)!}\frac{\mathrm{d}^i\left[\left(x^2-1\right)^i\right]}{\mathrm{d}x^i}. \tag{16}$$

Multi-dimensional basis sets can be created by combining the functions from one-dimensional sets. Typically, the set is constrained such that the total order is less than a threshold called the total polynomial order, $O$. A total polynomial order of $O$ provides a $O+1$ order accurate representation of the function being approximated. The number of Legendre basis functions and associated weights for a given value of $O$ is referred to as the number of degrees of freedom, $P$, and can be calculated using

$$P = \frac{1}{d!}\prod_{k=1}^{d}(O+k) = \frac{(O+d)!}{O!d!}. \tag{17}$$

For reference, the ten Legendre basis functions within the $q$th grid cell used in a second order ($O = 2$), three-dimensional ($d = 3$) implementation are

$$\boldsymbol{\phi} = \left[1,\ \chi_{q,1},\ \chi_{q,2},\ \chi_{q,3},\ \chi_{q,1}^2 - \frac{1}{3},\ \chi_{q,1}\chi_{q,2},\ \chi_{q,1}\chi_{q,3},\ \chi_{q,2}^2 - \frac{1}{3},\ \chi_{q,2}\chi_{q,3},\ \chi_{q,3}^2 - \frac{1}{3}\right]^\mathsf{T},$$

where $\boldsymbol{\chi}_q = \left(\chi_{q,1}, \chi_{q,2}, \chi_{q,3}\right)^\mathsf{T}$.

Note that, because the basis functions and weights are local to each grid cell, the DG representation of the level set function will be discontinuous at grid cell boundaries. In other words, the approximated function can have different values at a boundary between two cells depending on which cell is used to evaluate the function.

### 3.2. DG level set transport

Transport of the conservative level set is done by solving Eq. (8) using the quadrature-free DG method following the work of Marchandise et al. [19]. The first step to obtain the DG discretization of Eq. (8) is to write the weak form by multiplying by a test function $\phi_s$, integrating over the domain $\Omega$ with boundary $\Gamma$, and performing a formal integration by parts. The result is

$$\int_\Omega \frac{\partial\Psi}{\partial t}\phi_s \partial V - \int_\Omega \Psi u_j \frac{\partial\phi_s}{\partial x_j}\partial V + \oint_\Gamma \Psi\phi_s u_j N_j \partial S = 0 \tag{18}$$

for $s = 1, \ldots, P$, where $N_j$ is the $j$th component of the domain boundary normal vector $\mathtt{N}$.

Spatial discretization introduces the DG approximation of the level set within each grid cell described using Eq. (13) and results in

$$\int_{\omega_q} \frac{\partial \psi_{m,q}\phi_m}{\partial t}\phi_s \partial V - \int_{\omega_q} \psi_{i,q}\phi_i u_j \frac{\partial \phi_s}{\partial x_j}\partial V + \oint_{\gamma_q} \widehat{\psi_{i,q}\phi_i u_j} N_j^q \phi_s \partial S = 0 \tag{19}$$

for $s = 1,\ldots,P$ and $q = 1,\ldots,Q$, where $\boldsymbol{N}^q$ is the normal to the cell boundary with $j$th component $N_j^q$. The hat indicates a flux, which is not uniquely defined since the approximate solution is discontinuous at the cell boundaries. Therefore, an appropriate method to evaluate the flux, $\widehat{\psi_{i,q}\phi_i u_j N_j^q}$, must be chosen. For the transport equation, the flux can be up-winded based on the sign of $(\boldsymbol{u}\cdot\boldsymbol{N}^q)_{\gamma_q}$, which is the velocity at the cell boundary projected onto the cell boundary normal vector. The flux can be written as

$$\widehat{u_j\psi_{i,q}\phi_i}N_j^q = \left(u_jN_j^q\right)_{\gamma_q}\left(\psi_{i,q}\phi_i\right)_{\mathrm{up}} = \begin{cases} \left(u_jN_j^q\right)_{\gamma_q}\left(\psi_{i,q}\phi_i\right)^{\mathrm{in}} & \text{if } \left(u_jN_j^q\right)_{\gamma_q} > 0, \\ \left(u_jN_j^q\right)_{\gamma_q}\left(\psi_{i,q}\phi_i\right)^{\mathrm{out}} & \text{if } \left(u_jN_j^q\right)_{\gamma_q} < 0, \end{cases} \tag{20}$$

where $(*)^{\mathrm{in}}$ is $(*)$ evaluated at the face using information from the cell of interest; analogously, $(*)^{\mathrm{out}}$ is $(*)$ calculated at the face but using information from the neighboring cell. As described by Cockburn and Shu [28], any two-point Lipschitz continuous monotone flux is appropriate, and here the flux is simply up-winded. Adding the properties $\boldsymbol{\psi} = \boldsymbol{\psi}(t)$, $\boldsymbol{\phi} = \boldsymbol{\phi}\left(\boldsymbol{\chi}_q(\boldsymbol{x})\right)$, and $\boldsymbol{u} = \boldsymbol{u}(t)$ within the $q$th cell results in

$$\frac{\partial \psi_{m,q}}{\partial t}\int_{\omega_q}\phi_m\phi_s\partial V - \psi_{i,q}u_j\int_{\omega_q}\phi_i\frac{\partial\phi_s}{\partial x_j}\partial V + \left(u_jN_j^q\right)_{\gamma_q}\left(\psi_{i,q}\right)_{\mathrm{up}}\oint_{\gamma_q}\left(\phi_i\right)_{\mathrm{up}}\phi_s\partial S = 0 \tag{21}$$

for $s = 1,\ldots,P$ and $q = 1,\ldots,Q$. The assumption of $\boldsymbol{u}$ being constant within each cell allows for it to be removed from the integral, allowing for the integral to be evaluated using a quadrature-free approach. This assumption is reasonable since a second order accurate Navier–Stokes solver is used in our work. For higher order coupling with the Navier–Stokes solver, the velocity could be kept within the integral and allow to vary, however doing so is likely to require a quadrature be used to evaluate the integral. In addition, the surface normal, $\boldsymbol{N}^q$, is a constant on regular grids wherein cell boundaries are composed of a collection of flat faces, and the surface integral is split into multiple integrals on each flat face.

Since our DG formulation is based on orthogonal basis functions as shown by Eq. (14), the $P$ coupled equations in Eq. (21) can be decoupled and written as

$$\frac{\partial \psi_{m,q}}{\partial t}\int_{\omega_q}\phi_m^2\partial V - \psi_{i,q}u_j\int_{\omega_q}\phi_i\frac{\partial\phi_m}{\partial x_j}\partial V + \left(u_jN_j^q\right)_{\gamma_q}\left(\psi_{i,q}\right)_{\mathrm{up}}\oint_{\gamma_q}\left(\phi_i\right)_{\mathrm{up}}\phi_m\partial S = 0 \tag{22}$$

for $m = 1,\ldots,P$ and $q = 1,\ldots,Q$. If the basis is not orthogonal, the system of $P$ coupled equations described by Eq. (21) must be solved.

All the integrals in Eq. (22) are now a function of only the basis functions that are defined at the start of a simulation and can be computed once in an initialization routine. In practice on Cartesian meshes, one can compute the integrals over an arbitrary cell size, e.g. $[-1,1]^d$, and then use a scaling factor to adjust the pre-computed integrals to the size of the cell being updated. Equation (22) is the quadrature-free discontinuous Galerkin form of the transport equation that can easily and efficiency be updated once a time integration scheme is chosen, as described in Section 5.

### 3.3. DG level set reinitialization

Reinitialization is used to maintain the shape of the hyperbolic tangent profile and limit mass loss. Eq. (9) is solved using a DG discretization applied in a similar fashion as described in the transport section. The steps include construction of a weak form of the reinitialization equation, discretization on the grid that supports the approximate solution, definition of proper fluxes that provide a unique value at faces where discontinuities exist, and finally, decoupling of the equations if an orthogonal set of basis functions is used.

The weak form of Eq. (9) is found by multiplying by a test function $\phi_s$, integrating over the domain, and performing a formal integration by parts, resulting in

$$\int_\Omega \frac{\partial\Psi}{\partial\tau}\phi_s\partial V - \int_\Omega\left(\Psi-\Psi^2\right)n_j\frac{\partial\phi_s}{\partial x_j}\partial V + \oint_\Gamma\left(\Psi-\Psi^2\right)n_jN_j\phi_s\partial S = -\int_\Omega \varepsilon n_jn_k\frac{\partial\Psi}{\partial x_j}\frac{\partial\phi_s}{\partial x_k}\partial V + \oint_\Gamma\varepsilon\frac{\partial\Psi}{\partial x_j}n_jn_kN_k\phi_s\partial S \tag{23}$$

for $s = 1,\ldots,P$.

Discretization onto a grid involves integrating over cells and substituting in the DG approximation of the level set shown by Eq. (13). The spatially discretized form of Eq. (23) in the $q$th cell is

$$\begin{aligned}\int_{\omega_q}\frac{\partial\psi_{m,q}\phi_m}{\partial\tau}\phi_s\partial V - \int_{\omega_q}\psi_{i,q}\phi_in_j\frac{\partial\phi_s}{\partial x_j}\partial V + \oint_{\gamma_q}\widehat{\psi_{i,q}\phi_in_j}N_j^q\phi_s\partial S + \int_{\omega_q}\psi_{i,q}\phi_i\psi_{k,q}\phi_kn_j\frac{\partial\phi_s}{\partial x_j}\partial V - \oint_{\gamma_q}\widehat{\psi_{i,q}\phi_i\psi_{k,q}\phi_k}n_jN_j^q\phi_s\partial S \\ = -\int_{\omega_q}\varepsilon n_jn_k\frac{\partial\psi_{i,q}\phi_i}{\partial x_j}\frac{\partial\phi_s}{\partial x_k}\partial V + \oint_{\gamma_q}\varepsilon\widehat{\frac{\partial\psi_{i,q}\phi_i}{\partial x_j}}n_jn_kN_k^q\phi_s\partial S\end{aligned} \tag{24}$$

for $s = 1, \ldots, P$ and $q = 1, \ldots, Q$. The fluxes, still shown with hats, are not uniquely defined because the solution is discontinuous at the cell boundaries and must be constructed appropriately. The compressive fluxes are dealt with similarly to the flux in the transport equation and are up-winded using

$$\widehat{\psi_{i,q}\phi_i n_j} N_j^q = (\psi_{i,q}\phi_i)_{\text{up}}(n_j N_j^q)_{\gamma_q} = \begin{cases} (\psi_{i,q}\phi_i)^{\text{in}}(n_j N_j^q)_{\gamma_q} & \text{if } (1-2\psi_{i,q}\phi_i)(n_j N_j^q)_{\gamma_q} > 0, \\ (\psi_{i,q}\phi_i)^{\text{out}}(n_j N_j^q)_{\gamma_q} & \text{if } (1-2\psi_{i,q}\phi_i)(n_j N_j^q)_{\gamma_q} < 0, \end{cases} \tag{25a}$$

$$\widehat{\psi_{i,q}\psi_{k,q}\phi_i\phi_k n_j} N_j^q = (\psi_{i,q}\psi_{k,q}\phi_i\phi_k)_{\text{up}}(n_j N_j^q)_{\gamma_q} = \begin{cases} (\psi_{i,q}\psi_{k,q}\phi_i\phi_k)^{\text{in}}(n_j N_j^q)_{\gamma_q} & \text{if } (1-2\psi_{i,q}\phi_i)(n_j N_j^q)_{\gamma_q} > 0, \\ (\psi_{i,q}\psi_{k,q}\phi_i\phi_k)^{\text{out}}(n_j N_j^q)_{\gamma_q} & \text{if } (1-2\psi_{i,q}\phi_i)(n_j N_j^q)_{\gamma_q} < 0, \end{cases} \tag{25b}$$

where $(n_j N_j^q)_{\gamma_q}$ is the interface normal $\boldsymbol{n}$ projected onto the cell face normal $\boldsymbol{N}^q$. The diffusive flux, $\widehat{\varepsilon\psi_{i,q}\frac{\partial\phi_i}{\partial x_j}n_j n_k N_k^q}$, is also required at a face where a discontinuity exists. However, unlike the compressive fluxes, an up-wind approach is not appropriate. A logical and simple implementation is to take the arithmetic mean of diffusive fluxes calculated on the left and right sides of the face, but this method does not take into consideration the discontinuity at the face and, therefore, is inconsistent [29]. A number of strategies have been proposed to discretize a diffusive flux in the context of DG, such as the local discontinuous Galerkin (LDG) method [30] and the reconstructed discontinuous Galerkin (RDG) approach [29]. The RDG method is straightforward to implement and is used here to define a unique flux value at the face that is consistent with the DG solution. To do this, a reconstructed function $R$ is introduced in the discontinuous Galerkin space that spans the two cells of interest. The function can be written as $R(\tilde{\boldsymbol{\chi}}, t) = \boldsymbol{r}\cdot\tilde{\boldsymbol{\phi}}$ where $\boldsymbol{r}(t)$ is a vector containing the weights and $\tilde{\boldsymbol{\phi}}(\tilde{\boldsymbol{\chi}})$ is a vector of the basis functions on a modified coordinate system $\tilde{\boldsymbol{\chi}}$. The modified coordinate system is local to the two cells that contain the cell face of interest and is defined such that $\tilde{\boldsymbol{\chi}} = [-1, 1]^d$ within the domain formed by the union of the two cells. To find the weights $\boldsymbol{r}$, the following set of $2P$ constraints are applied in a least squares sense:

$$\int_{\omega_{q^-}} \psi_{i,q^-}\phi_i\tilde{\phi}_s \partial V = \int_{\omega_{q^-}} r_i\tilde{\phi}_i\tilde{\phi}_s \partial V, \tag{26a}$$

$$\int_{\omega_{q^+}} \psi_{i,q^+}\phi_i\tilde{\phi}_s \partial V = \int_{\omega_{q^+}} r_i\tilde{\phi}_i\tilde{\phi}_s \partial V, \tag{26b}$$

for $s = 1, \ldots, P$, where $\omega_{q^-}$ and $\omega_{q^+}$ are the volumes of the cells to the left and right of the face, respectively. Luo et al. [29] suggested the properties of the reconstruction function can be improved by adding an extra term of order $O+1$ in the direction approximately normal to the interface. We add the term in the direction that has the largest component of the interface normal vector. For example, if the normal vector at the face of interest is predominantly in the $x$-direction, $\phi_{q,11} = \chi_{q,1}^3 - 3\chi_{q,1}/5$ would be added to the 10 basis shown in Eq. (18). This is possible because $2P$ ($P \geqslant 1$) constraints are available to find $P+1$ unknowns, $\boldsymbol{r}$, when the extra term is added. Using the reconstructed function, $R$, the diffusive flux is written as

$$\varepsilon\psi_{i,q}\widehat{\frac{\partial\phi_i}{\partial x_j}}n_j n_k N_k^q = \varepsilon r_i \frac{\partial\tilde{\phi}_i}{\partial x_j}(n_j n_k N_k^q)_{\gamma_q}. \tag{27}$$

Combining Eq. (24) with fluxes from Eqs. (25a), (25b), and (27), using the properties $\boldsymbol{\psi} = \boldsymbol{\psi}(t), \boldsymbol{\phi} = \boldsymbol{\phi}(\boldsymbol{\chi}(\boldsymbol{x})), \boldsymbol{r} = \boldsymbol{r}(t)$, and $\tilde{\boldsymbol{\phi}} = \tilde{\boldsymbol{\phi}}(\boldsymbol{\chi}(\boldsymbol{x}))$, and applying the property shown in Eq. (14) for orthogonal basis functions leads to

$$\begin{aligned} &\frac{\partial\psi_m}{\partial\tau}\int_{\omega_q}\phi_m^2\partial V - \psi_{i,q}n_j\int_{\omega_q}\phi_i\frac{\partial\phi_m}{\partial x_j}\partial V + (\psi_{i,q})_{\text{up}}\left(n_j N_j^q\right)_{\gamma_q}\oint_{\gamma_q}(\phi_i)_{\text{up}}\phi_m\partial S + \psi_{i,q}\psi_{k,q}n_j\int_{\omega_q}\phi_i\phi_k\frac{\partial\phi_m}{\partial x_j}\partial V \\ &- (\psi_{i,q}\psi_{k,q})_{\text{up}}\left(n_j N_j^q\right)_{\gamma_q}\oint_{\gamma_q}(\phi_i\phi_k)_{\text{up}}\phi_m\partial S = -\varepsilon\psi_{i,q}n_j n_k\int_{\omega_q}\frac{\partial\phi_i}{\partial x_j}\frac{\partial\phi_m}{\partial x_k}\partial V + \varepsilon r_i(n_j n_k N_k^q)_{\gamma_q}\oint_{\gamma_q}\frac{\partial\tilde{\phi}_i}{\partial x_j}\phi_m\partial S \end{aligned} \tag{28}$$

for $m = 1, \ldots, P$ and $q = 1, \ldots, Q$. Similar to the transport equation, the discretized reinitialization equation shown above can be easily updated after a time integration scheme is chosen. Also, all the integrals in the equation only depend on the basis functions and can be pre-computed suggesting that a quadrature-free approach should work. However, numerical experiments have shown that using a quadrature-free implementation of the compressive term is unstable. Details of the instability and our solution are described in Section 4.

## 4. Numerical stability

We found that using a quadrature-free approach to evaluate the integrals in Eq. (28) is unstable. The problem arises due to the compressive term which will amplify overshoots or undershoots of the conservative level set. We tried two approaches to address this problem. The first, referred to as the bounded flux approach, checks the boundedness of the con-

servative level set function and only use the compressive term in regions of the computational cell that are bounded. The second approach uses a minimum/maximum preserving (MMP) limiter to avoid overshoots and undershoots. Details of both approaches are provided below. From our test cases, we found that the bounded flux approach remains stable and does not significantly impact the solution. Contrarily, the MMP limiter significantly modifies the interface. As a result, only the bounded flux approach is used in our method and the MMP limiter is presented to provide insight into the effects of limiting on accuracy.

### 4.1. Bounded flux approach

This approach assesses the boundedness of the conservative level set in different regions of the computational cell and only applies the compressive term in regions that do not have excessive overshoots or undershoots. A Gauss quadrature of sufficient accuracy is introduced to evaluate each of the compressive terms that appear in Eq. (28). At each quadrature point, if $-\zeta \leqslant \Psi_h \leqslant 1+\zeta$ for some small $\zeta$, then this point is included in the integration; else, the level set is significantly overshooting or undershooting and this quadrature point and is not included in the integral. We use $\zeta = 10^{-5}$. This methodology is written explicitly below for each compressive term in Eq. (28). Starting with the first term, which can be written as

$$\psi_{i,q} n_j \int_{\omega_q} \phi_i \frac{\partial \phi_m}{\partial x_j} \partial V \approx \sum_{n=1}^{3N_{2O}} \omega'_n \psi_{i,q} \phi_i(\boldsymbol{x}_n) n_j \left.\frac{\partial \phi_m}{\partial x_j}\right|_{\boldsymbol{x}=\boldsymbol{x}_n} \tag{29}$$

where $\boldsymbol{x}_n$ is the location of the quadrature point, $N_{2O}$ is the smallest integer satisfying $2O < 2N_{2O} - 1$, and $\omega'_n$ is a modified quadrature weight that reflects the boundedness of the level set at point $n$ and is evaluated using

$$\omega'_n = \begin{cases} \omega_n & \text{if } -\zeta \leqslant \Psi(\boldsymbol{x}_n) \leqslant 1+\zeta, \\ 0 & \text{else}, \end{cases} \tag{30}$$

where $\omega_n$ is the standard Gauss quadrature weight. The other terms in Eq. (28) that are the discretized form of the compressive term can be written similarly using

$$(\psi_{i,q})_{\text{up}} \left(n_j N_j^q\right)_{\gamma_q} \oint_{\gamma_q} (\phi_i)_{\text{up}} \phi_m \partial S \approx \sum_{n=1}^{3N_{2O}} \omega'_n \left(\psi_{i,q} \phi_i(\boldsymbol{x}_n)\right)_{\text{up}} \left(n_j N_j^q\right)_{\gamma_q} \phi_m(\boldsymbol{x}_n), \tag{31}$$

$$\psi_{i,q} \psi_{k,q} n_j \int_{\omega_q} \phi_i \phi_k \frac{\partial \phi_m}{\partial x_j} \partial V \approx \sum_{n=1}^{3N_{3O}} \omega'_n \psi_{i,q} \phi_i(\boldsymbol{x}_n) \psi_{k,q} \phi_k(\boldsymbol{x}_n) n_j \left.\frac{\partial \phi_m}{\partial x_j}\right|_{\boldsymbol{x}=\boldsymbol{x}_n}, \tag{32}$$

$$(\psi_{i,q}\psi_{k,q})_{\text{up}} \left(n_j N_j^q\right)_{\gamma_q} \oint_{\gamma_q} (\phi_i \phi_k)_{\text{up}} \phi_m \partial S \approx \sum_{n=1}^{3N_{3O}} \omega'_n \left(\psi_{i,q} \phi_i(\boldsymbol{x}_n) \psi_{k,q} \phi_k(\boldsymbol{x}_n)\right)_{\text{up}} \left(n_j N_j^q\right)_{\gamma_q} \phi_m(\boldsymbol{x}_n), \tag{33}$$

where $N_{3O}$ is the smallest integer satisfying $3O < 2N_{3O} - 1$. Up-winding in the calculation of the flux in the surface integrals is performed at each quadrature point using Eqs. (25a) and (25b).

This approach has been found to be stable for all of the test cases studied.

### 4.2. Minimum/maximum preserving limiter

In an effort to improve the boundedness of the level set function the MMP limiter of Zhang and Shu [21] was implemented. The limiter was designed to maintain the order of the DG scheme while modifying the formulation such that the function stays within the interval $[0,1]$. Testing of the limiter highlighted that while the MMP limiter improves the boundedness of the level set function, it introduces an excessive amount of numerical diffusion, thereby impeding the mass conservation properties of the scheme. As a result, the MMP limiter is not used in the test cases provided in this paper and is included here to provide insight into the effect of limiting overshoots and undershoots on the conservation properties of the scheme.

Implementation of the limiter is straight forward for the transport equation and requires replacing the DG representation of the level set within cell $q$, $\Psi_{h,q}$, with a modified function $\widetilde{\Psi}_{h,q}$ given by

$$\widetilde{\Psi}_{h,q} = \Theta_q \left(\Psi_{h,q} - \overline{\Psi}_{h,q}\right) + \overline{\Psi}_{h,q}, \tag{34}$$

where $\overline{\Psi}_{h,q}$ is the mean value of the DG representation of the level set in the $q$th cell. $\Theta$ is a measure for how close the level set function within the cell is to the interval bounds and is defined to be

$$\Theta_q = \min\left\{ \left|\frac{m - \overline{\Psi}_{h,q}}{m_q - \overline{\Psi}_{h,q}}\right|, \left|\frac{M - \overline{\Psi}_{h,q}}{M_q - \overline{\Psi}_{h,q}}\right|, 1 \right\}, \tag{35}$$

where $m = 0$ is the lower bound, $M = 1$ is the upper bound, and $m_q$ and $M_q$ are the minimum and maximum of the DG approximation of the level set within the $q$th cell, respectively. To avoid finding the minimum and maximum of a high order polynomial, $m_q$ and $M_q$ can be approximated by finding the minimum and maximum of the level set function at quadrature points. We use an $N$-point Gauss–Lobatto quadrature rule in each direction within the grid cell of interest. $N$ is chosen to be the smallest integer satisfying $2O < 2N - 3$, which is the quadrature needed to exactly integrate a polynomial of order $2O$.

Substituting the modified conservative level set, given by Eq. (34), into the DG discretized level set transport equation, Eq. (22), results in

$$\frac{\partial \psi_{m,q}}{\partial t}\int_{\omega_q}\phi_m^2\partial V - \Theta_q\psi_{i,q}u_j\int_{\omega_q}\phi_i\frac{\partial\phi_m}{\partial x_j}\partial V + (\Theta_q-1)\overline{\Psi}_{h,q}u_j\int_{\omega_q}\frac{\partial\phi_m}{\partial x_j}\partial V + (\Theta_q)_{\text{up}}\left(u_jN_j^q\right)_{\gamma_q}(\psi_{i,q})_{\text{up}}\oint_{\gamma_q}(\phi_i)_{\text{up}}\phi_m\partial S - \left((\Theta_q)_{\text{up}}-1\right)(u_jN_j)_{\gamma_q}\left(\overline{\Psi}_{h,q}\right)_{\text{up}}\oint_{\gamma_q}\phi_m\partial S = 0 \tag{36}$$

for $m = 1,\ldots,P$ and $q = 1,\ldots,Q$, where $(\Theta_q)_{\text{up}}$ and $(\overline{\Psi}_q)_{\text{up}}$ are up-winded using the criteria defined in Eq. (20).

When $\Theta = 1$, which occurs when the minimum and maximum of the grid cell are contained within the interval $[0,1]$, the original DG scheme is used. However, when $\Theta = 0$ the flux becomes based on only the mean of the DG approximation instead of the entire DG polynomial. A value of $\Theta$ between 0 and 1 results in a weighted combination of the fluxes based on the mean and on the full DG representation.

Applying the MMP limiter to the reinitialization equation is more difficult because of the non-linear compressive term. Note that we do not apply the MMP limiter to the diffusion term since it should not contribute to creating an unbounded function. Our approach for the non-linear compressive term is to apply the MMP limiter to the entire portion of the non-linear term that depends on $\Psi_h$, resulting in the modified function

$$\widetilde{\Psi_{h,q}-\Psi_{h,q}^2} = \Theta_q'\left(\left(\Psi_{h,q}-\Psi_{h,q}^2\right)-\left(\overline{\Psi}_{h,q}-\overline{\Psi}_{h,q}^2\right)\right)+\left(\overline{\Psi}_{h,q}-\overline{\Psi}_{h,q}^2\right), \tag{37}$$

with

$$\Theta_q' = \min\left\{\left|\frac{(m-m^2)-\left(\overline{\Psi}_{h,q}-\overline{\Psi}_{h,q}^2\right)}{\left(m_q-m_q^2\right)-\left(\overline{\Psi}_{h,q}-\overline{\Psi}_{h,q}^2\right)}\right|,\left|\frac{M-M^2-\left(\overline{\Psi}_{h,q}-\overline{\Psi}_{h,q}^2\right)}{\left(M_q-M_q^2\right)-\left(\overline{\Psi}_{h,q}-\overline{\Psi}_{h,q}^2\right)}\right|,1\right\}. \tag{38}$$

Substituting Eq. (37) into the DG discretized reinitialization equation, Eq. (28), results in

$$\begin{aligned}&\frac{\partial\psi_m}{\partial\tau}\int_{\omega_q}\phi_m^2\partial V - \Theta_q'\psi_{i,q}n_j\int_{\omega_q}\phi_i\frac{\partial\phi_m}{\partial x_j}\partial V + \Theta_q'(\psi_{i,q})_{\text{up}}\left(n_jN_j^q\right)_{\gamma_q}\oint_{\gamma_q}(\phi_i)_{\text{up}}\phi_m\partial S + \Theta_q'\psi_{i,q}\psi_{k,q}n_j\int_{\omega_q}\phi_i\phi_k\frac{\partial\phi_m}{\partial x_j}\partial V\\ &- \Theta_q'(\psi_{i,q}\psi_{k,q})_{\text{up}}\left(n_jN_j^q\right)_{\gamma_q}\oint_{\gamma_q}(\phi_i\phi_k)_{\text{up}}\phi_m\partial S - \left(\Theta_q'-1\right)\left(\overline{\Psi}_{h,q}-\overline{\Psi}_{h,q}^2\right)\int_{\omega_q}\frac{\partial\phi_m}{\partial x_j}\partial V\\ &+\left(\left(\Theta_q'\right)_{\text{up}}-1\right)\left(\left(\overline{\Psi}_{h,q}\right)_{\text{up}}-\left(\overline{\Psi}_{h,q}\right)_{\text{up}}^2\right)\oint_{\gamma_q}\phi_m\partial S\\ &= -\varepsilon\psi_{i,q}n_jn_k\int_{\omega_q}\frac{\partial\phi_i}{\partial x_j}\frac{\partial\phi_m}{\partial x_k}\partial V + \varepsilon r_i\left(n_jn_kN_k^q\right)_{\gamma_q}\oint_{\gamma_q}\frac{\partial\tilde{\phi}_i}{\partial x_j}\phi_m\partial S\end{aligned} \tag{39}$$

for $m = 1,\ldots,P$ and $q = 1,\ldots,Q$. The MMP limiter parameters appearing in the flux terms, $\left(\Theta_q'\right)_{\text{up}}$ and $\left(\overline{\Psi}_{h,q}\right)_{\text{up}}$, are up-winded using the criteria defined in Eqs. (25a) and (25b).

### 4.3. Comparison of stabilization strategies

The bounded flux strategy was compared to the MMP limiter using a two-dimensional deformation test case that assesses the performance of the interface tracking scheme by stretching a two-dimensional disk in a vortex and then reversing the flow. The final shape of the disk should match the initial condition. Details of the test case are provided in Section 6.2. Fig. 2 shows the results in the middle of the simulation when maximum deformation is obtained and at the end of the simulation where the solution should match the initial condition. Qualitative comparison of the images in Fig. 2 reveals that the MMP limiter modifies the solution significantly more than using the bounded flux approach. Comparing the interface location in the middle of the simulation shows the MMP limiter breaks the liquid into more droplets. At the end of the simulation, the MMP solution has a larger deviation from the exact solution than the bounded flux approach.

Quantitative comparison of the simulation with the bounded flux approach and the MMP limiter is provided in Table 1. The results show that the MMP limiter does produce smaller overshoots and undershoots as expected. However, the effect on the overall accuracy of the scheme is found to be large and for this case the maximum mass error is 23.6% when using the MMP limiter compared to 1.96% when the bounded flux approach is used. The results show that while the limiter does

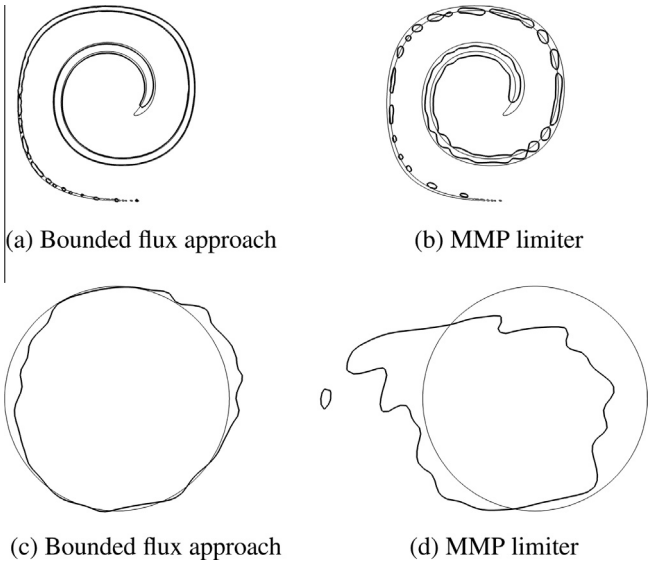



**Fig. 2.** Interface location of deformation test case with the bounded flux approach and the MMP limiter. (a) and (b) show results obtained at maximum deformation, i.e., $t = 4$. (c) and (d) show results obtained at the end of the simulation, i.e., $t = 8$. The exact solution is shown with a thin line for reference. For both cases a $128^2$ mesh, second order polynomials in the DG discretization, and a reinitialization factor $F = 0.5$ were used.

**Table 1**
Undershoot, overshoot, and mass errors for deformation test case with the bounded flux approach and the MMP limiter.

| | Maximum undershoot | Maximum overshoot | Maximum mass error (%) |
|---|---|---|---|
| Bounded flux approach | −0.0006 | 0.029 | 1.96 |
| MMP Limiter | 0.0000 | 0.005 | 23.6 |

achieve the goal of reducing overshoots and undershoots the method represents a significant hinderance to accuracy. As a result, the bounded flux approach will be used for the remainder of the test cases shown in this paper.

## 5. Level set time advancement

As described previously, the level set function is transported with the velocity field and then reinitialized using Eqs. (22) and (28), respectively. Temporal discretization is performed using a total variation diminishing third order Runge–Kutta (TVD-RK3) scheme. This scheme has been shown by Cockburn and Shu [28] to be stable to at least eighth order polynomial basis functions. Using the explicit TVD-RK3 method maintains the local nature of the scheme and does not hinder the highly scalable properties of DG.

The procedure to update the level set in time requires two steps. The first step transports the level set using Eq. (22) for a total time $\Delta t$, which is equal to the time-step used by the flow solver. Then, the reinitialization equation, Eq. (28), is solved for an amount of pseudo-time $\Delta\tau$ to maintain the shape of the level set profile.

While reinitialization is important to the conservation properties of the scheme it also introduces errors into the solution. The errors can be limited by choosing to reinitialize enough to maintain the proper profile shape while avoiding unneeded reinitialization. To control the amount of reinitialization, we introduce the reinitialization factor $F$ that relates the total reinitialization pseudo-time-step $\Delta\tau$ to the flow solver time-step $\Delta t$ using

$$\Delta\tau = F\Delta t \max(|\boldsymbol{u}\cdot\boldsymbol{n}|). \tag{40}$$

$F = 0$ correspond to no reinitialization and $F = 1$ equates to an amount of reinitialization that can move the level set the same distance it was moved by the transport step. The appropriate value for $F$ might depend on the nature of the test case. In some flows, the transport step does not change the level set profile and very little or no reinitialization is needed; this type of flow includes uniform flow and Zalesak's disk test case that uses solid body rotation. Other flows, such as stagnation flows and the vortex used in the deformation test case have complex flow fields that significantly deform the level set profile. Such flows require more reinitialization in order to control mass loss. Zalesak's disk and the deformation test case are discussed in Section 6, where the effect of $F$ on the solution is analyzed. For most of the applications considered here, a value of $F = 0.5$ seem to provide an appropriate amount of reinitialization.

The time discretization of Eqs. (22) and (28) need to respect the Courant–Friedrichs–Lewy (CFL) constraint, which is approximately

$$\mathrm{CFL} \leqslant \frac{1}{2O+1} \tag{41}$$

for the DG discretization, where $O$ is the highest order of the polynomials used as the basis functions [18]. This CFL constraint may be more restrictive than the flow solver CFL constraint. In-order to not hinder the efficiency of the rest of the computational code, sub-stepping the level set transport and reinitialization equations is used.

The transport equation includes a convective term with corresponding CFL number

$$\mathrm{CFL}_{\mathrm{trans}} = \frac{|\boldsymbol{u}|_{\max}\Delta t_{\mathrm{step}}}{h}, \tag{42}$$

where $|\boldsymbol{u}|_{\max}$ is the maximum of the velocity magnitude within the domain and $h$ is the smallest characteristic mesh size taken to be $\min(\Delta \boldsymbol{x})$ for our Cartesian mesh. Plugging Eq. (42) into Eq. (41) provides an upper bound for the time-step size, $\Delta t_{\mathrm{step}}$. If $\Delta t_{\mathrm{step}}$ is smaller than the flow solver time step $\Delta t$ sub-steps are utilized.

The reinitialization equation contains both a compressive and a diffusive term resulting in the CFL condition

$$\mathrm{CFL}_{\mathrm{reinit}} = \max\left(\frac{\Delta\tau_{\mathrm{step}}}{h}, \frac{4\varepsilon\Delta\tau_{\mathrm{step}}}{h^2}\right). \tag{43}$$

The first term is the compressive CFL. The second term is the diffusive CFL and contains $\varepsilon$ which was defined previously and sets the thickness of the hyperbolic tangent profile. Eq. (43) is combined with Eq. (41) to find an upper bound on the reinitialization pseudo-time-step size $\Delta\tau_{\mathrm{step}}$, which if less than total reinitialization pseudo-time-step $\Delta\tau$, sub-stepping is used for the reinitialization equation.

For completeness, the procedure used to update the level set in time is:

1. The level set is transported using Eq. (22) for a total time of $\Delta t$, where $\Delta t$ is the time-step of the flow solver. Sub-steps are employed if $\Delta t_{\mathrm{step}}$ given by the CFL constraint, Eq. (42), is smaller than $\Delta t$.
2. Interface normal vectors and curvature are calculated using the procedure outlined in Section 7.
3. The level set is reinitialized using Eq. (28) by a total amount of pseudo-time $\Delta\tau$, which is calculated using Eq. (40) for a chosen value of $F$. If $\Delta\tau$ is larger than the maximum reinitialization time-step $\Delta\tau_{\mathrm{step}}$ calculated using the CFL constraint in Eq. (43), then multiple sub-steps are used.

## 6. Validation cases

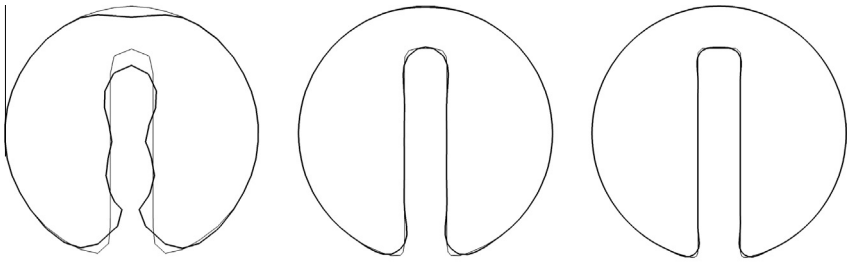
### 6.1. Zalesak's disk

The first validation test case is known as Zalesak's disk [31] and tests the ability of the DG-CLS scheme to transport a complex geometry with sharp corners. The test consists of solid body rotation of a notched disk with radius 0.15, notch width of 0.05, and center at $(x, y) = (0, 0.25)$ within a square domain of size $[-0.5, 0.5]^2$. The notched disk is subjected to rotation using the two-dimensional velocity field,

$$u = -2\pi y, \tag{44a}$$

$$v = +2\pi x. \tag{44b}$$

For this test, the disk's shape should remain unchanged. Fig. 3 shows how mesh refinement affects the final shape of the notched disk. Meshes consisting of $50^2$, $100^2$, and $200^2$ cells were tested. For reference, the $50^2$ mesh has only two grid cells



(a) $50^2$ mesh (b) $100^2$ mesh (c) $200^2$ mesh

**Fig. 3.** The calculated solutions of Zalesak's disk after one full rotation using various meshes. Second order polynomials and reinitialization factor of $F = 0.5$ were used for all three cases. Solution is shown with the thick line and the exact solution, shown on the same mesh, is given with the thin line.

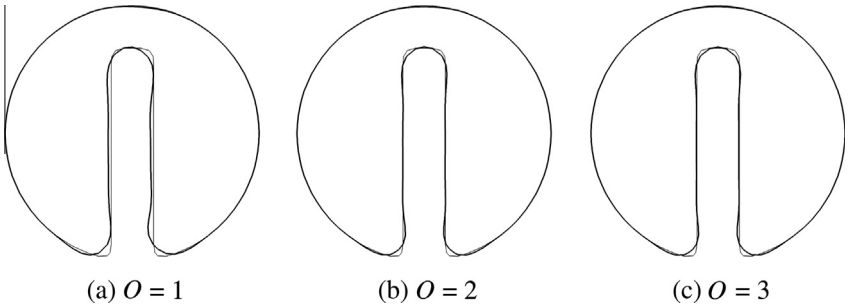



**Fig. 4.** The calculated solutions of Zalesak's disk after one full rotation using various orders of polynomials in the DG representation of the level set. A $100^2$ mesh and reinitialization factor of $F = 0.5$ were used for all three cases. Solution is shown with the thick line and the exact solution is given with the thin line.

across the notch, yet the notch is preserved after a rotation. When the mesh is refined to $100^2$ and $200^2$ cells, we find very good results after the disk has been rotated.

Next, the effect of the order of the polynomials used in the DG scheme to represent the level set was studied. Fig. 4 shows results for polynomials with orders $O = 1$, $O = 2$, and $O = 3$. The magnitude of the error decreases for $O = 2$ and $O = 3$. However, the difference between the solutions obtained using second and third order polynomials was relatively small suggesting other errors, such as errors from the interface normal calculation, are dominant. Furthermore, the differences indicate that running a simulation with second order basis functions may be a good compromise between accuracy and cost.

Fig. 5 shows results for different values of the reinitialization factor $F$, which characterizes the amount of reinitialization. The smallest errors are obtained when no reinitialization is performed ($F = 0$). This result is expected for Zalesak's disk because the velocity is prescribed to produce solid body rotation of the notched circle, which should not induce kinematic deformation of the level set profile. As more reinitialization is performed, additional errors are introduced and the final shape of the notched circle degrades slightly. The case with no reinitialization, $F = 0$, was tested further by increasing the number of rotations of the notched circle from 1 to 50 and reducing the mesh from $100^2$ to $50^2$. Results are in Fig. 6 show that even after 50 rotations and a very coarse mesh, the DG transport maintains the notched circle very well.

### 6.2. Two-dimensional deformation

The two-dimensional deformation test case consists of the stretching and un-stretching of a disk in a vortex. The simulation is initialized with a two-dimensional disk of diameter 0.3 centered at $(x, y) = (0, 0.25)$ within a unit square domain, $[-0.5, 0.5]^2$. The disk is stretched by the velocity field until time is equal to four, $t = 4$; then, the velocity field reverses for another four time units and the disk should return to its initial state. The velocity field used to achieve the stretching and un-stretching is

$$u = -2\sin^2(\pi x)\sin(\pi y)\cos(\pi y)\cos(\pi t/8), \tag{45a}$$

$$v = +2\sin^2(\pi y)\sin(\pi x)\cos(\pi x)\cos(\pi t/8). \tag{45b}$$

Fig. 7 shows snapshots of the progression from initial state ($t = 0$), to the fully stretched state ($t = 4$), and to the final state ($t = 8$). When the disk is in the fully stretched state, the tail will drop below the mesh resolution, which would lead to mass loss with a classical signed distance level set method. Because the conservative level set is designed to limit mass loss, the

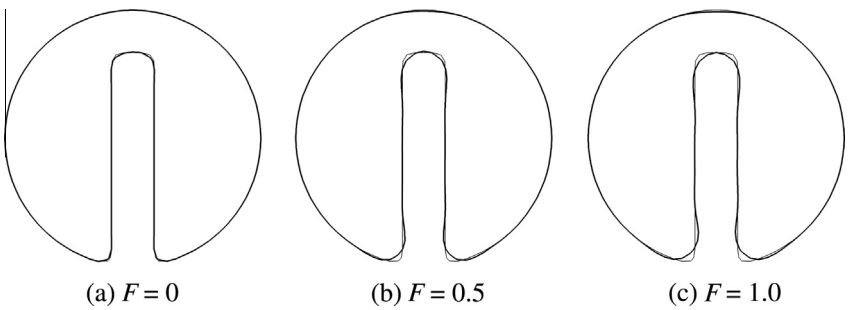



**Fig. 5.** The calculated solutions of Zalesak's disk after one full rotation different amounts of reinitialization. A $100^2$ mesh and second order polynomials were used for all three cases. Solution is shown with the thick line and the exact solution is given with the thin line.

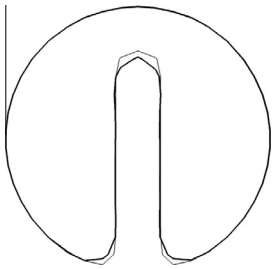


**Fig. 6.** Zalesak's disk after 50 rotations. A $50^2$ mesh, second order polynomials, and reinitialization factor $F = 0$ were used. Solution is shown with the thick line and the exact solution is given with the thin line.

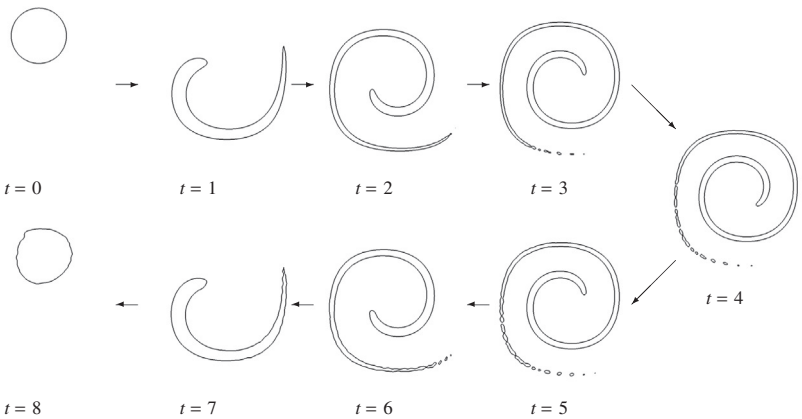



**Fig. 7.** Snapshots of two-dimensional deformation test case at various times. Interface is shown as a function of time using a $128^2$ mesh, second order polynomial basis functions, and a reinitialization factor of $F = 0.5$.

unresolvable tail breaks into "droplets" resolvable on the mesh. This phenomenon is clearly visible in Fig. 7 at $t = 3, 4$, and 5 where the tail has been replaced by droplets. A consequence of the mass-conserving scheme deforming the tail is that the final interface location may not match the expected exact solution.

To investigate the effect of reinitialization on the solution, simulations were conducted with varying values of the reinitialization factor $F$. The test used a mesh with $128^2$ cells and second order polynomial basis functions. Results are provided in Fig. 8 and show that as more reinitialization is performed, the tail breaks into more droplets. However, for the case with no reinitialization, the tail becomes under-resolved and the volume within this portion of the tail is simply lost.

To study the convergence properties of the scheme, the mesh was refined and the order of the polynomial basis was varied. Fig. 9 shows how the solution changes as the mesh is refined from $64^2$ to $256^2$ mesh points. When a finer mesh was used, the tail is almost totally captured; however, with the coarse mesh, the tail becomes under-resolved and shows a higher tendency to break into droplets. Similar results are shown in Fig. 10 wherein the effect of the DG order was analyzed. At the fully stretched state, the higher order polynomials have the ability to capture more of the tail region, leading to fewer droplets.

Next, we turn our attention to the conservation properties of the scheme. Here we refer loosely to the volume (in 3D) or area (in 2D) contained under the $\Psi = 0.5$ isosurface as "mass". Mass is calculated using the methodology of van der Pijl et al. [32]. Fig. 11(a) shows non-dimensionalized mass as a function of time for various amounts of reinitialization. The figure shows that when reinitialization is used the largest mass error occurs around maximum stretching. This apparent mass loss is due to the difference between the physical liquid volume fraction and our approximation of the quantity using the conservative level set. Note that little difference can be observed between the $F = 0.5$ and the $F = 1$ cases, which indicates that using a reinitialization factor of $F = 0.5$ may provide enough reinitialization to limit significant mass losses. An important quantity to consider is the mass error at the end of the simulation and indicates the total mass lost throughout the stretching and un-stretching of the disk. Table 2(a) provides the percentage of mass lost throughout the simulation for various amounts of reinitialization. As expected, the largest error occurs when no reinitialization is performed. As the amount of reinitialization is increased, to $F = 0.5$ and $F = 1$, the mass error is reduced.

The effect of mesh refinement on the mass conservation properties of the scheme was studied and the results are provided in Fig. 11(b) and Table 2(b). As expected, even on the coarsest grid the amount of mass that was lost throughout

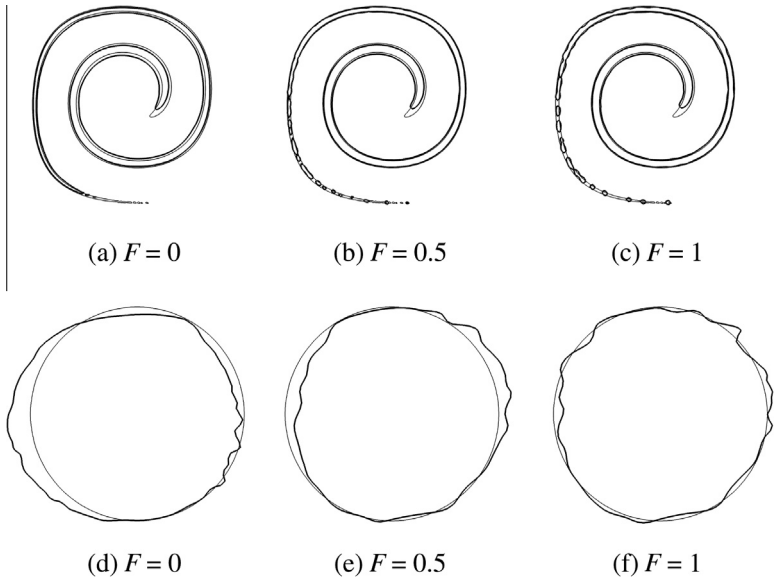



**Fig. 8.** Interface location of deformation test case with various amounts of reinitialization. The reinitialization factor, $F$, was varied from 0 to 1. Figures (a)–(c) show results obtained at maximum deformation, i.e., $t = 4$. Figures (d)–(f) show results obtained at the end of the simulation, i.e., $t = 8$. The exact solution is shown with a thin line for reference. For all cases a $128^2$ mesh and second order polynomials in the DG discretization were used.

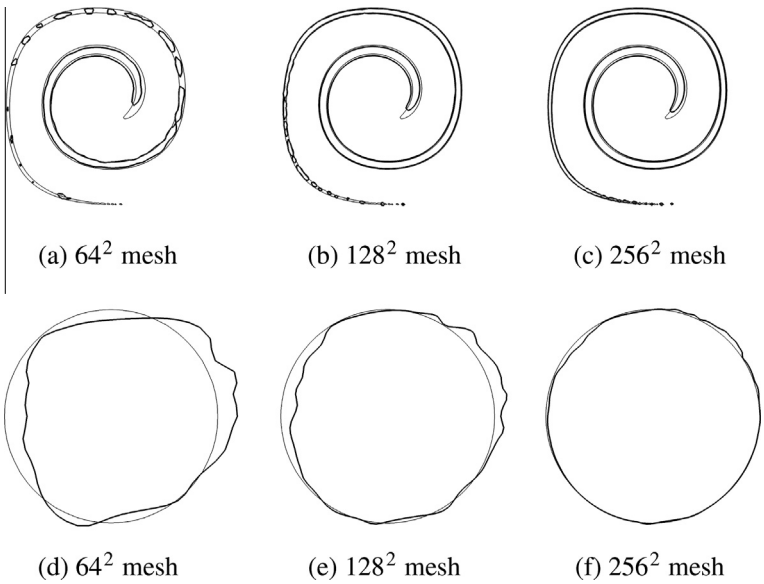



**Fig. 9.** Interface location of deformation test case on different meshes, namely: $64^2$, $128^2$, and $256^2$. Figures (a)–(c) show results obtained at maximum deformation, i.e., $t = 4$. Figures (d)–(f) show results obtained at the end of the simulation, i.e., $t = 8$. The exact solution is shown with a thin line for reference. For all cases second order polynomials were used in the DG discretization and the reinitialization factor was set to $F = 0.5$.

the simulation was very small. On the finest grid the amount of the tail region that fell below mesh resolution and was transferred into resolvable droplets was the least and therefore the conservation error around $t = 4$ is the smallest. For the coarser meshes, more of the tail fell below mesh resolution and therefore more of the liquid was moved into droplets resulting in large curvature changes and the significant apparent loss of mass in the middle of the simulation. The mass loss error at the end of the simulation shows that for all of the meshes the amount of mass loss is small (less than 1%).

Fig. 11(c) and Table 2(c) shows the results from simulations with different DG orders. The differences in the results are very small indicating the order of the DG polynomials does not have a large effect on the mass conservation properties for this test case.

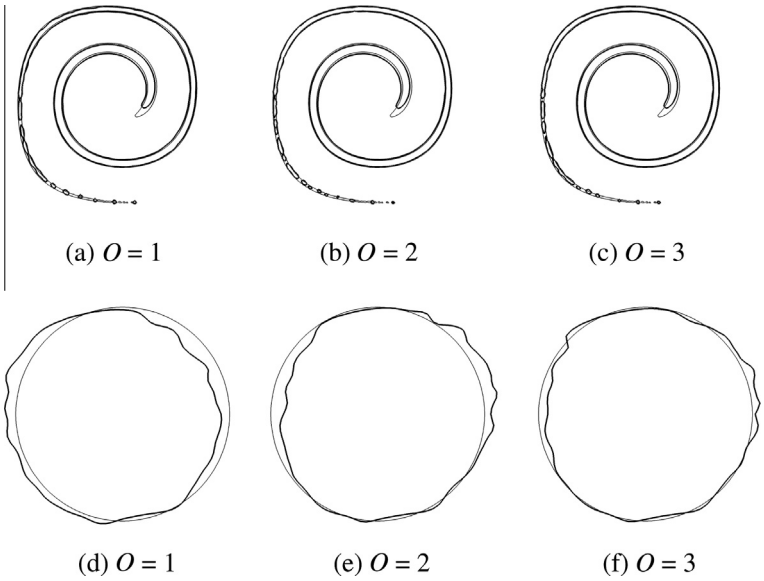
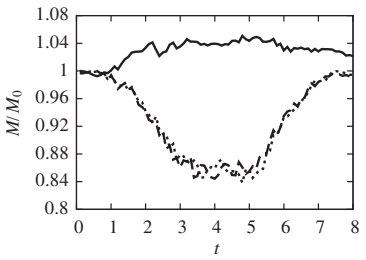
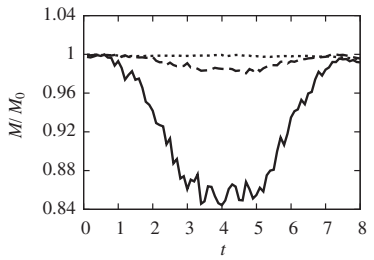


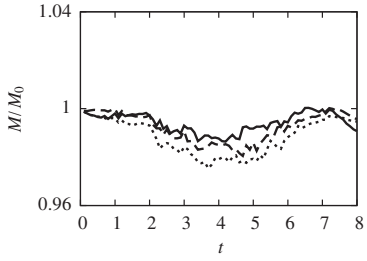

**Fig. 10.** Interface location of deformation test case using different polynomials orders in the DG discretization of the level set. Figures (a)–(c) show results obtained at maximum deformation, i.e., $t = 4$. Figures (d)–(f) show results obtained at the end of the simulation, i.e., $t = 8$. The exact solution is shown with a thin line for reference. For all cases a mesh with $128^2$ points was used and the reinitialization factor was set to $F = 0.5$.




(a) Varying reinitialization factor: $F = 0$ (solid), $F = 0.5$ (dashed), and $F = 1$ (dotted), $64^2$ mesh, DG order=2




(b) Varying mesh: $64^2$(solid), $128^2$ (dashed), and $256^2$ (dotted), $F = 0.5$, DG order=2




(c) Varying DG Order: $O = 1$ (solid), $O = 2$ (dashed), and $O = 3$ (dotted), $F = 0.5$, $128^2$ mesh

**Fig. 11.** Mass of liquid normalized by initial mass plotted versus time for two-dimensional deformation test. Figure (a) shows results when the amount of reinitialization is varied. The effect of different meshes was plotted in Figure (b). The DG order was changed and the results are shown in Figure (c).

### 6.3. Scalability of DG-CLS scheme

The DG-CLS scheme only uses information from cells that share a face to construct fluxes and update the solution, resulting in the smallest possible computational stencil. A small stencil reduces the amount of data that needs to be communicated between processors, which speeds up the simulation and reduces parallel overhead. Therefore, the DG-CLS scheme should have excellent parallel performance. To test this claim, a series of simulations were conducted with varying number of cores ranging from 16 through 2048. The number of grid cells per core was held constant at $256^2$ cells/core. All of the simulations used the two-dimensional deformation test case with second order DG polynomials and a reinitialization factor $F = 0.5$.

**Table 2**
Mass loss error at the end of the simulation normalized by the initial mass. The effect of the reinitialization factor, mesh, and DG order are shown in (a)–(c), respectively.

| (a) Varying reinit. factor, $64^2$ mesh, $O = 2$ | | (b) Varying mesh, $F = 0.5$, $O = 2$ | | (c) Varying DG order, $F = 0.5, 128^2$ mesh | |
|---|---|---|---|---|---|
| Reinit. factor | $M_{final}/M_0$ | Mesh | $M_{final}/M_0$ | DG order | $M_{final}/M_0$ |
| 0.0 | 2.12% | $64^2$ | 0.89% | 1 | 0.94% |
| 0.5 | 0.89% | $128^2$ | 0.41% | 2 | 0.41% |
| 1.0 | 0.62% | $256^2$ | 0.46% | 3 | 0.54% |

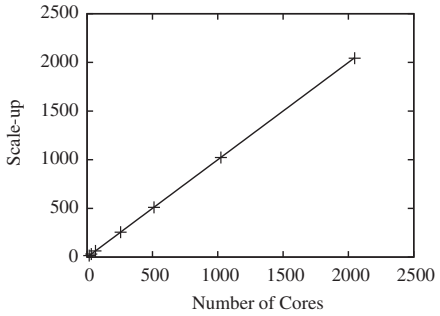



**Fig. 12.** Scale-up of the DG-CLS scheme. Linear scaling shown with solid line. Results obtained by simulating the deformation test case on different meshes shown with + symbols.

Fig. 12 shows results from the study. The plot compares the scale-up of the DG-CLS scheme to the theoretical case with perfect scale-up. As expected the DG-CLS scheme obtains almost ideal scale-up.

## 7. Interface normal and curvature

### 7.1. Methodology

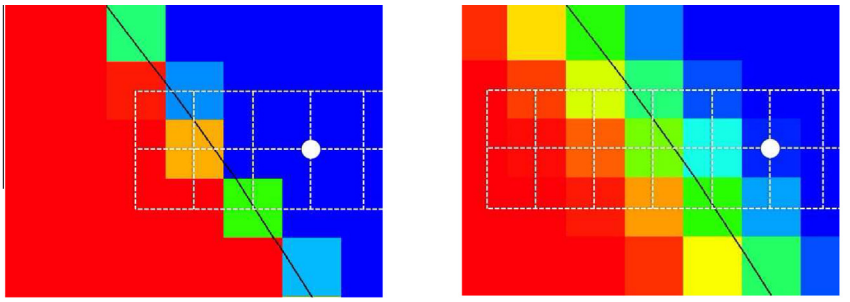
The normal vector to the interface is used for the reinitialization of the level set and must be calculated accurately. The curvature is used to determine the pressure jump that results from surface tension in the solution of the Navier–Stokes equations and has a direct effect on the solution. Therefore, having the normal vector and curvature converge under mesh refinement is necessary for mesh independence studies and predictive simulations. Convergence is difficult to obtain because as the mesh is refined, the thickness of the conservative level set function is also reduced so that the number of grid cells across the hyperbolic tangent profile is fixed. Desjardins et al. [15] was confronted with this problem when developing the ACLS method and applied a least squared approach proposed by Marchandise et al. [20] for the signed distance level set. The scheme calculates the normal and curvature from a least squares polynomial fit of a reconstructed signed distance level set over the cell of interest and its 26 nearest neighbors. The signed distance level set is used because of the presences of oscillations on the conservative level set. We use a parallel fast marching method [33] to reconstruct the distance level set from the conservative level set.

The least squares method has been shown to converge with second order for the normal and first order for the curvature when applied to the conservative level set [15]. While the convergence of the normal vector is acceptable, a first order curvature is sub-optimal. Therefore, we use the least squares approach to calculate the interface normal but introduce another strategy to compute interface curvature.

The proposed method to calculate interface curvature is to use a height function technique popular in VOF schemes [22]. The method has been shown, within VOF schemes, to calculate a curvature that converges with second order accuracy.

By integrating the volume fraction in a pseudo-normal direction, a liquid height is formed in the cell of interest and the neighboring cells. The curvature is then calculated using finite difference operators on the heights. The pseudo-normal direction is defined as the direction $x$, $y$, or $z$ with the largest component of the interface normal vector. The method assuming the pseudo-normal direction was found to be the $x$-direction in the cell with coordinates $i$, $j$, $k$ is detailed below. In three dimensions, a stencil of $7 \times 3 \times 3$ cells is used with seven cells in the $x$-direction and three cells in the $y$ and $z$ directions [22]. Seven cells are chosen for accuracy when the interface is at an angle, as shown in Fig. 13(a).

The nine liquid heights are calculated over a $3 \times 3$ mesh that is perpendicular to the pseudo-normal vector by integrating in the $x$-direction at each location using



(a) Volume fraction and stencil used in VOF height function formulation

(b) First DG degree of freedom and stencil used in DG height function formulation

**Fig. 13.** Stencils used to compute curvature at cell indicated with white circle. Stencil for VOF is shown by the white dotted lines in (a) and is a total of $7 \times 3$ ($7 \times 3 \times 3$ in 3D). Correspondingly, the stencil used for DG is shown in (b) and has a $11 \times 3$ stencil ($11 \times 3 \times 3$ in 3D).

$$H_{j'k'} = \sum_{i'=i-3}^{i+3} f_{i'j'k'}\Delta x \quad \text{for} \quad \begin{cases} j' = j-1, j, j+1, \\ k' = k-1, k, k+1, \end{cases} \tag{46}$$

where $f_{i'j'k'}$ is the volume fraction in cell $i'$, $j'$, $k'$ and $\Delta x$ is the width of the cell in the pseudo-normal direction. Using the heights, the curvature is calculated using second order finite difference operators that can be written as

$$\kappa = \frac{H_{yy} + H_{zz} + H_{yy}H_z^2 + H_{zz}H_y^2 - 2H_{yz}H_yH_z}{\left(1 + H_y^2 + H_z^2\right)^{3/2}} \left( \frac{\partial f_{ijk}/\partial x}{|\partial f_{ijk}/\partial x|} \right), \tag{47}$$

with

$$H_y = \frac{H_{j+1,k} - H_{j-1,k}}{2\Delta y}, \tag{48a}$$

$$H_z = \frac{H_{j,k+1} - H_{j,k-1}}{2\Delta z}, \tag{48b}$$

$$H_{yy} = \frac{H_{j+1,k} - 2H_{jk} + H_{j-1,k}}{\Delta y^2}, \tag{48c}$$

$$H_{zz} = \frac{H_{j,k+1} - 2H_{jk} + H_{j,k-1}}{\Delta z^2}, \quad \text{and} \tag{48d}$$

$$H_{yz} = \frac{H_{j+1,k+1} - H_{j+1,k-1} - H_{j-1,k+1} + H_{j-1,k-1}}{2\Delta x\ 2\Delta y}. \tag{48e}$$

To extend this method to the conservative level set, we modify the way the heights, $H$, are calculated. Instead of integrating volume fraction, the conservative level set, $\Psi_h$, is integrated. For example, if the pseudo-normal direction is still assumed to be in the $x$-direction, then

$$H_{j'k'} = \sum_{i'=i-\frac{S-1}{2}}^{i+\frac{S-1}{2}} f^{\mathrm{DG}}_{i'j'k'}\Delta x \quad \text{for} \quad \begin{cases} j' = j-1, j, j+1, \\ k' = k-1, k, k+1, \end{cases} \tag{49}$$

with

$$f^{\mathrm{DG}}_{i'j'k'} = \int_{\omega_{q'}} \Psi_{h,q'}\partial V = \psi_{i,q'} \int_{\omega_{q'}} \phi_i \partial V, \tag{50}$$

where $q'$ is the index for the $i'$, $j'$, $k'$ cell. For Legendre polynomials, Eq. (50) reduces to, $f^{\mathrm{DG}}_{i'j'k'} = \psi_{1,q'}$.

The stencil size has also been changed from $7 \times 3 \times 3$ to $S \times 3 \times 3$ where $S$ is chosen to achieve high accuracy by capturing the width of the interface within the stencil and should be based on the thickness of the interface. For example, an interface thickness corresponding to $\varepsilon = 0.5\Delta x$ is captured on roughly four cells and results in $S = 11$ or a $11 \times 3 \times 3$ stencil as shown by Fig. 13. For other values of $\varepsilon$, the stencil can be determined by noting the profile thickness is approximately $8\varepsilon/\Delta x$ cells, which was determined using numerical tests. Therefore, the stencil should be seven cells plus the thickness of the profile or $S \approx 7 + 8\varepsilon/\Delta x$.

The large stencil leads to numerical difficulties in two circumstances that must be dealt with if accuracy and robustness are to be maintained. The first scenario manifests when two interfaces approach each other. This can occur when two liquid structures come close together or when a liquid entity becomes thin. To avoid mixing information from the two interfaces, the stencil size should be reduced in the pseudo-normal direction such that the other interface does not influence the calculation of the heights. The second circumstance occurs when the structure has a large curvature, and the interface is at an angle with respect to the coordinate system. Defining a stencil that is aligned with one of the coordinate axis and captures the entire profile in all of the nine heights can be impossible to construct. Our solution to this problem is to give up on the height function approach in the problematic cell and revert back to the least squares approach of Marchandise et al. [20] and Desjardins et al. [15] mentioned previously. Note that while the least squares approach is readily useable in an unstructured environment, the height function approach proposed here is not, since it relies on one-dimensional integrations over large stencils. This issue is not discussed further here, and is left for future considerations.

Convergence of the level set height function approach was studied by calculating the curvature of a circle. The problem was initialized with an exact DG level set field. The curvature was calculated on various meshes ranging from $16^2$ to $128^2$ cells and using hyperbolic tangent thicknesses ranging from $\varepsilon = 0.2\Delta x$ to $\varepsilon = 0.5\Delta x$. Fig. 14 presents the $L_\infty$ convergence of the normal and curvature using different profile thicknesses. As the profile thickness is decreased (i.e., $\varepsilon = 0.2\Delta x$), the level set approaches a step function representing the liquid volume fraction and the curvature calculation is improved but the calculation of the interface normal is more prone to errors. Conversely, when the profile thickness is increased (i.e., $\varepsilon = 0.5\Delta x$) the level set becomes smoother and errors are reduced in the calculation of the normal vector but the curvature calculation deteriorates. Therefore, a balance must be obtained where good convergence of both the normal and curvature is achieved. Computational tests showed that a profile thickness defined by $\varepsilon = 0.4\Delta x$ achieves approximately second order convergence for the curvature and between first and second order convergence for the normal. Therefore, $\varepsilon = 0.4\Delta x$ was used in all of the test cases shown in this paper.

### 7.2. Spurious currents

Even though the curvature has been shown to have close to second order convergence, induced velocities due to curvature errors could be large. Therefore, a spurious currents test was conducted to determine whether the curvature errors will lead to significant spurious velocities. This test case uses the DG-CLS scheme coupled with a Navier–Stokes solver, which is described in Section 8.

The spurious currents test consists of simulating a two-dimensional drop of diameter $D = 0.4$ inside of a unit box. The physical properties used in the simulation are density ratio set to unity or $\rho_1 = \rho_2 = \rho$, surface tension coefficient $\sigma = 1$,

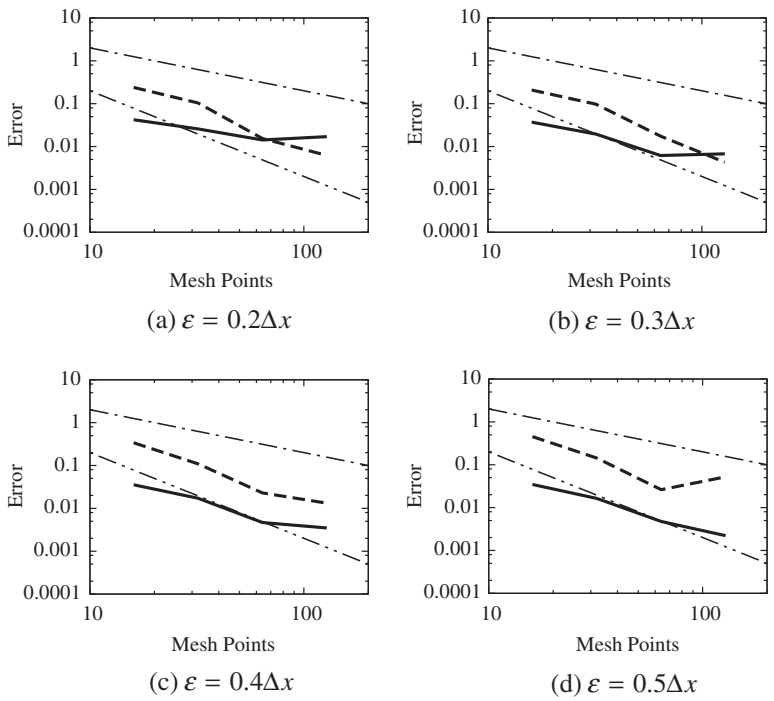



**Fig. 14.** Convergence of normal and curvature for different level set thicknesses set by $\varepsilon$. The thick solid line and the thick dashed line show $L_\infty$ errors for the interface normal and curvature, respectively. First and second order convergence is shown with the dot-dashed and the dot-dot-dashed lines for reference.

and unity viscosity ratio with $\mu_1 = \mu_2 = 0.1$. The free parameter is the density of both fluids and is used to set the Laplace number, $La = 1/(Oh)^2 = \sigma\rho D/\mu^2$. After a non-dimensional time of $t\sigma/(\mu D) = 250$, the capillary number, $Ca = |\boldsymbol{u}|_{\max}\mu/\sigma$, is computed. The capillary number is a non-dimensional estimate of the magnitude of spurious velocities. These parasitic velocities result from errors in the curvature calculation propagating through the calculation of the surface tension force in the Navier–Stokes equations.

Table 3 shows the capillary number for varying Laplace numbers on $32^2$ and $64^2$ meshes. For all Laplace numbers and both meshes, the capillary number remains small. To investigate the mesh convergence properties of the spurious currents, a test was conducted with the Laplace number fixed at 12,000 and the mesh varied from $16^2$ through $128^2$. The results in Table 4 show convergence is obtained up to a $64^2$ mesh and low capillary numbers are found for all meshes. The results indicate the curvature errors do not lead to excessive spurious velocities.

## 8. Solution of the Navier–Stokes equations

The DG conservative level set approach provides the location of the interface needed to solve the Navier–Stokes equations. The coupling of the level set method with the flow solver, NGA [23], is described in this section.

NGA solves the variable density, low Mach number Navier–Stokes equations. Discretization is done using high order conservative finite difference methods that are staggered in both space and time. The resulting scheme has been shown to be well suited for simulations of turbulent flows [23]. Time discretization is performed using an iterative, second order, Crank–Nicolson formulation. To improve robustness, each sub-iteration of the Crank–Nicolson scheme is performed using a semi-implicit approach similar to Choi and Moin [34].

The large jump in density and the pressure jump due to surface tension that occur at the interface are handled using the ghost fluid method (GFM) [2] that explicitly accounts for the discontinuity. The jump in viscous stresses is modeled using a height fraction approach [35]. Details of the incompressible Navier–Stokes equations and the discretization of the discontinuous pressure and viscous terms are provided in the following sections.

### 8.1. Incompressible Navier–Stokes equations

The incompressible form of the Navier–Stokes equations is used to describe the flow in the two phases and can be written as

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u}\cdot\nabla\boldsymbol{u} = -\frac{1}{\rho}\nabla p + \frac{1}{\rho}\nabla\cdot\left(\mu\left[\nabla\boldsymbol{u} + \nabla\boldsymbol{u}^{\mathrm{T}}\right]\right) + \boldsymbol{g}, \tag{51}$$

where $\rho$ is the density, $p$ is the pressure, $\mu$ is the dynamic viscosity, and $\boldsymbol{g}$ is the gravitational acceleration. The continuity equation can be written as

$$\frac{\partial \rho}{\partial t} + \nabla\cdot(\rho\boldsymbol{u}) = \frac{\partial \rho}{\partial t} + \boldsymbol{u}\cdot\nabla\rho = 0 \tag{52}$$

using the incompressibility constraint $\nabla\cdot\boldsymbol{u} = 0$.

The interface $I$ separates the gas phase from the liquid phase, indicated by the subscripts $g$ and $l$. Discontinuous material properties at the interface are written using the jump, i.e., $[\rho]_I = \rho_l - \rho_g$ and $[\mu]_I = \mu_l - \mu_g$ for the density and dynamic viscosity, respectively. The velocity field is continuous across the interface, i.e., $[\boldsymbol{u}]_I = 0$. The pressure is discontinuous due to surface tension and the normal component of viscous stresses, i.e.,

$$[p]_I = \sigma\kappa + 2[\mu]_I\boldsymbol{n}^{\mathrm{T}}\cdot\nabla\boldsymbol{u}\cdot\boldsymbol{n}, \tag{53}$$

where $\sigma$ is the surface tension coefficient.

**Table 3**
Capillary number for various Laplace numbers on a $32^2$ mesh and a $64^2$ mesh.

| Laplace number | Capillary number | |
|---|---|---|
| | $32^2$ Mesh | $64^2$ Mesh |
| 12 | $1.493\times10^{-4}$ | $1.602\times10^{-5}$ |
| 120 | $1.230\times10^{-4}$ | $9.981\times10^{-6}$ |
| 1,200 | $1.109\times10^{-4}$ | $7.899\times10^{-6}$ |
| 12,000 | $9.424\times10^{-5}$ | $5.536\times10^{-6}$ |
| 120,000 | $5.299\times10^{-5}$ | $9.324\times10^{-6}$ |
| 1,200,000 | $1.784\times10^{-5}$ | $2.401\times10^{-6}$ |

**Table 4**
Capillary for a Laplace numbers of 12,000 on various meshes.

| Laplace number | Capillary number | | | |
|---|---|---|---|---|
| | $16^2$ Mesh | $32^2$ Mesh | $64^2$ Mesh | $128^2$ Mesh |
| 12,000 | $1.186 \times 10^{-4}$ | $9.424 \times 10^{-5}$ | $5.536 \times 10^{-6}$ | $3.334 \times 10^{-6}$ |

### 8.2. Pressure term formulation

Discretizing the pressure gradient term in the Navier–Stokes equations requires special consideration because of the discontinuity of density as well as the jump in pressure due to surface tension and viscous stresses. The GFM [2] is used to discretize the pressure term and explicitly accounts for the jump. The GFM allows for standard finite difference operators to be used by extending the pressure across the interface using Taylor series expansions. The jump in pressure is then added explicitly afterwards.

For example, the GFM is applied to the variable coefficient pressure Laplacian in one dimension. If the interface $I$ is located at $x_I$ between the two grid locations $x_i$ and $x_{i+1}$ and $x_{i+1}$ is within the liquid phase the GFM is used. The GFM consists of first introducing $\zeta = (x_I - x_i)/\Delta x$, where $\Delta x = x_{i+1} - x_i$, and a modified density $\rho^* = \rho_g \zeta + \rho_l (1-\zeta)$. Then the pressure Laplacian used at $x_i$ within the gas phase in the pressure Poisson equation is written as

$$\frac{\partial}{\partial x}\left(\frac{1}{\rho}\frac{\partial p}{\partial x}\right)\bigg|_{g,i} = \frac{\frac{1}{\rho^*}\left(p_{l,i+1} - p_{g,i}\right) - \frac{1}{\rho_g}\left(p_{g,i} - p_{g,i-1}\right)}{\Delta x^2} - \frac{[p]_I}{\rho^* \Delta x^2}. \tag{54}$$

More details of the derivation of the previous equations are provided by Desjardins et al. [15].

### 8.3. Viscous term formulation

The GFM could be used for the viscous terms and formulations have been proposed [36], but an implicit formulation has proven challenging to develop. Therefore, the height fraction approach [35] is used to discretize the discontinuous viscosity that appears in the viscous term of the Navier–Stokes equations. In our formulation, the viscosity is required at cell vertices and cell centers [23]. At the cell centers the viscosity can be determined directly from the level set using,

$$\mu(\boldsymbol{x}, t)|_{i,j} = \begin{cases} \mu_l & G_{i,j} \geqslant 0, \\ \mu_g & \text{otherwise} \end{cases} \tag{55}$$

for the $i$, $j$ cell. $G$ is the classical signed distance level set function that is generated using a parallel fast marching routine in the calculation of the interface normal as described in Section 7. The viscosity at the cell vertices is evaluated using,

$$\mu(\boldsymbol{x}, t)|_{i+1/2,j+1/2} = \begin{cases} \mu_l & \theta_{i+1/2,j+1/2} = 1, \\ \mu_g & \theta_{i+1/2,j+1/2} = 0, \\ 0 & \mu_g = 0 \text{ and } 0 < \theta_{i+1/2,j+1/2} < 1, \\ \frac{\mu_g \mu_l}{\mu_g \theta_{i+1/2,j+1/2} + \mu_l\left(1-\theta_{i+1/2,j+1/2}\right)} & \text{otherwise}, \end{cases} \tag{56}$$

where $\theta_{i+1/2,j+1/2}$ is defined with

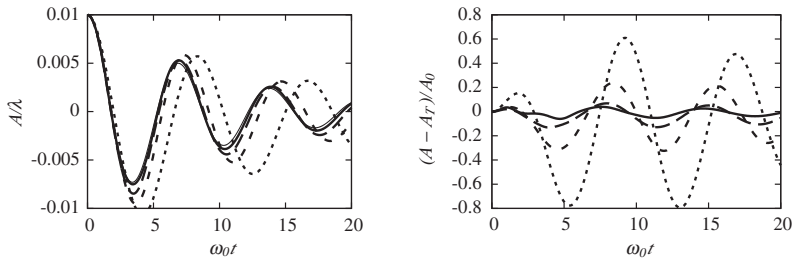
$$\theta_{i+1/2,j+1/2} = \begin{cases} 1 & G_{i+1,j} \geqslant 0, G_{i,j} \geqslant 0, \text{ and } G_{i+1,j+1} \geqslant 0, \\ 0 & G_{i+1,j} <, G_{i,j} < 0, \text{ and } G_{i+1,j+1} < 0, \\ \frac{G^+_{i+1,j} + G^+_{i,j+1} + G^+_{i,j} + G^+_{i+1,j+1}}{|G_{i+1,j}| + |G_{i,j+1}| + |G_{i,j}| + |G_{i+1,j+1}|} & \text{otherwise}. \end{cases} \tag{57}$$

The $(*)^+$ operator returns the positive part, i.e., $G^+_{i,j} \equiv \max(G_{i,j}, 0)$ [35].

Using the height fraction approach provides a sharp representation of the viscous term that is second order accurate away from the interface and first order at the interface.

## 9. Applications

Several tests are now presented that couple the proposed interface capturing scheme with the NGA flow solver. The first case is a standing wave that tests the interplay between viscous and surface tension forces. Then, the Kelvin–Helmholtz instability is studied to verify that the DG-CLS scheme and flow solver are capable of capturing this shear instability, which commonly arises in turbulent atomization problems. Finally, a three-dimensional, turbulent, two-phase atomization problem is studied.




(a) Non-dimensional wave amplitude versus non-dimensional time

(b) Non-dimensional wave amplitude error versus non-dimensional time

**Fig. 15.** Standing wave test case with unity density ratio. $8^2$ mesh shown with dotted line, $16^2$ mesh given with dashed line, long-dashed line shows $32^2$ mesh, and $64^2$ mesh shown with thick solid line. Thin solid line in (a) provides the theoretical solution.

### 9.1. Standing wave

The first test case consists of the viscous damping of a surface wave and provides insight into problems that include significant interaction between surface tension and viscous forces. The test is two-dimensional with a domain of $[0, 2\pi]^2$. Periodic boundary conditions are used in the $x$-direction along with slip conditions on the top and bottom. Two immiscible fluids are placed in the domain separated by a flat interface initially perturbed by a sinusoidal wave. The initial interface location is given by

$$\Psi(x, y, t = 0) = \frac{1}{2}\left(\tanh\left(\frac{\pi - y + A_0 \cos(2\pi x/\lambda)}{2\varepsilon}\right) + 1\right), \tag{58}$$

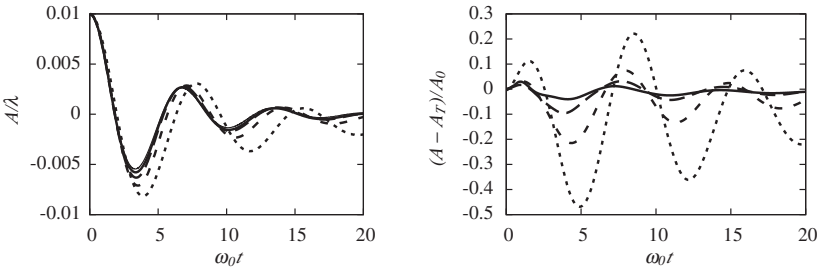
where $\lambda$ is the perturbation wavelength which is set to $2\pi$ and $A_0$ is the initial amplitude of the wave chosen to be $A_0 = 0.01\lambda$. Prosperetti [37] derived an analytical solution to the evolution of the wave amplitude with time, provided the kinematic viscosity of both fluids are equal. Time is non-dimensionalized using the inviscid oscillation frequency

$$\omega_0 = \sqrt{\frac{\sigma}{\rho_l + \rho_g}}. \tag{59}$$

For our investigation, the density ratio is set to either 1 or 1000 and four different meshes are tested, namely an $8^2$ mesh, a $16^2$ mesh, a $32^2$ mesh, and a $64^2$ mesh. Simulations are performed until a non-dimensional time of $\omega_0 t = 20$, which captures approximately three interface oscillation periods. This parameter space was chosen to follow previous studies by Herrmann [38] and Desjardins et al. [15].

For the unity density ratio case, both fluid densities were set to 1. For the high density ratio case, the liquid and gas densities were set to $\rho_l = 1000$ and $\rho_g = 1$, respectively. The non-dimensional surface tension coefficient was set to $\sigma = 2$ and the non-dimensional kinematic viscosity was set to $\nu = 0.064720863$ in both fluids. The time step was set to $\Delta t = 0.01$ for all of the mesh sizes considered. The DG parameters were fixed with the reinitialization factor $F = 0.5$ and second order DG polynomials.

Results are shown in Figs. 15 and 16 for a density ratio of 1 and 1000, respectively. The results include the wave amplitude, $A$, versus time and the error between the computational and theoretical solutions, $A_T$, as a function of time, shown in (a) and (b), respectively. Convergence to the theoretical solution with mesh refinement is shown for both tests. The results are




(a) Non-dimensional wave amplitude versus non-dimensional time

(b) Non-dimensional wave amplitude error versus non-dimensional time

**Fig. 16.** Standing wave test case with density ratio of 1000. $8^2$ mesh shown with dotted line, $16^2$ mesh given with dashed line, long-dashed line shows $32^2$ mesh, and $64^2$ mesh shown with thick solid line. Thin solid line in (a) provides the theoretical solution.

comparable to previous studies [15,38,39] and suggest the wave physics are properly captured and that the DG-CLS scheme converges to the theoretical results.

### 9.2. Kelvin–Helmholtz instability

The Kelvin–Helmholtz (KH) instability is an important shear instability mechanism for two-phase flows. In particular it is thought to play an important role in the atomization of liquid jets. In this test case we assess the ability of the DG scheme coupled with NGA to capture the KH instability and correctly predict its growth-rate.

The case considered is that of a temporal mixing layer between liquid and gas. The frame of reference is defined so that the base flow velocity at the interface is zero, leading to the following definitions of base flow velocity in the gas and liquid phases,

$$U_g(y) = U_{g,\infty}\mathrm{erf}\left(\frac{y}{\delta_g}\right) \quad \text{for} \quad y > 0, \tag{60a}$$

$$U_l(y) = U_{l,\infty}\mathrm{erf}\left(\frac{y}{\delta_l}\right) \quad \text{for} \quad y < 0. \tag{60b}$$

Note that gas and liquid are denoted by the subscripts $g$ and $l$, respectively. $U_{g,\infty}$ and $U_{l,\infty}$ represent the asymptotic velocities away from the interface. The boundary layer thicknesses are represented by $\delta_g$ and $\delta_l$. For reference, Fig. 17 shows a graphical representation of the parameters.

Eqs. (60a) and (60b) are not independent but are coupled by the continuity of shear stress at the interface. The four parameters $U_{g,\infty}$, $U_{l,\infty}$, $\delta_g$, and $\delta_l$ appearing in Eqs. (60a) and (60b) are related by

$$\frac{\mu_g U_{g,\infty}}{\delta_g} = \frac{\mu_l U_{l,\infty}}{\delta_l}. \tag{61}$$

Therefore, only three of the four parameters can be chosen and the fourth must be calculated using Eq. (61).

To analyze the growth-rate of a given disturbance to the base flow, we utilize linear stability analysis. As in the work of Bagué et al. [40], we write the disturbance to the base flow, $u$ and $v$, in terms of the stream functions $\psi_g$ and $\psi_l$ using

$$u_g = \frac{\partial \psi_g}{\partial y}, \quad v_g = -\frac{\partial \psi_g}{\partial x}, \tag{62a}$$

$$u_l = \frac{\partial \psi_l}{\partial y}, \quad v_l = -\frac{\partial \psi_l}{\partial x}. \tag{62b}$$

Since the domain is periodic in the $x$ direction and the base flow does not depend on time, we may write the stream functions as

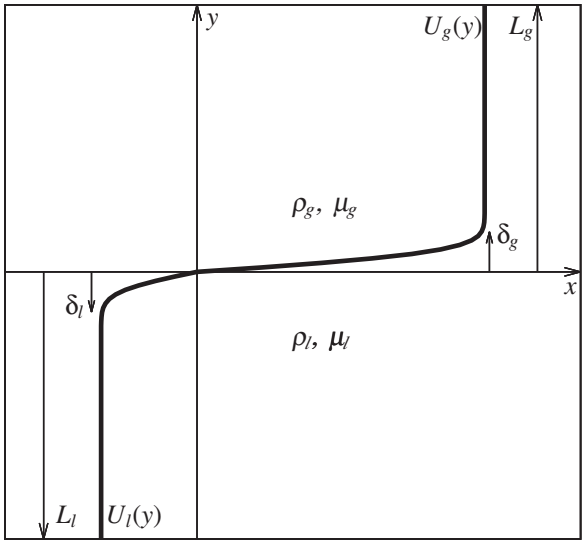



**Fig. 17.** Geometry used for Kelvin–Helmholtz test case. Gas is located in the top half of the domain and liquid in the bottom. The liquid and gas boundary layer thicknesses, $\delta_l$ and $\delta_g$, are shown along with the distance from the interface to the top and bottom of the domain represented by $L_g$ and $L_l$, respectively. The parallel base flow profile is depicted and labeled $U_g(y)$ in the gas and $U_l(y)$ in the liquid.

$$\psi_g = \phi_g(y)\exp(i\alpha(x-ct)) \quad \text{for} \quad y>0, \tag{63a}$$

$$\psi_l = \phi_l(y)\exp(i\alpha(x-ct)) \quad \text{for} \quad y<0, \tag{63b}$$

where $\phi_g$ and $\phi_l$ are the eigenfunctions within the gas and liquid, $\alpha$ is the real wave number, and the complex eigenvalue $c = c_r + ic_i$ provides the wave speed, $c_r$, and the growth-rate of the wave, $\alpha c_i$. Throughout this section, the subscripts $r$ and $i$ refer to the real and imaginary parts of the variable, respectively. Note that there is no connection to $\boldsymbol{\psi}$ and $\boldsymbol{\phi}$ used in the DG representation of the conservative level set, as $\psi$ and $\phi$ are used here to represent the stream functions and eigenfunctions for historical consistency.

The Orr–Sommerfeld equation describes the evolution of the linearized momentum equations due to the perturbation and are written in each phase,

$$U_g\phi_g'' - \alpha^2 U_g\phi_g - c\phi_g'' + c\alpha^2\phi_g - U_g''\phi_g = \frac{m}{r}\frac{1}{i\alpha \mathrm{Re}_l}\left(\phi_g^{(4)} - 2\alpha^2\phi_g'' + \alpha^4\phi_g\right) \quad \text{for} \quad y>0, \tag{64a}$$

$$U_l\phi_l'' - \alpha^2 U_l\phi_l - c\phi_l'' + c\alpha^2\phi_l - U_l''\phi_l = \frac{1}{i\alpha \mathrm{Re}_l}\left(\phi_l^{(4)} - 2\alpha^2\phi_l'' + \alpha^4\phi_l\right) \quad \text{for} \quad y<0, \tag{64b}$$

where $m = \mu_g/\mu_l$ is the viscosity ratio and $r = \rho_g/\rho_l$ is the density ratio. $\mathrm{Re}_l$ is the liquid Reynolds number and is defined, along with the other non-dimensional Reynolds and Weber numbers, as

$$\mathrm{Re}_l = \frac{\rho_l U_{g,\infty}\delta_g}{\mu_l}, \quad \mathrm{Re}_g = \frac{\rho_g U_{g,\infty}\delta_g}{\mu_g}, \quad \mathrm{We}_l = \frac{\rho_l U_{g,\infty}^2\delta_g}{\sigma}, \quad \mathrm{We}_g = \frac{\rho_g U_{g,\infty}^2\delta_g}{\sigma}, \tag{65}$$

which are all defined using the asymptotic gas velocity $U_{g,\infty}$ and the gas boundary layer thickness $\delta_g$.

There are four boundary conditions for each fourth order generalized eigenvalue problem. The four conditions (two for each Orr–Sommerfeld equation) that are located at the domain boundaries are

$$\phi_g = \phi_g' = 0 \quad \text{for} \quad y = L_g, \tag{66a}$$

$$\phi_l = \phi_l' = 0 \quad \text{for} \quad y = -L_l, \tag{66b}$$

and provide no-slip conditions for the perturbation velocities. The height of the domain is controlled by $L_g$ and $L_l$, which should be large enough to not impact the results. At the interface are the remaining four boundary conditions that ensure continuity of the normal and tangential components of the velocity as well as the normal and shear stresses. These boundary conditions can be written as

$$\phi_g = \phi_l \quad \text{for} \quad y = 0, \tag{67a}$$

$$\phi_g' + \frac{U_g'}{c}\phi_g = \phi_l' + \frac{U_l'}{c}\phi_l \quad \text{for} \quad y = 0, \tag{67b}$$

$$-\frac{\alpha^2}{rc\mathrm{We}_l} = \frac{1}{r}\left(c\phi_l' + U_l'\phi_l\right) + \frac{1}{i\alpha r\mathrm{Re}_l}\left(\phi_l^{(3)} - 3\alpha^2\phi_l'\right) - \left(c\phi_g' + U_g'\phi_g\right) + \frac{m}{i\alpha r\mathrm{Re}_l}\left(\phi_g^{(3)} - 3\alpha^2\phi_g'\right) \quad \text{for} \quad y = 0, \tag{67c}$$

$$\left(\phi_l'' + \alpha^2\phi_l + \frac{U_l''}{c}\phi_l\right) = m\left(\phi_g'' + \alpha^2\phi_g + \frac{U_g''}{c}\phi_g\right) \quad \text{for} \quad y = 0. \tag{67d}$$

All of the boundary conditions except for the one given by Eq. (67c) are linear with respect to the eigenvalue $c$. Because of the nonlinear condition, an iterative procedure needs to be employed to solve for the solution unless the boundary condition can be linearized. Boomkamp et al. [41] linearized Eq. (67c) using Eqs. (67a) and (67b) resulting in

$$-\frac{\alpha^2}{r\mathrm{We}_l}\frac{\phi_l' - \phi_g'}{U_l' - U_g'} = \frac{1}{r}\left(c\phi_l' + U_l'\phi_l\right) + \frac{1}{i\alpha r\mathrm{Re}_l}\left(\phi_l^{(3)} - 3\alpha^2\phi_l'\right) - \left(c\phi_g' + U_g'\phi_g\right) + \frac{m}{i\alpha r\mathrm{Re}_l}\left(\phi_g^{(3)} - 3\alpha^2\phi_g'\right) \quad \text{for} \quad y = 0. \tag{68}$$

The linearized form is appropriate when $U_l' \neq U_g'$. Our base flow meets this criteria and the linearized form of the boundary condition is used in this study to avoid the iterative procedure.

The Orr–Sommerfeld equation is solved numerically by approximating the eigenfunctions using a series of Chebyshev polynomials. The functions are orthogonal on an interval $[-1,1]$. Therefore, we must transform the intervals $y = [0, L_g]$ and $y = [-L_l, 0]$ to $z = [-1,1]$ within each phase using linear transforms, which can be written as

$$z_g = -\frac{2y}{L_g} + 1 \quad \text{for} \quad 0 < y < L_g, \tag{69a}$$

$$z_l = -\frac{2y}{L_l} - 1 \quad \text{for} \quad -L_l < y < 0. \tag{69b}$$

Using the transformed coordinate system, the eigenfunctions can be approximated with the Chebyshev collocation method, namely $\phi_g(z) = \sum_{n=0}^{N} a_n T_n(z)$ and $\phi_l(z) = \sum_{n=0}^{N} b_n T_n(z)$, where $T_n$ is the $n^{\text{th}}$ Chebyshev polynomial of the first kind, and $N$ is the number of polynomials used to approximate the eigenfunction. As described by Bagué et al. [40], $N$ is an important parameter

that is chosen so that a balance is achieved between errors that arise from a poor approximation of the eigenfunction and round off errors. A poor fit arises when a small number of Chebyshev polynomials are used and the series is not able to conform to the eigenfunctions. Round-off errors dominate when a large number of polynomials is used. The approach we used to find a good value for $N$ followed the method outlined by Bagué et al. [40] and consisted of increasing $N$ until a range is found where the eigenvalue remains relatively constant. Then $N$ is taken to be the number of polynomials at the beginning of this range.

Solution of the Orr–Sommerfeld equation results in the calculation of the eigenvalue $c$, and the eigenvectors $\phi_g$ and $\phi_l$, which provide the theoretical growth-rate and the perturbed velocity field needed to initialize the NGA simulation. The theoretical growth-rate is calculated from the imaginary part of the eigenvalue using $\alpha c_i$. The growth-rate is non-dimensionalized using the gas boundary layer thickness and the free stream gas velocity and can be written as $\alpha c_i \delta_g / U_{g,\infty}$. The initial perturbed velocity field used in simulations of the Kelvin–Helmholtz instability is the sum of the base flow and a perturbation that can be calculated from the eigenvectors using,

$$u(x,y,t=0) = U_g(y) + \lambda\left(\phi'_{g,r}\cos(\alpha x) - \phi_{g,i}\sin(\alpha x)\right) \quad \text{for} \quad y > 0, \tag{70a}$$

$$u(x,y,t=0) = U_l(y) + \lambda\left(\phi'_{l,r}\cos(\alpha x) - \phi_{l,i}\sin(\alpha x)\right) \quad \text{for} \quad y < 0, \tag{70b}$$

$$v(x,y,t=0) = \lambda\alpha\left(\phi_{g,r}\sin(\alpha x) + \phi_{g,i}\cos(\alpha x)\right) \quad \text{for} \quad y > 0, \tag{70c}$$

$$v(x,y,t=0) = \lambda\alpha\left(\phi_{l,r}\sin(\alpha x) + \phi_{l,i}\cos(\alpha x)\right) \quad \text{for} \quad y < 0, \tag{70d}$$

where $\alpha$ is the amplitude of the perturbation which was set to $\alpha = 10^{-3}U_{g,\infty}$ for all of the test cases. The interface is also initially perturbed and the displacement of the level set is given using the signed distance level set

$$\begin{aligned} G(x,y,t=0) &\\ &= y + \frac{\lambda\alpha^2}{\alpha^2|c|^2}\left[c_i\left(\phi_{g,i}(y=0)\cos(\alpha x) + \phi_{g,r}(y=0)\sin(\alpha x)\right) + c_r\left(\phi_{g,r}(y=0)\cos(\alpha x) - \phi_{g,i}(y=0)\sin(\alpha x)\right)\right], \end{aligned} \tag{71}$$

where $\phi_{g,r}(y=0)$ and $\phi_{g,i}(y=0)$ are the real and imaginary parts of the gas eigenfunction evaluated at the interface where $y = 0$. The conservative level set can be calculated from the signed distance level set using Eq. (6).

The growth-rate from the simulations was calculated from the temporal evolution of the perturbation. Initially, the disturbance is sinusoidal and during the linear growth period remains a sinusoidal function. Therefore, we found the amplitude of the perturbation by first fitting the function $y = \beta\sin(2\pi x/L_x + \xi)$ through the interface, where $\beta$ is the amplitude and $\xi$ is the phase of the sinusoidal function. $\beta$ and $\xi$ are found using the method of least squares. Next, the growth-rate is calculated from the slope of $\log(\beta)$ plotted versus time. It is important to only calculate the growth-rate from the linear stability regime since non-linear growth of the instability is not described by the Orr–Sommerfeld equation. In order to systematically find the end of the linear stability regime, we calculate the coefficient of determination,

$$R^2 = \frac{\left(\sum_{n=1}^{T}\left[\left(t_n - \bar{t}\right)\left(\log(\beta_n) - \log(\bar{\beta})\right)\right]\right)^2}{\sum_{n=1}^{T}\left(t_n - \bar{t}\right)^2\sum_{n=1}^{T}\left(\log(\beta_n) - \log(\bar{\beta})\right)^2}, \tag{72}$$

where $T$ is the number of time-steps since the beginning of the simulation, $t_n$ and $\beta_n$ are the time and amplitude at time-step $n$, and the bar ($\bar{*}$) denotes the average over all time-steps. If the value of $R^2$ calculated at the current time-step is less than the previous value of $R^2$, then the new data point is an outlier and therefore corresponds to the beginning of the non-linear stability region. We should remark that the actual distinction between the linear and non-linear stability regimes is difficult to define and we use the aforementioned method as a systematic way to find the transition and not a strict mathematical definition of the transition.

Four different cases were considered following a previous investigation by Bagué et al. [40]. For all of the cases, the vertical domain size was set to $L_g = 6\delta_g$ and $L_l = 6\delta_l$ and the boundary layer thicknesses were set to $\delta_g = \delta_l = 2.5 \times 10^{-3}$ m. The free-stream velocity in the gas phase was constant for all of the cases with a value of $U_{g,\infty} = 10$ m s$^{-1}$. The gas properties, $\rho_g = 1$ kg m$^3$ and $\mu_g = 1.25 \times 10^{-5}$ kg m$^{-1}$ s$^{-1}$, were also fixed throughout the study. Liquid properties were varied by changing the density ratio and viscosity ratio. Cases A and B had a density ratio of unity, $r = 1$, and a viscosity ratio $m = 0.1$. The density and viscosity ratios were changed to $r = 0.1$ and $m = 0.99$ for cases C and D. The surface tension coefficient was also varied from zero in cases A and C to $\sigma = 2.5 \times 10^{-3}$ J$^2$ m$^{-1}$ in cases B and D. Table 5 shows the important non-dimensional numbers for the four test cases. The width of the domain, $L_x$, was set to the same size of the wavelength, $\alpha$, so that one period of the disturbance fits within the computational space.

For the four test cases, numerous simulations were run with varying perturbation wavelengths and the growth-rate was calculated for each. Fig. 18 shows the evolution of the growth-rate versus the wavenumber. The process was repeated using different meshes and the results from $8^2$, $32^2$, and $128^2$ meshes are shown on the plots. The theoretical growth-rate was also calculated for each wavelength using the Orr–Sommerfeld equation and plotted for reference. As the mesh is refined the numerical growth-rates approach the theoretical growth-rates for all of the test cases. The coarsest mesh, with $8^2$ cells, is

**Table 5**
Non-dimensional numbers used in the four cases used to study the Kelvin–Helmholtz instability.

| Case | $r = \rho_g/\rho_l$ | $m = \mu_g/\mu_l$ | $\mathrm{Re}_g$ | $\mathrm{Re}_l$ | $\mathrm{We}_g$ | $\mathrm{We}_l$ |
|---|---|---|---|---|---|---|
| A | 0.1 | 1 | 2000 | 200 | $\infty$ | $\infty$ |
| B | 0.1 | 1 | 2000 | 200 | 10 | 10 |
| C | 0.99 | 0.1 | 2000 | 19800 | $\infty$ | $\infty$ |
| D | 0.99 | 0.1 | 2000 | 19800 | 10 | 10 |

capable of predicting the general trend of the growth-rate as a function of wavelength. The simulations on the $32^2$ mesh predict the growth-rates well and could provide sufficient resolution for simulations that include Kelvin–Helmholtz type instabilities. Additional meshes were also tested, namely $16^2$ and $64^2$. Using the results from all of the meshes a convergence plot, Fig. 19, was produced wherein the $L_2$ error between the numerical and theoretical growth-rates for all wavelengths is shown. For all four test cases, first order convergence was demonstrated suggesting that the DG formulation coupled to the NGA code is capable of capturing the Kelvin–Helmholtz instability.

### 9.3. Turbulent atomization

In order to assess the performance of the DG-CLS in a three-dimensional turbulent flow, the simulation of a turbulent liquid jet in quiescent air was performed. The important non-dimensional properties include the liquid Reynolds number, liquid Weber number, density ratio, and viscosity ratio, defined as

$$\mathrm{Re}_l = \frac{\rho_l u_{\mathrm{jet}} D}{\mu_l} = 5000, \tag{73}$$

$$\mathrm{We}_l = \frac{\rho_l u_{\mathrm{jet}}^2 D}{\sigma} = 5000, \tag{74}$$

$$\rho_l/\rho_g = 40, \quad \text{and} \tag{75}$$

$$\mu_l/\mu_g = 40, \tag{76}$$

where $u_{\mathrm{jet}}$ is the mean velocity of the liquid at the exit of the injector and $D$ is the injector diameter. The injector was simulated by first running a preliminary simulation of a turbulent flow in a periodic pipe to generate inflow data. After the pipe flow simulation reached statistical steady state, the velocity field was saved at a plane perpendicular to the flow direction for a large number of time-steps. The saved velocity field was used as the inflow boundary condition for the atomizing jet.

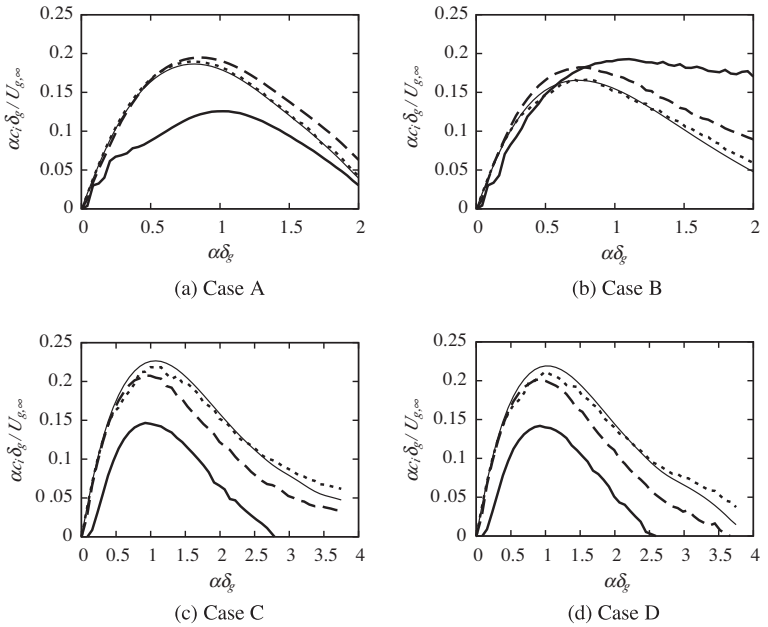



**Fig. 18.** Numerical and theoretical growth-rate for the four Kelvin–Helmholtz cases asa function of wavenumber. Numerical results are shown on three different meshes namely $8^2$ (thick solid), $32^2$ (dashed), and $128^2$ (dotted). The thin solid line shows the theoretical solution.

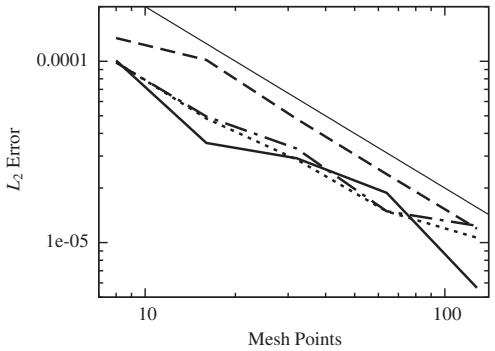



**Fig. 19.** Mesh refinement convergence for the four test cases used in the Kelvin–Helmholtz test case. Figure shows $L_2$ error of the growth-rate versus the number of mesh points for Case A (thick solid), Case B (dashed), Case C (dotted), and Case D (dot-dashed). The thin solid line shows first order convergence.

**Table 6**
Varying parameters in atomizing jet simulations.

| Case | Reinitialization factor, $F$ | DG order, $O$ |
|---|---|---|
| A | 0.5 | 1 |
| B | 0.5 | 2 |
| C | 1.0 | 1 |

Three simulations were conducted on a relatively small domain to gain insight into the effect of changing the amount of reinitialization and the order of the polynomials used in the DG discretization. The computational domain was $12D \times 3D \times 3D$, which was discretized using a $600 \times 150 \times 150$ mesh. The parameters varied between the three cases are summarized in Table 6.

Fig. 20 shows a snapshots of the interface from the simulation of case A. The overall shape and location of the jet and droplets within the domain was about the same for all of the cases indicating that changing the amount of reinitialization or the DG order does not change the overall physics of the problem. Next we turn our attention to the mass conservation properties of the scheme with the various parameters. Fig. 21 shows the mass, normalized by the expected value, as a function of time for the three cases. For all of the cases the mass loss remains on the order of 3–4% after 20 non-dimensional time units, where time was non-dimensionalized using a characteristic time defined as $T = D/u_{\text{jet}}$. As expected the least amount of mass loss was observed in case B with the higher order polynomials since the solution is more accurate. Increasing the amount of reinitialization also reduces mass loss.

Table 7 provides the amount of time spent in the DG-CLS routines, the velocity solver, and the pressure solver for the three cases. Increasing the DG order from 1 to 2 causes a large increase in the amount of time required within the DG-CLS routines. This is because when $O = 1$ there are only 4 polynomials used as basis functions, compared to the 10 polynomials needed when $O = 2$. As a result, the computational requirement is significantly larger. Increasing the reinitialization factor from $F = 0.5$ to $F = 1.0$ causes an increase, but the increase is much smaller than the increase found when the order of the DG polynomials was changed.

All three simulations were performed on 384 compute cores, and the total run time was 477 min, 788 min, and 534 min for cases A–C, respectively. For comparison, the same atomizing jet flow was also simulated using ACLS [15] on the same number of cores, which led to a total run time of 337 min. These results suggest that DG-CLS used with $F = 0.5$ and $O = 1$ is only 40% more expensive than ACLS, while DG-CLS used with $F = 0.5$ and $O = 2$ is about twice as expensive as ACLS. The memory requirements of the DG-CLS method are similar to the finite difference based ACLS method when $O = 0$, since the DG representation of the level set is the based on a single degree of freedom. The memory requirements of the DG-CLS

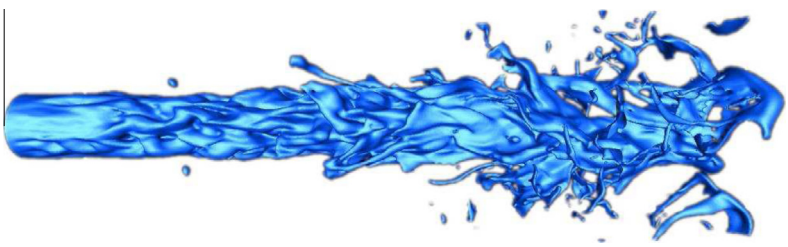


**Fig. 20.** Gas–liquid interface in simulations of a turbulent atomizing jet.

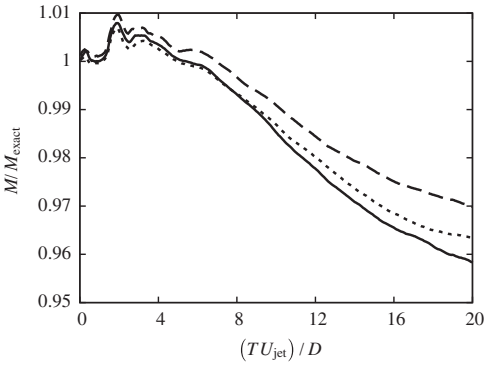



**Fig. 21.** Mass normalized by the exact mass as a function of non-dimensional time. Cases A–C are shown with the solid, dashed and dotted lines respectively.

**Table 7**
Amount of time required per time-step for DG-CLS routines, velocity solver, and pressure. The total time/time-step is also provided for completeness.

| Case | DG-CLS | | Velocity solver | | Pressure solver | | Total |
|---|---|---|---|---|---|---|---|
| A | 0.94 s | 29.3% | 0.33 s | 10.3% | 1.94 s | 60.4% | 3.21 s |
| B | 3.12 s | 58.5% | 0.32 s | 6.0% | 1.89 s | 35.5% | 5.33 s |
| C | 1.34 s | 37.1% | 0.32 s | 8.9% | 1.95 s | 54.0% | 3.61 s |

scheme increase with the number of basis functions, and four arrays are needed to store the degrees of freedom for $O = 1$ and ten for $O = 2$. Note that in comparison, the NGA code [23] uses on the order of a hundred arrays for the momentum and pressure solvers, which suggests that memory requirements of the DG-CLS method remain limited even for $O = 2$.

## 10. Conclusions

A discontinuous Galerkin discretization of the conservative level set has been developed. The conservative level set provides good mass conservation and DG offers an arbitrarily high order representation of the level set while only requiring communication with neighbors that share a cell face. This novel combination of the accurate conservative level set with a DG discretization leads to an accurate, highly parallelizable scheme, with good mass conservation properties. Calculation of the curvature was improved by using the height function approach commonly used in VOF formulations. The method results in a curvature that converges with near second order accuracy and has low errors. The DG-CLS methodology, coupled with the NGA code [23], was applied to a variety of test cases including Zalesak's disk, a two-dimensional deformation test case, the viscous damping of a surface wave, an investigation of the Kelvin–Helmholtz instability, and atomization of a turbulent jet. The test cases highlighted the accuracy and mass conserving properties of the scheme under a wide range of parameters.

## Acknowledgment

The authors thank XSEDE for the computational resources provided through a TRAC allocation.

## References

[1] J.U. Brackbill, D.B. Kothe, C. Zemach, A continuum method for modeling surface tension, J. Comput. Phys. 100 (2) (1992) 335–354.
[2] R.P. Fedkiw, T. Aslam, B. Merriman, S. Osher, A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the ghost fluid method), J. Comput. Phys. 152 (2) (1999) 457–492.
[3] W. Dettmer, P.H. Saksono, D. Perić, On a finite element formulation for incompressible newtonian fluid flows on moving domains in the presence of surface tension, Commun. Numer. Methods Eng. 19 (9) (2003) 659–668.
[4] C.W. Hirt, A.A. Amsden, J.L. Cook, An arbitrary Lagrangian–Eulerian computing method for all flow speeds, J. Comput. Phys. 135 (2) (1997) 203–216.
[5] T.J.R. Hughes, W.K. Liu, T.K. Zimmermann, Lagrangian–Eulerian finite element formulation for incompressible viscous flows, Comput. Methods Appl. Mech. 29 (3) (1981) 329–349.
[6] J. Sarrate, A. Huerta, J. Donea, Arbitrary Lagrangian–Eulerian formulation for fluid-rigid body interaction, Comput. Methods Appl. Mech. 190 (24–25) (2001) 3171–3188.
[7] M. Rudman, Volume-tracking methods for interfacial flow calculations, Int. J. Numer. Methods Fluids 24 (7) (1997) 671–691.
[8] C. Hirt, B. Nichols, Volume of fluid (VOF) method for the dynamics of free boundaries 1, J. Comput. Phys. 39 (1) (1981) 201–225.
[9] J.E. Pilliod, E.G. Puckett, Second-order accurate volume-of-fluid algorithms for tracking material interfaces, J. Comput. Phys. 199 (2) (2004) 465–502.
[10] R. Scardovelli, S. Zaleski, Direct numerical simulation of free-surface and interfacial flow, Annu. Rev. Fluid Mech. 31 (1) (1999) 567–603.

[11] S. Osher, J.A. Sethian, Fronts propagating with curvature-dependent speed: algorithms based on Hamilton–Jacobi formulations, J. Comput. Phys. 79 (1) (1988) 12–49.
[12] J.A. Sethian, Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Cambridge University Press, 1999.
[13] E. Olsson, G. Kreiss, A conservative level set method for two phase flow, J. Comput. Phys. 210 (1) (2005) 225–246.
[14] E. Olsson, G. Kreiss, S. Zahedi, A conservative level set method for two phase flow II, J. Comput. Phys. 225 (1) (2007) 785–807.
[15] O. Desjardins, V. Moureau, H. Pitsch, An accurate conservative level set/ghost fluid method for simulating turbulent atomization, J. Comput. Phys. 227 (18) (2008) 8395–8416.
[16] P. Rasetarinera, M.Y. Hussaini, An efficient implicit discontinuous spectral Galerkin method, J. Comput. Phys. 172 (2) (2001) 718–738.
[17] J.-F. Remacle, N. Chevaugeon, A. Marchandise, C. Geuzaine, Efficient visualization of high-order finite elements, Int. J. Numer. Methods Eng. 69 (4) (2007) 750–771.
[18] B. Cockburn, C.-W. Shu, Runge–kutta discontinuous Galerkin methods for convection-dominated problems, J. Sci. Comput. 16 (3) (2001) 173–261.
[19] E. Marchandise, J.-F. Remacle, N. Chevaugeon, A quadrature-free discontinuous Galerkin method for the level set equation, J. Comput. Phys. 212 (1) (2006) 338–357.
[20] E. Marchandise, P. Geuzaine, N. Chevaugeon, J.-F. Remacle, A stabilized finite element method using a discontinuous level set approach for the computation of bubble dynamics, J. Comput. Phys. 225 (1) (2007) 949–974.
[21] X. Zhang, C.-W. Shu, Maximum-principle-satisfying and positivity-preserving high-order schemes for conservation laws: survey and new developments, Proc. R. Soc. A-Math. Phys. 467 (2011) 2752–2776.
[22] M.M. Francois, S.J. Cummins, E.D. Dendy, D.B. Kothe, J.M. Sicilian, M.W. Williams, A balanced-force algorithm for continuous and sharp interfacial surface tension models within a volume tracking framework, J. Comput. Phys. 213 (1) (2006) 141–173.
[23] O. Desjardins, G. Blanquart, G. Balarac, H. Pitsch, High order conservative finite difference scheme for variable density low mach number turbulent flows, J. Comput. Phys. 227 (15) (2008) 7125–7159.
[24] D.L. Chopp, Some improvements of the fast marching method, SIAM J. Sci. Comput. 23 (1) (2001) 230–244.
[25] D.L. Chopp, Computing minimal surfaces via level set curvature flow, J. Comput. Phys. 106 (1) (1993) 77–91.
[26] M. Sussman, P. Smereka, S. Osher, A level set approach for computing solutions to incompressible two-phase flow, J. Comput. Phys. 114 (1) (1994) 146–159.
[27] R.K. Shukla, C. Pantano, J.B. Freund, An interface capturing method for the simulation of multi-phase compressible flows, J. Comput. Phys. 229 (19) (2010) 7411–7439.
[28] B. Cockburn, C.-W. Shu, TVB Runge–Kutta local projection discontinuous Galerkin finite element method for conservation laws II. General framework, Math. Comput. 52 (186) (1989) 411–435.
[29] H. Luo, L. Luo, R. Nourgaliev, V. Mousseau, N. Dinh, A reconstructed discontinuous Galerkin method for the compressible Navier–Stokes equations on arbitrary grids, J. Comput. Phys. 229 (19) (2010) 6961–6978.
[30] B. Cockburn, C.-W. Shu, The local discontinuous Galerkin method for time-dependent convection–diffusion systems, SIAM J. Numer. Anal. 35 (6) (1998) 2440–2463.
[31] S.T. Zalesak, Fully multidimensional flux-corrected transport algorithms for fluids, J. Comput. Phys. 31 (3) (1979) 335–362.
[32] S.P. van der Pijl, A. Segal, C. Vuik, P. Wesseling, A mass-conserving level-set method for modelling of multi-phase flows, Int. J. Numer. Methods Fluids 47 (4) (2005) 339–361.
[33] J.A. Sethian, A fast marching level set method for monotonically advancing fronts, Proc. Natl. Acad. Sci. 93 (1995) 1591–1595.
[34] H. Choi, P. Moin, Effects of the computational time step on numerical solutions of turbulent flow, J. Comput. Phys. 113 (1) (1994) 1–4.
[35] M. Sussman, K. Smith, M. Hussaini, M. Ohta, R. Zhi-Wei, A sharp interface method for incompressible two-phase flows, J. Comput. Phys. 221 (2) (2007) 469–505.
[36] M. Kang, R.P. Fedkiw, X.-D. Liu, A boundary condition capturing method for multiphase incompressible flow, J. Sci. Comput. 15 (3) (2000) 323–360.
[37] A. Prosperetti, Motion of two superposed viscous fluids, Phys. Fluids 24 (7) (1981) 1217–1223.
[38] M. Herrmann, A balanced force refined level set grid method for two-phase flows on unstructured flow solver grids, J. Comput. Phys. 227 (4) (2008) 2674–2706.
[39] S. Popinet, S. Zaleski, A front-tracking algorithm for accurate representation of surface tension, Int. J. Numer. Methods Fluids 30 (6) (1999) 775–793.
[40] A. Bagué, D. Fuster, S. Popinet, R. Scardovelli, S. Zaleski, Instability growth rate of two-phase mixing layers from a linear eigenvalue problem and an initial-value problem, Phys. Fluids 22 (9) (2010) 092104.
[41] P.A.M. Boomkamp, B.J. Boersma, R.H.M. Miesen, G.V. Beijnon, A Chebyshev collocation method for solving two-phase flow stability problems, J. Comput. Phys. 132 (2) (1997) 191–200.