Journal of Computational Physics 229 (2010) 7520–7544

Contents lists available at ScienceDirect

Journal of Computational Physics

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

Efficient simulation of surface tension-dominated flows through enhanced interface geometry interrogation

Petar Liovic [a,*], Marianne Francois [b], Murray Rudman [c], Richard Manasseh [d]

[a] Mathematics Informatics and Statistics Division, Commonwealth Scientific and Industrial Research Organisation (CSIRO), Graham Road, Highett VIC 3190, Australia
[b] Computer, Computational and Statistical Sciences Division, Los Alamos National Laboratory, Los Alamos, NM 87545, USA
[c] Mathematical and Information Sciences Division, Commonwealth Scientific and Industrial Research Organisation (CSIRO), Normanby Road, Clayton VIC 3168, Australia
[d] Department of Mechanical Engineering, The University of Melbourne, Parkville VIC 3010, Australia

ARTICLE INFO

Article history:
Received 13 August 2009
Received in revised form 16 June 2010
Accepted 17 June 2010
Available online 25 June 2010

Keywords:
Height functions
Curvature
Surface tension
Volume-of-Fluid
Level-set
Elliptic solvers

ABSTRACT

In this paper, three improvements for modelling surface tension-dominated interfacial flows using interface tracking-based solution algorithms are presented. We have developed an improved approach to curvature estimation for incorporation into modern mesh-based surface tension models such as the Continuum Surface Force (CSF) and Sharp Surface Force (SSF) models. The scheme involves generating samples of curvature estimates from the multitude of height functions that can be generated from VOF representations of interfaces, and applying quality statistics based on interface orientation and smoothness to choose optimal candidates from the samples. In this manner, the orientation-dependence of past schemes for height function-based curvature estimation is ameliorated, the use of compact stencils for efficient computation can be maintained, and robustness is enhanced even in the presence of noticeable subgrid-scale disturbances in the interface representation. For surface tension-dominated flows, the explicit capillary timestep restriction is relaxed through timescale-separated slope limiting that identifies spurious modes in curvature evolution and omits them from contributing to surface force computations, thus promoting efficiency in simulation through the use of less timesteps. Efficiency in flow simulation is further promoted by incorporating awareness of interface location into multigrid preconditioning for Krylov subspace-based solution of elliptic problems. This use of interface-cognizance in solving problems such as the Helmholtz equation and the Poisson equation enables multigrid-like convergence in discontinuous-coefficient elliptic problems without the expense of constructing the Galerkin coarse-grid operator. The key improvements in the surface tension modelling and the numerical linear algebra are also applicable to level-set-based interfacial flow simulation.

© 2010 Elsevier Inc. All rights reserved.

1. Introduction

The need for surface tension modelling in transient Computational Fluid Dynamics (CFD) simulation is a defining feature of interfacial flows spanning a wide range of length scales. In large-scale free surface and multiphase flows such as bubbling due to gas venting in pools [24] and coastal wave breaking [25,23], interfaces with large mean radii of curvature also feature small-scale wrinkling, fragmentation and coalescence. Such events occur on length scales much smaller than that of the

* Corresponding author. Tel.: +61 3 9252 6341; fax: +61 3 9252 6586.
E-mail address: Petar.Liovic@csiro.au (P. Liovic).

0021-9991/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2010.06.034

mean interface, and may involve many separate fluid bodies of differing sizes undergoing deformation. Low error for well-resolved interfaces, and the graceful degradation of simulation behaviour when interfaces are only partially resolved, are required if multi-material flow simulation is to be used in preference to interdispersed-phase modelling as the basis for computation of such flows. In idealized small-scale flows involving few bubbles or drops, surface tension is often the dominant force, and the feasibility of simulation is dependent on the robustness of both the surface tension scheme and the flow solver in minimizing the introduction and amplification of errors. For both large-scale and small-scale flows in which surface tension effects are relatively important, it is desirable for any multi-material flow solver applied to the flow to feature: (i) accurate curvature estimation through better resolution of interfaces by larger simulation; (ii) better curvature estimation with less interface resolution; (iii) accurate and stable simulation with longer timesteps to enable efficient high-resolution simulation.

Surface tension is one physics element that needs to be captured in mesh-based Eulerian simulation of interfacial flows based on *interface tracking*. Of the approaches to interface tracking for Eulerian CFD solution algorithms, volume tracking (Volume-of-Fluid (VOF) methods [15,27]) is one of the most prominent featuring acceptable accuracy, conservation and ease of implementation properties. To accompany VOF methods for interface tracking, methods for surface tension modelling have been developed that utilize the VOF interface representation to compute appropriate surface tension forces at or near the interface. The original Continuum Surface Force (CSF) method [4] for achieving this did so by computing a surface force within a multiple-cell-thick support around the interface. More recent efforts have aimed for Sharp Surface Force representations that, coupled with a balanced-force discretization, are able to preserve static equilibrium [9]. In real flow simulations featuring no imposed or *a priori* knowledge of interface shape, the accuracy of surface force estimates generated using either approach is ultimately most dependent on the accuracy of interface curvature estimation.

Curvature estimation from fractional volume data has been identified as the main weakness of VOF-based surface tension modelling [35], and has motivated significant research effort (e.g. [4,46,43], among others). Derived from the second-derivative of interface location, curvature estimates from VOF representations of interfaces are known to be prone to aliasing errors [6] that result in visible spurious modes in solutions. The most well known of the spurious modes associated with surface tension modelling in interfacial flow simulation are parasitic currents: these currents are well-documented in VOF-based simulation from back to the original CSF work by Brackbill et al. [4]. Past approaches used in curvature estimation from fractional volume data have included kernel smoothing of the VOF interface representation [4], kernel-based expressions for curvature derivatives [37], direct differencing of continuous distance functions extracted from fractional volume data [6], and height functions [13,40,10]. Curvature estimates based on distance function level-sets have been found in various studies to not be reliable in realizing oft-cited second-order accuracy. Such locally large errors in curvature occur due to the aliasing in curvature computations of high-frequency error – whether the error be in distance functions extracted from VOF data [6], or in distance functions after advection and re-initialization in level-set interface tracking [31]. Curvature estimates based on height functions extracted from VOF data are reliable in achieving second-order accuracy, and are accurate when the height function reliably represents the interface. However, large errors in height function-based curvature estimates have been found to occur when height functions are inconsistent and inaccurate representations of the interface, such as when interfaces are poorly resolved and when interface normals are oriented away from all Cartesian axis directions [35,29]. Parasitic currents in surface tension modelling from curvature errors are one of the main factors currently hampering the widespread application of interface tracking-based CFD to small-scale flows (mm-scale and smaller). The ultimate aim of the current work is to improve the simulation of surface tension-dominated flows, such that VOF or level-sets can be confidently applied to small-scale problems.

In this paper, we introduce: (1) a novel numerical scheme for enhanced curvature estimation in interfacial flows using VOF and/or level-set interface tracking; (2) a novel curvature evolution scheme for enhanced stability of interfacial flow simulation; (3) a new multigrid-based Krylov subspace solver for linear discontinuous-coefficient elliptic problems. The improvements in curvature estimation pursue paths (ii) and (iii) in flow solver improvement for high-fidelity interfacial flow simulation. The paper also addresses path (i) – better resolution by larger simulation – by reducing the computational time associated with converging the elliptic systems that arise in most mesh-based flow solver algorithms. The paper makes progress in curvature computation and elliptic solver convergence by introducing more geometric insight into curvature estimation, as well as into discontinuity capturing within multigrid coarse-grid operators. The paper demonstrates the possibilities for existing interface tracking-based CFD codes to be modified for enhanced robustness in their handling of small-scale surface tension-dominated flows.

2. Numerical method

2.1. Governing equations

Volume tracking is applied to the single-field formalism for describing two-fluid flow by introducing a phase indicator or *color* function

$$C(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{x} \text{ occupied by phase } k = G, \\ 0 & \text{if } \mathbf{x} \text{ occupied by phase } k = L, \end{cases} \tag{1}$$

Volume-of-Fluid methods use the color function for the purpose of interface tracking according to the advection equation

$$\frac{\partial C}{\partial t} + \mathbf{U} \cdot \nabla C = 0. \tag{2}$$

The interface is implicitly defined by the spatial distribution $C(x,y,z)$ in the discrete sense as regions of $0 < C < 1$. The Product Rule is used to express the equation in a form amenable to conservative finite-volume discretization

$$\frac{\partial C}{\partial t} + \nabla \cdot (\mathbf{U}C) = -C\nabla \cdot \mathbf{U}. \tag{3}$$

By defining a C-weighted local density

$$\rho = C\rho_G + (1 - C)\rho_L, \tag{4}$$

the equation describing mass conservation for individual fluid species tracked by the indicator function is

$$\frac{\partial \rho_S}{\partial t} + \mathbf{U} \cdot \nabla \rho_S = -\rho_S \nabla \cdot \mathbf{U}, \tag{5}$$

where S denotes fluid species. The aggregate result across the mixture is the total mass conservation equation: in the general case, this is

$$\frac{\partial \rho}{\partial t} + \mathbf{U} \cdot \nabla \rho = -\rho \nabla \cdot \mathbf{U}. \tag{6}$$

In incompressible flow, the Equation of Continuity

$$\nabla \cdot \mathbf{U} = 0 \tag{7}$$

reduces the RHS contributions of Eqs. 3, 5, 6 to zero. In compressible flow, substituting the thermodynamic relation

$$\frac{dP}{d\rho} = c^2 \tag{8}$$

combined with density derivatives $\nabla\rho = \frac{d\rho}{dP}\nabla P$ and $\frac{\partial \rho}{\partial t} = \frac{d\rho}{dP}\frac{\partial P}{\partial t}$ into Eq. (6) yield the modified Equation of Continuity

$$\frac{\partial P}{\partial t} + \mathbf{U} \cdot \nabla P = -\rho c^2 \nabla \cdot U. \tag{9}$$

For gas–liquid flows featuring low gas-phase speeds, a "limited compressibility" treatment that simplifies the Equation of Continuity to

$$\nabla \cdot \mathbf{U} = -\frac{1}{\rho c^2}\frac{\partial P}{\partial t} \tag{10}$$

has been proposed by Hirt and Nichols [14] as an acceptable treatment of compressibility associated with acoustic waves. The multi-material flow solution algorithm being presented here enables gas-sided velocity fields to be generated as part of the solution (unlike liquid-sided flow simulation codes that incorporate vapor models to model compressible gas effects), which is highly advantageous for the purpose of versatility and widespread applicability.

In the single-field formalism, the momentum equation is

$$\frac{\partial \rho \mathbf{U}}{\partial t} + \nabla \cdot (\rho \mathbf{U}\mathbf{U}) = -\nabla P + \rho \mathbf{g} + \nabla \cdot \tau + \sigma\kappa\hat{\mathbf{n}}_i\delta, \tag{11}$$

where the last term on the RHS is the surface tension force that is only non-zero at the interface. The surface tension force is dependent on interface orientation $\hat{\mathbf{n}}$ and interface curvature κ: the former is a first-derivative of interface location

$$\hat{n} = \frac{\nabla C}{|\nabla C|}, \tag{12}$$

while the latter requires second-derivatives of interface location

$$\kappa = \nabla \cdot \hat{n} = \nabla \cdot \frac{\nabla C}{|\nabla C|}. \tag{13}$$

2.2. Flow solver and discretization

The flow solver used to solve the small-scale flows described in this paper is the MFVOF-3D code [27,24,25]. The flow solver is based on finite-difference methods applied to orthogonal meshes. For various terms in the governing equations, discretizations variously described as being finite-difference or finite-volume are used to promote consistency and conservation in simulated solutions. The MFVOF-3D code features 3D PLIC-VOF for interface tracking on a mesh twice as fine as the flow

field mesh, as proposed in [37]. The Centroid-Vertex Triangle-Normal Averaging (CVTNA) scheme for interface reconstruction [27] enables second-order accuracy in interface tracking to be achieved, and unsplit advection in 3D [27,26] enables this to be achieved in a single-stage VOF method while locally conserving mass. Key additional numerics – including the novel surface tension model and linear solver for efficient and scalable large-scale simulation – are discussed below.

2.2.1. Solution algorithm

Our methodology for solving incompressible multi-material flows is to base the solution algorithm on a projection method for achieving pressure–velocity coupling. Rudman [37] outlines a conventional projection method solution algorithm for staggered meshes, while Francois et al. [9] describes a consistent balanced-force algorithm with the projection method combined with a Rhie-Chow-based interpolation for strong pressure–velocity coupling on unstructured (colocated) meshes. The current work using the MFVOF-3D code [24,25] is based on the Rudman algorithm, using a staggered MAC mesh featuring pressure P at (i,j,k) mesh cell-centers, and velocity components at face-centers that are half a cell offset from the cell-centered locations. The basic steps in integrating the solution from $(\mathbf{U}^n, P^n, C^n)$ to $(\mathbf{U}^{n+1}, P^{n+1}, C^{n+1})$ are best illustrated with the first-order accurate in time description, and are:

(1) Explicit interface location update

$$C^{n+1} = C^n - \nabla \cdot (\mathbf{U}C) + C^n \nabla \cdot \mathbf{U}. \tag{14}$$

(2) Update bulk-property distributions (Eq. (4)).
(3) Generate estimate of velocity field:

$$\mathbf{U}^* = \mathbf{U}^n + ADV + VISC + SFORCE, \tag{15}$$

where *ADV*, *VISC* and *SFORCE* are discretizations of the advection, viscous stress and surface tension force terms in Eq. (11).
(4) Solve linear system generated by consistent discretization of the Helmholtz equation

$$\nabla \cdot \left(\frac{1}{\rho}\nabla P\right) = \frac{1}{\delta t}\nabla \cdot \mathbf{U}^* + \frac{1}{\delta t \rho c^2}\frac{\partial P}{\partial t}. \tag{16}$$

(5) Complete update of velocity field to time $n+1$:

$$\mathbf{U}^{n+1} = \mathbf{U}^* - \frac{\delta t}{\rho}\nabla P. \tag{17}$$

The basic algorithm is repeated twice when second-order Euler timestepping is used for time integration, with $n+1/2$ estimates used for the RHS terms (Eq. (15)) in the second half of the time integration.

2.3. Surface tension modelling

The CSF method introduced a finite kernel width for mimicking the effect of the delta function in the surface tension term $\sigma\kappa\hat{n}\delta$. Given a VOF representation of the interface, Brackbill et al. [4] defined the surface tension force to be

$$\sigma\kappa\hat{n}\delta = \sigma\kappa\nabla C, \tag{18}$$

which is discretized at $(i+1/2,j,k)$ as

$$SFORCE_{x|i+1/2,j,k} = \sigma\kappa_{i+1/2,j,k}\left.\frac{\partial C}{\partial x}\right|_{i+1/2,j,k}. \tag{19}$$

More recently, Francois et al. [9] introduced the Sharp Surface Force (SSF) method – a Ghost Fluid-inspired approach in which the force is computed only at face-velocity locations immediately either side of the interface. For the same $(i+1/2,j,k)$ point,

$$SFORCE_{x|i+1/2,j,k} = \pm\frac{\sigma\kappa_I}{\delta x_{i+1/2}}, \tag{20}$$

where κ_I is the curvature at (interpolated to) the interface, and the sign depends on which side of the interface the meshpoint lies.

With regards to the time integration, we compute $SFORCE^{n+1/2}$ in the update $\mathbf{U}^n \to \mathbf{U}^{n+1/2}$ by Eq. (15), and then $SFORCE^{n+1}$ in the full update $\mathbf{U}^n \to \mathbf{U}^{n+1}$. In the context of balanced-force discretization across all terms in the momentum update, computing surface force at the advanced times helps ensure that the localized acceleration prescribed by the surface tension model is countered by the jump in pressure.

A cost associated with incorporating SSF into VOF-based interfacial flow simulation is a need for a sharp interface representation. Planar interface reconstructions can be used for this purpose, but it is more convenient to adopt the standard Ghost Fluid Method (GFM) practise of using the zero level of a distance function. Distance functions are not naturally associated with VOF-based simulation (the Combined Level-Set Volume-of-Fluid (CLSVOF) [43] being an exception), but are

easily generated by utilizing interface reconstruction information already generated during the course of the VOF update of C (as first demonstrated in [6]). The method developed for 2D and 3D Reconstructed Distance Function (RDF) generation by Liovic and Lakehal [25] is especially convenient: a summary of the method is:

(1) determine the closest interface plane centroid to each mesh cell-center (i,j,k) in a 1–2 cell support around the interface;
(2) compute $\phi^{\mathrm{RDF}}_{i,j,k}$ in the support as the distance to the centroid, scaled by a cosine dependence reflecting the deviation of the centroid/mesh-point line away from the interface normal;
(3) tauten the RDF distribution in the vicinity of the zero level such that the sum of $|\phi^{\mathrm{RDF}}|$ at two points either side of the interface reflects the actual distance between those two mesh-points (shown in Fig. 1);
(4) fill out the support.

As we will now discuss, even if CSF is used in preference to SSF for surface tension modelling, there is still significant merit in having an RDF available.

2.3.1. Curvature computation

Generating a "height function" from the distribution of C and then computing curvature using second-order centered differencing has been found to be second-order accurate [40], and superior to curvatures based on finite-differences or kernel derivatives of the C or RDF distributions [40,6,9]. As such, many modern surface tension model implementations are based on the height function methodology for curvature estimation. With the resultant increased understanding of the height function approach for curvature estimation, shortcomings in the approach that have required significant and ongoing efforts at improvement have included poor and inconsistent behaviour in low-resolution regions [35], and high sensitivity of curvature estimates to interface orientation [29]. While such recent works represent improvement in height function-based curvature estimation, they do not necessarily exploit the geometric foundations of the various interface constructs and the curvature stencil to their fullest. We explore the geometric foundations of height functions further to identify new routes towards accurate and robust curvature estimation, especially when interfaces are poorly resolved.

The height function approach is effective for curvature estimation from VOF data because it parametrizes the interface: the subsequent formula for curvature

$$\kappa = -\frac{h_{yy} + h_{zz} + h_{yy}h_z^2 + h_{zz}h_y^2 - 2h_{yz}h_y h_z}{\left(1 + h_y^2 + h_z^2\right)^{3/2}}, \tag{21}$$

does not feature derivatives in the x-direction. As long as the height function varies in a direction essentially perpendicular to the yz-plane, transverse variations in h in the curvature stencil remain small (as shown in Fig. 2(a)) and well approximated by centered differencing. As such, the height function is a significant improvement on past finite-differencing or

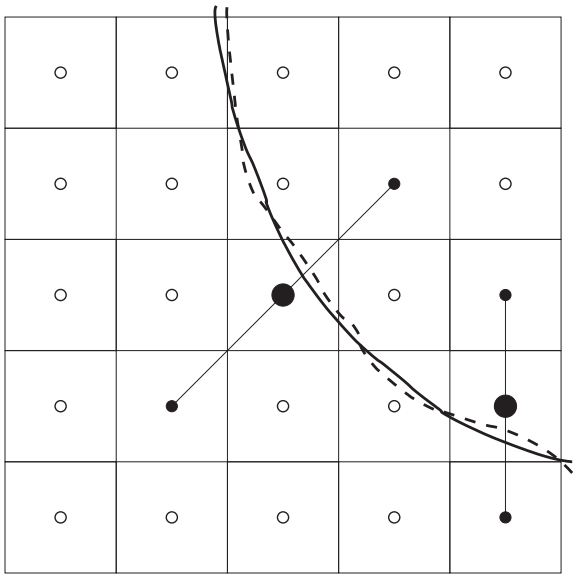

Fig. 1. Economical RDF generation from existing data from piecewise-linear/planar Volume-of-Fluid methods.

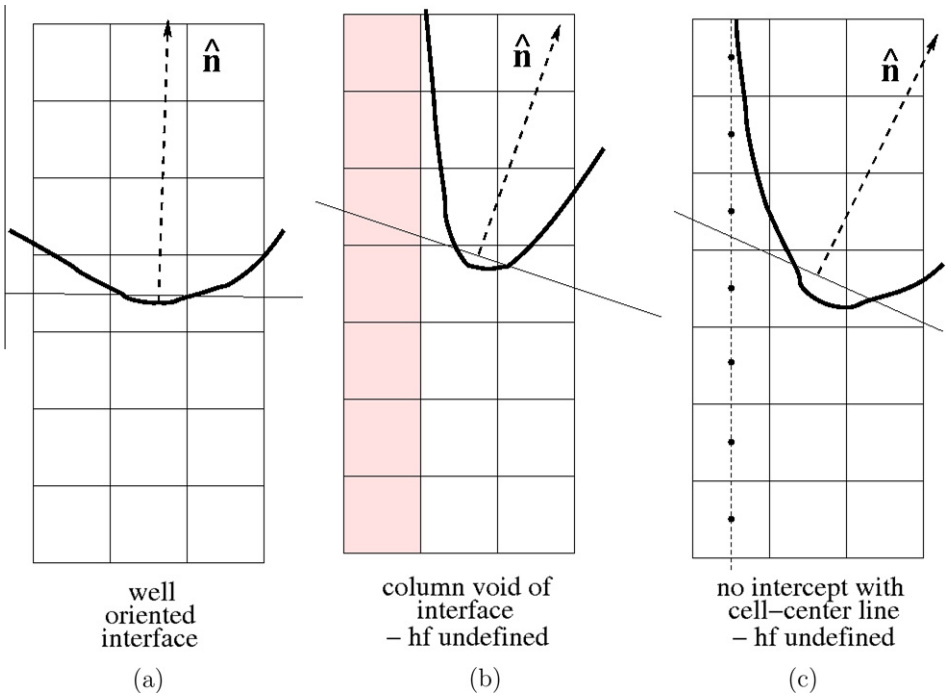

Fig. 2. Interplay between height function orientation and interface curvature in the regular height function method (e.g. as described in [6,9]). In (a), good correspondence between interface normal and height function normal means that heights in all columns of the height function stencil are defined. In (b), when the mean interface normal is oriented at 45 degrees to the height function normal and there is adequate interface curvature within the stencil, then the possibility of no interface residing in a column of a $7 \times 3 \times 3$ stencil can arise. In (c), height functions based on the distance function level-set has a higher probability of resulting in an undefined interface height within a column of the height function stencil, because the interface curvature is significant enough to result in no intercept with the vertical line through the column-centre even though some of the interface resides in the column.

convolution-based curvature schemes that involved computing derivatives based on discrete operators applied across the contact discontinuity that the steep variation in C represents. This is the reason for the usual prescription that the dominant interface normal component designate the direction in which the height function is computed.

When the interface normal $\hat{\mathbf{n}}$ is oriented away from the direction of height function variation (the "height function normal"), it is well known and documented (e.g. [29]) that curvature estimates using the regular height function method deteriorate, towards being worst when $|\hat{n}_x| \approx |\hat{n}_y| \approx |\hat{n}_z|$ (or $|\hat{n}_x| \approx |\hat{n}_y|$ in 2D where $\hat{n}_z \equiv 0$). Interface normal orientation at 45 degrees between orthogonal axis directions implies a larger variation in height across the 2D height function stencil (the 3×3 (or larger) stencil in the plane perpendicular to the direction of height function growth). Interface curvature within a fixed-size stencil of C data used for local height function generation can exacerbate the variation in height. This can occur to the extent that some part of the interface in the stencil is not intersected by a height function column, hence resulting in an undefined or inaccurate interface height in a column of the 2D height function stencil; this is shown in Fig. 2(b). In short, such symptoms show previously documented height function-based curvature schemes based on the fixed $7 \times 3 \times 3$ stencil size to suffer from issues such as dependence on interface orientation, limited stencil size and poor low-resolution behaviour [29,35]. The current work is specifically aimed at making curvature estimates *orientation-independent*.

2.3.1.1. Height functions in arbitrary directions. The obvious prescription in our new method for curvature estimation is to generate height functions in directions other than those aligned with coordinate axes, such that interface segments oriented away from the axes can fully benefit from estimation of curvature using well-oriented height functions. On an orthogonal mesh, the task is conceptually straight-forward – define additional height functions in diagonal directions. In our new curvature scheme, we compute and store curvature estimates generated from the various height functions we can define. The number of curvature candidates we generate depends on: (1) the number of different height function stencils we can conveniently define; (2) the different types of height functions we can conveniently generate in any particular stencil.

The first set of candidates for curvature we introduce are those based on height functions generated in the usual manner from VOF data on a $7 \times 3 \times 3$ stencil [9] – that is, height functions generated by sweeping in the coordinate axis vector directions, as shown in Fig. 3(a). Taking the example of a height function defined in the $\mathbf{i}$ direction, the local 3×3 height function stencil for curvature computation is filled out using the formula

$$h^{\mathbf{i}}_{jj,kk} = \sum_{ii=-3}^{3} C_{i+ii,j+jj,k+kk} \delta x_{i+ii} \tag{22}$$

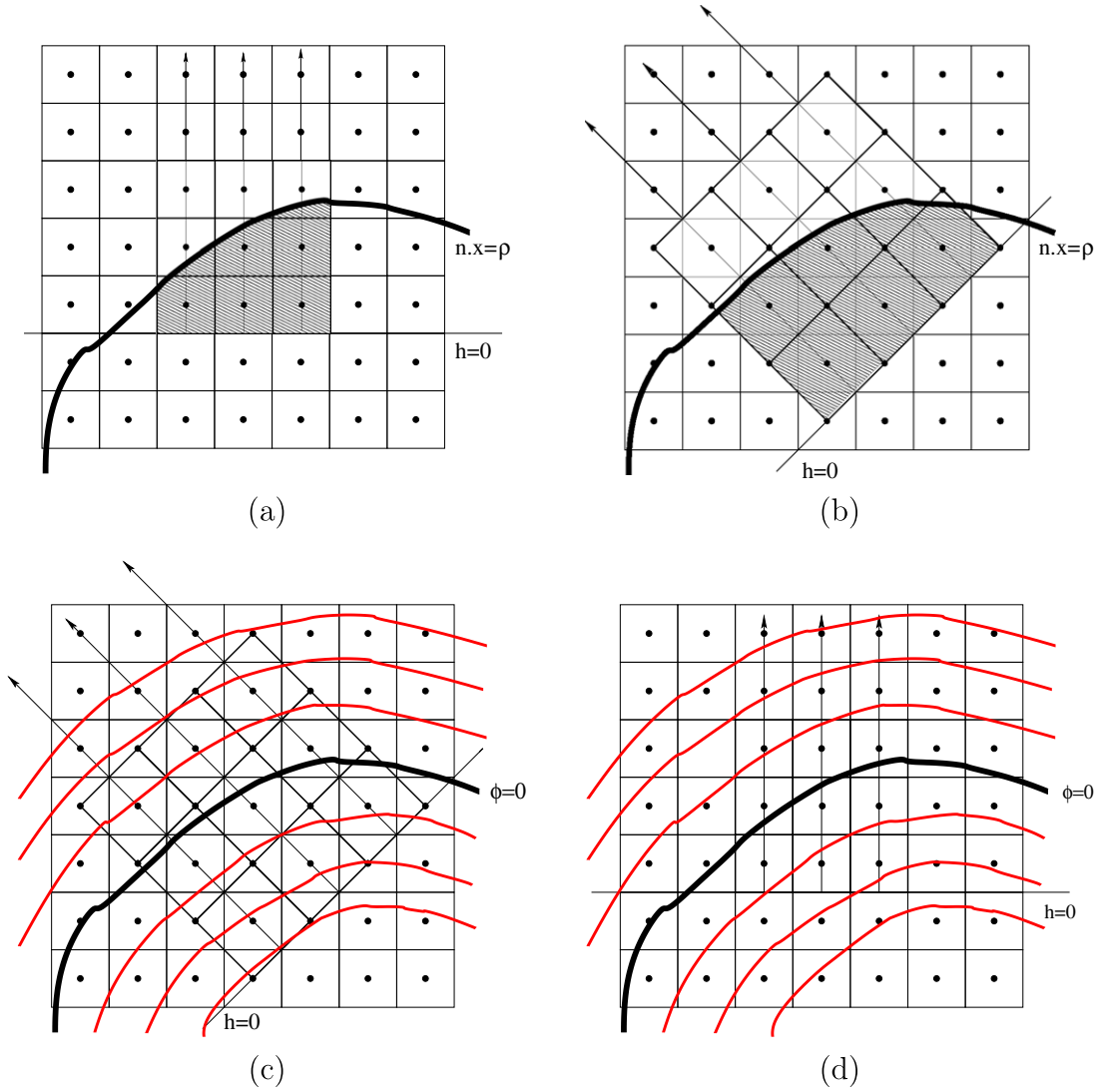

Fig. 3. Generating height functions of diagonal orientation on a uniform mesh. In (a), the underlying computational mesh with control volumes centered about (i,j,k) locations, and with vertices at $(i+1/2,j+1/2,k)$ locations. The local C distribution on this mesh is used for the height functions that yield the first set of candidates for curvature. In (b), a newly-defined mesh is tilted 45 degrees in the xy-plane, and is generated by connecting $(i+1/2,j+1/2,k)$ vertices in the **i**–**j** and **i** + **j** directions; C on this mesh yields the second set of candidates. In (c), the distance function-based alternative to generating height functions on diagonally-oriented meshes yield the third set of candidates. The fourth set of candidates results from the RDF on the regularly-oriented mesh (in (d)).

over the indices $-1 \leqslant jj,kk \leqslant 1$. Height functions are defined in the **j** and **k** directions in a similar manner. The first set of candidates for curvature are generated from these three different height function definitions, and the curvatures are denoted as $\kappa^{\mathbf{i}}$, $\kappa^{\mathbf{j}}$ and $\kappa^{\mathbf{k}}$.

The second set of possible candidates we consider are those based on the color function distribution as defined on diagonally-oriented meshes. Because C is the fractional volume within control volumes about cell centers (i,j,k), cells on the diagonally-oriented meshes are generated by diagonally connecting $(i+1/2,j+1/2,k)$, $(i+1/2,j,k+1/2)$ or $(i,j+1/2,k+1/2)$ locations on the underlying (regularly-oriented) mesh. Fig. 3(b) shows the appearance of the mesh defined at a 45 degree tilt in the (x,y)-plane relative to the underlying mesh (sub-plot (a)). The task of mapping C from the normal mesh to the diagonally-oriented mesh is done in much the same way as flux calculation is done in unsplit-advection Volume-of-Fluid methods [27]: (i) all cells on the underlying mesh are subdivided into four triangular prisms and the vertices of each prism are stored; (ii) planar interface reconstruction description $(\hat{\mathbf{n}}, \rho)$ and each set of prism vertices are used to compute the volumes truncated by the interface within the prisms; (iii) the prisms are reconstituted into diagonally-oriented mesh cells, and the truncated volumes and total volumes are summated to generate fractional volumes C^{xy} on the tilted mesh. The cells on the diagonally-oriented mesh are different to those on the underlying mesh, so the mesh cell spacing supplied to the height

function and curvature discretization must be adjusted accordingly. For the tilt in the *xy*-plane, unique height functions can be defined in mutually orthogonal directions, e.g. $h^{\mathbf{i}-\mathbf{j}}$ and $h^{\mathbf{i}+\mathbf{j}}$. The height function in the $-\mathbf{i}+\mathbf{j}$ direction is computed as

$$h_{0,0}^{-\mathbf{i}+\mathbf{j}} = \sum_{ll=ll\_lower}^{ll\_upper} C^{xy}_{i-ll,j+ll,k}\sqrt{(\delta x_{i-ll})^2 + (\delta y_{j+ll})^2}. \tag{23}$$

The second set of candidates are denoted in the general form $\kappa^{a\mathbf{i}+b\mathbf{j}}$. The six candidates generated here are $\kappa^{\mathbf{i}-\mathbf{j}}$, $\kappa^{\mathbf{i}+\mathbf{j}}$, $\kappa^{\mathbf{i}-\mathbf{k}}$, $\kappa^{\mathbf{i}+\mathbf{k}}$, $\kappa^{\mathbf{j}-\mathbf{k}}$ and $\kappa^{\mathbf{j}+\mathbf{k}}$. [Height functions in diagonal directions $a\mathbf{i}+b\mathbf{j}+c\mathbf{k}$ are neglected in the current work.]

Noting the possibility for generating distance function level-sets from fractional volume VOF data [6,25], additional sets of candidates can be obtained by following the prescription of recent level-set methods (e.g. [41]) in generating height functions from distance function level-sets. As such, a third possible set of candidates worth considering is obtained from RDF data by computing the distance from the $\phi^{\mathrm{RDF}} = 0$ isosurface to some reference plane along diagonal lines through (i,j,k) locations. In Fig. 3(c), we again use the case of the height function being oriented in the $-\mathbf{i}+\mathbf{j}$ direction; the height function in this case is denoted $h^{\mathrm{RDF}-\mathbf{i}+\mathbf{j}}$. The height function in this case is computed by locating the interface between (i,j,k) and $(i-1,j+1,k)$, which is easily detected if $\phi^{\mathrm{RDF}}_{i,j,k}\phi^{\mathrm{RDF}}_{i-1,j+1,k} < 0$; setting $a=-1$ and $b=1$, the height function is computed as

$$h_{0,0}^{\mathrm{RDF}a\mathbf{i}+b\mathbf{j}} = \frac{\left|\phi^{\mathrm{RDF}}_{i,j,k}\right|}{\left|\phi^{\mathrm{RDF}}_{i,j,k}\right| + \left|\phi^{\mathrm{RDF}}_{i+a,j+b,k}\right|}\left|\mathbf{x}_{i+a,j+b,k} - \mathbf{x}_{i,j,k}\right| + \left|\mathbf{x}_{i,j,k} - \mathbf{x}_{ref}\right|, \tag{24}$$

where $\mathbf{x}_{i,j,k}$ and $\mathbf{x}_{i+a,j+b,k}$ are consecutive cell-center coordinate points along a diagonal line through (i,j,k) locations, and $\mathbf{x}_{ref}$ is a reference point upstream of the line between (i,j,k) and $(i+a,j+b,k)$ from the (i,j,k) side. The third set of candidates are denoted in the general form $\kappa^{\mathrm{RDF}a\mathbf{i}+b\mathbf{j}}$, and again features six candidates. A fourth possible set of candidates is trivially generated using RDF data with height functions defined in coordinate axis directions, as shown in Fig. 3(d).

It is worth noting that there is no need to compute all sets of candidates: we only compute candidates if the data from which the height functions are to be extracted is readily available. Alternative scenarios subsequently exist for implementing an orientation-independent curvature scheme in a miscellaneous multi-material flow solver.

- *No VOF interface tracking.* If VOF is not used, then we presume that distance function level-set data is available instead, hence only compute the third and fourth sets of candidates.
- *VOF with no RDF generator.* If a flow solver module for generating a sufficiently accurate RDF is not available, then the first and second sets of candidates must be computed. This then requires the code infrastructure – specifically a 3D PLIC-VOF interface tracking scheme that features vertex-based volume computations [27] – be in place to enable regular *C*-mesh data to be transferred to the three different diagonally-oriented meshes (for C^{xy}, C^{xz}, C^{yz}).
- *VOF with no diagonal-C creator.* If a flow solver module is not available for decomposing fractional volumes in rectangular regular-mesh cells and then reconstituting the volumes to create the diagonal-mesh *C* distributions, then a module for generating an RDF from VOF data [6,25] is required. The orientation-independent curvature scheme in this case uses the first and third set of candidates.
- *VOF with RDF generator and diagonal-C creator.* Provided the diagonal-*C* and RDF distributions are economically generated, then the orientation-independent curvature scheme can utilize the first, second and third sets of candidates.

For computation of the height functions, the maximum stencil size for all diagonally-oriented height functions is $5\times3\times3$ (i.e. *ll_lower* = −2 and *ll_upper* = 2), while the usual $7\times3\times3$ stencil is used as the maximum size for the regular-orientation *C*-based height function. The choice of $5\times3\times3$ here is for robust performance for low-resolution application where the change in height across the stencil may be significant. This is because of the fact that height function computation based on ϕ is more prone to resulting in undefined interface heights than when *C* is used, as shown in Fig. 2(c). It should be stressed that they refer to *maximum* stencil size, and that in reality smaller stencils achieved through compression in the vertical direction are also used. Prior to the height function computation, a first rapid sweep of the stencil is performed to identify interfaces in all columns, and compression in the vertical direction is used if the actual variation in height in the stencil warrants it. Minimum stencil size for the *C*-based height functions is $3\times3\times3$ – one cell of height in which the interface resides, and one cell of height either side to verify that the interface does not reside outside of the central row in the *C*-stencil. For ϕ-based height functions, the height of the ϕ-stencil can be reduced even further. [The stencil compactness achieved in the new scheme is an alternative trend in height function-based curvature scheme development as compared to, for example, the high-order schemes using large stencils (e.g. the $13\times5\times5$ stencil in [42]).]

The additional use of diagonally-oriented meshes in the current work reduces the maximum divergence in angle between the interface normal and the best-oriented height function normal from 45 degrees to 22.5 degrees. Lopez et al. [29] show for a sphere that maximum error is introduced into curvature estimates when the angle between the interface normal and the height function normal is maximal i.e. $\theta = \mathrm{Cos}^{-1}(\max(|n_x|,|n_y|,|n_z|))$, that is 45 degrees when projected onto 2D planes. However, error in the case of a 22.5 degree angle is shown in [29] to be little different to the ideal $\theta = 0$ case. Use of a finite (small) number of conveniently constructed diagonally-oriented meshes is sufficient to extract one curvature candidate in which orientation error is essentially minimized, hence enabling a scheme that samples from the pool of regular-mesh and diagonal-mesh candidates to be considered orientation-independent. [It should be noted that an infinite number of height

functions can be defined, through incremental rotation of the height function stencil. Use of a local mesh stencil that is incrementally rotated – such that the interface normal and height function normal align – is one alternative concept that could be interpreted as achieving "strict" orientation-independence without sample-based consideration of pools of curvature candidates. There is little benefit to optimizing height function orientation, however: (i) mapping regular-mesh VOF data to ever-different local meshes is tedious and inefficient; (ii) there are advantages (to be discussed next) in a sample-based curvature scheme, in that sample processing can "optimize". In the current work, incremental stencil rotation is not considered further.]

2.3.1.2. Candidates and a "best" candidate. The second core prescription in our new method for curvature estimation is, for each interface cell (i,j,k), to seek out the best candidate from within the pool of candidates for that cell. As a sample-based method, the orientation-independent curvature scheme depends on the effectiveness of the decision mechanism for choosing a best candidate from a sample of candidates obtained through different height function definitions. This task has few precedents to date in the literature, and so warrants more detailed discussion here. The discussion here revolves around multiple alternatives for a decision mechanism for best-candidate identification that we have considered.

Dot product method: The scalar product between the normal of the interface and the normal of the height function definition (the direction in which the height function is defined)

$$s = \hat{\mathbf{n}}^{interface} \cdot \frac{\nabla h}{|\nabla h|} \tag{25}$$

is introduced here as the primary quality measure. The "best" candidate is identified as the curvature estimated from the height function that maximizes s.

In reality, the dot-product based on the interface normal in mesh cell (i,j,k) is insufficient on its own to achieve optimization of height function orientation for curvature computation. The decision mechanism is ill-equipped for any efforts at fine-tuned choice between candidates featuring similar values for the dot-product. For candidates from diagonally-oriented height functions, an offset $s^{offset} = 0.2$ is introduced, to ensure that curvature candidates from diagonally-oriented height functions are only chosen to supercede the regular curvature estimate (from the first set only) if the diagonal orientation is clearly more representative of the interface normal direction. Considering an example where $s^{\mathbf{i}+\mathbf{k}} = 0.9 = s^{\max}$, $\kappa^{\mathbf{i}+\mathbf{k}}$ will replace $\kappa^{\mathbf{k}}$ as the best candidate if $s^{\mathbf{k}} = 0.7$ because the $\mathbf{i}+\mathbf{k}$ orientation is clearly more representative of the interface normal. If $s^{\mathbf{k}} = 0.8$ in the same example, then $\kappa^{\mathbf{k}}$ is retained as the best candidate: errors in interface normal estimates mean $\mathbf{i}+\mathbf{k}$ is not clearly superior to $\mathbf{k}$ as the orientation most closely aligned with the interface normal, and the regular height function-based curvature scheme performs rather well for $s^{\mathbf{k}} > 0.8$ in any case.

At first glance, the prospect of having to compute all possible candidates in a sample-based orientation-independent scheme may seem computationally expensive. In the first instance, any increase in computation associated with the sample-based scheme is of little concern to us, because most effort in a timestep of a FD/FV multi-material flow simulation is spent in the Helmholtz (or Poisson) solver for the pressure. In any case, it is worth noting that the dot product is used to reduce the number of height functions and candidates for curvature that are ultimately computed, and hence achieve substantial reductions in computation. Dot products for all height function orientations are computed before any height functions are generated, because low s is used *a priori* to identify poorly-oriented height functions, and hence prevent their computation and avoid wasting computational effort.

Surrounding local average method: The dot-product itself as a quality measure is not infallible: (i) it neglects error in the interface normal estimate; (ii) it is unable to distinguish between candidates from distance function-based and C-based height functions; (iii) it is only a first-derivative of interface location and provide little indication of the quality of the second-derivative estimates that make up the curvature computation. To improve the smoothness of curvature variation over the interface, we can implement a localized outlier filtering based on computation of a surrounding local average. Given an estimate of the curvature distribution, the average curvature in the cells around point (i,j,k) can be computed as

$$\bar{\kappa}_{i,j,k} = \sum_{ii+jj+kk\neq 0} \omega_{i+ii,j+jj,k+kk} \kappa_{i+ii,j+jj,k+kk}, \tag{26}$$

where ω are appropriate weightings of valid curvatures from within the 27-cell stencil around (i,j,k) (zero for $ii = jj = kk = 0$ and $(i+ii, j+jj, k+kk)$ points not within the interface support, equal weightings otherwise summating to unity). This representative estimate of the underlying supergrid-scale interface curvature is then compared to all candidates for the curvature at point (i,j,k); if another candidate is closer to the average than the initial estimate, that estimate is changed to use that new best candidate. Based on the manner in which regular height function curvature estimates are changed, use of the surrounding local average as a decision mechanism is best when smooth spatial variations in the curvature of the underlying interface prevail.

Two-parameter method: Noting the potential of the surrounding local average method to smooth out extrema in curvature, there is merit in considering an alternative that will tend towards preserving extrema if they can be verified to be physical. A quick and efficient means of ascertaining whether a height function curvature estimate is realistic or not is to compute a statistic that is an indicator of curvature variation within the 2D height function stencil. One measure devised for this work is:

$$P = \sum_{j=-1}^{1} \left| \left| \frac{\partial^2 h}{\partial x^2} \right|_j - \overline{h_{cross}} \right| + \sum_{i=-1}^{1} \left| \left| \frac{\partial^2 h}{\partial y^2} \right|_i - \overline{h_{cross}} \right|, \tag{27}$$

where

$$\overline{h_{cross}} = \frac{1}{2} \sum_{j=-1}^{1} \left| \frac{\partial^2 h}{\partial x^2} \right|_j \Bigg| + \frac{1}{2} \sum_{i=-1}^{1} \left| \frac{\partial^2 h}{\partial y^2} \right|_i \Bigg|. \tag{28}$$

Descriptively, Eqs. (27) and (28) indicate to what extent the local height function in a curvature computation is spatially uniform within the 2D height function stencil. High values of P indicate that there is an increased probability that unrealistic second-derivatives of the height function have been computed somewhere in the stencil, that would contaminate any curvature computation based on that height function.

The P statistic for the quality of a height function representation within a stencil is also inadequate in itself for fine-tuned delineation of good and slightly worse curvature candidates. However, a two-parameter method has been seen to be rather predictive in mapping out a parameter space of (s,P) combinations that may identify the best candidate from within the sample of candidates for curvature. More importantly, observations in testing of changes in s and P indicate that relative ratios involving both measures apply across a range of problem configurations, meaning that two-parameter methods involving s and P can be robust. Appendix A describes and summarises useful heuristics for the two-parameter method that have been used in the current work. In summary:

- use of s in the two-parameter method addresses the interface orientation cause of poor curvature estimates identified in [29];
- use of P (or the surrounding local average for that matter) addresses the issue of potential inconsistency in height function estimates identified in [35];
- use of samples of curvature candidates and decision mechanisms facilitate some degree of optimization in poorly-resolved regions of the interface.

2.3.2. Timestep constraint relaxation

The usual capillary timestep restriction on explicit surface tension schemes [4]

$$\delta t_{\text{explicit}} = \sqrt{\frac{\rho_1 + \rho_2}{4\pi\sigma}} \delta x^{3/2} \tag{29}$$

can bottleneck timestep size used in time integration in small-scale flows. Documented ideas for relaxing the timestep restriction on surface tension schemes have included implicit surface tension schemes [17,18] and volume preserving motion by mean curvature [41]. The documented ideas have had mixed success in achieving real improvement in the economy of stable interfacial flow simulation, and unknown success in coupling with 3D VOF interface representations. An alternative strategy is proposed here for relaxing the explicit capillary timestep constraint, that is based on ameliorating instability from the surface force discretization at the disturbance level. By minimizing the propagation of curvature-scheme errors into the surface force update, stability is promoted regardless of whether the explicit capillary timestep constraint is violated. This approach to relaxing the timestep constraint is supported by the finding of Hochstein and Williams [17] that interfacial flow simulation may remain stable even if Eq. (29) is violated.

In the current work, we propose ameliorating the propagation of curvature-scheme errors by introducing a framework of curvature evolution, and by introducing the idea of timescale separation for delineating physical and spurious numerical contributions to curvature evolution. Consider the curvature evolution equation

$$\kappa^{n+1} = \kappa^n + \delta t \overline{\frac{\partial \kappa}{\partial t}}, \tag{30}$$

where $\overline{\frac{\partial \kappa}{\partial t}}$ is a representative change-in-curvature over the time advance $n \to n+1$. The most basic estimate

$$\overline{\frac{\partial \kappa}{\partial t}} \approx \frac{\kappa^{n+1} - \kappa^n}{\delta t} \tag{31}$$

is an explicit update. Knowing the interface configuration (fractional volume (VOF) distributions C^n, C^{n+1}, and/or distance function (level-set) distributions ϕ^n, ϕ^{n+1}) enables an infinite number of (linear) interpolants to be defined: e.g.

$$C^{n+frac} = C^n + frac(C^{n+1} - C^n), \tag{32}$$

where $0 < frac < 1$. For each interpolated location in time, the change-in-curvature can be estimated as either

$$\left. \overline{\frac{\partial \kappa}{\partial t}} \right|_{n+frac}^{f} = \frac{\kappa^{n+frac} - \kappa^n}{t^{n+frac} - t^n} \quad \text{(forward in time)}, \tag{33}$$

or

$$\left.\overline{\frac{\partial\kappa}{\partial t}}\right|^{b}_{n+frac} = \frac{\kappa^{n+1} - \kappa^{n+frac}}{t^{n+1} - t^{n+frac}} \quad \text{(backward in time)}. \tag{34}$$

We use the intuition that $\frac{\partial\kappa}{\partial t}$ varies linearly (given linearly-interpolated interface representations and $\delta t \to 0$) to detect and eliminate possible spurious modes in explicit curvature evolution as they develop.

The scheme we introduce here for relaxing the explicit surface tension timestep restriction is best described as *timescale-separated slope limiting for curvature evolution*. Starting with the curvature at time n, the height function curvature scheme is used to generate explicit "geometry-only" curvature estimates at multiple temporal substeps between times n and $n+1$ using the interface representation at the temporal substeps from Eq. (32). For each of the estimates, the change-in-curvature is computed from Eqs. (33) and (34). The minimum of these change-in-curvature estimates is identified and used to prescribe the curvature evolution that is least likely to be spurious:

$$\left|\left.\overline{\frac{\partial\kappa}{\partial t}}\right|_{n+frac\_min}\right| = min\left(\left|\left.\frac{\partial\kappa}{\partial t}\right|_{cand1}\right|, \left|\left.\frac{\partial\kappa}{\partial t}\right|_{cand2}\right|, \ldots, \left|\left.\frac{\partial\kappa}{\partial t}\right|_{candnum}\right|\right). \tag{35}$$

The final curvature estimate from this curvature evolution over the single timestep $n \to n+1$ is completed as

$$\kappa^{n+1}_{final} = \kappa^{n} + \delta t \overline{\frac{\partial\kappa}{\partial t}}^{n+frac\_min}, \tag{36}$$

and represents a curvature update to $n+1$ that has had most of the spurious contributions to the update identified and removed, leaving the most geometrically-realistic result.

The mix of time increments in Eqs. (33) and (34) used to generate candidates for Eq. (35) represents timescale separation, and is used to detect whether the explicit update to time $n+1$ results in an anomalous local update that would grow into instability. Fig. 4(a) shows the change in 2D PLIC-VOF line–segment interface reconstructions for the case of a near-circle being advected half a mesh cell right in a slightly converging flow, hence involving slight distortion towards elliptical. In reality, $\frac{\partial\kappa}{\partial t}$ is small all over the interface, yet the full explicit update from $n \to n+1$ results in locally large $\frac{\partial\kappa}{\partial t}$ at various locations along the interface (represented at a sample point at the front of the circle in Fig. 4(c)) simply because of spurious subgrid-scale curvature modes being introduced due to imperfect interface representation. In contrast, a relative toggle of the interface from $n \to n + small$ (shown in Fig. 4(b)) results in all interface reconstructions remaining in the same cells, and barely noticeable visual changes to the interface representation beyond the slight rightwards shift and dilation induced by advection. Scaling that incremental curvature evolution over the full timestep eliminates the creation and destruction of VOF interfaces as a source of disturbances that result in local roughness in the spatial curvature distribution. The framework of curvature evolution only applies to individual timesteps; the curvature estimate κ^{n+1}_{final} used in the surface force computation to update velocity $\mathbf{U}^n \to \mathbf{U}^{n+1}$ may differ from the geometry-only estimate κ^{n+1}. After the surface force is computed, κ^{n+1}_{final} is discarded, and the κ^n estimate used in Eq. (36) at the next timestep is the geometry-only estimate κ^{n+1} from the previous timestep. Typically, we use $frac = 0.5$ and $frac = 0.01$, such that the scheme involves five repetitions of the basic geometry-only height function-based curvature module – five candidates in Eq. (35) (three if one-sided estimates (i.e. only one of Eq. (33) and (34)) are used).

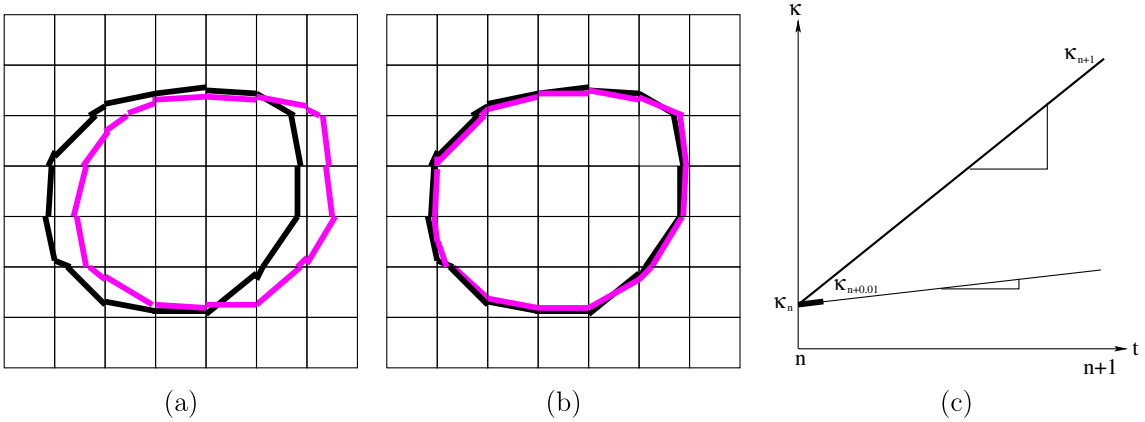

Fig. 4. Timescale-separated slope limiting in curvature evolution: (a) large change in interface configuration in full update from $n \to n+1$ results in large curvature jump in one timestep (as seen in (c)); (b) small change in interface configuration results in small curvature jump from $n \to n+0.01$ – multiplied by 100, the resultant κ^{n+1} estimate is still smaller in this case (but is definitely not the usual case).

The curvature evolution element of the scheme is decoupled from the instantaneous spatial distribution of curvature, in that the final update κ^{n+1} in cell (i,j,k) only depends on explicit κ estimates at other points in time, and does not involve data from a stencil of cells surrounding (i,j,k). The current formulation of the timescale-separated slope limiting for curvature evolution scheme is therefore able to retain local curvature extrema introduced by flow physics. This means that if the height function scheme applied at all interpolants sampled within full timestep δt predict that an extremum in curvature is to be introduced locally, then all sampled $\frac{\partial \kappa}{\partial t}$ gradients will reflect this, and hence will be reflected in the final update κ^{n+1}. Curvature evolution not explicitly depending on the spatial distribution of curvature also makes the scheme weakly monotonic, in that the scheme is not self-healing. For dynamic problems where the desire for less restrictive timestep constraints is moderated by the need to adequately resolve flow transients, we use the timescale-separated slope limiting for curvature evolution to relax the timescale prescribed by the explicit condition Eq. (29), i.e. enable the use of timestep sizes

$$\delta t = f_{cap}\delta t_{\text{explicit}}, \tag{37}$$

where $f_{cap} > 1$. For problems that do not feature large spatial gradients in curvature and large interface motions, we have maintained stability with $f_{cap} \gg 1$. [There is scope for the scheme to feature dependence on the spatial distribution of curvature, i.e. $\frac{\partial \kappa}{\partial t} = f(\kappa^n(\mathbf{x}), \kappa^{n+1}(\mathbf{x}))$. Such a variation would give the scheme a mechanism for the spatial distribution of curvature to locally regain monotonicity, which we conjecture would enable further relaxation of the timestep constraint at the potential expense of information loss regarding physics-induced changes in curvature. This will be investigated in future work.]

The new scheme seeks to remove surface tension stiffness in the spirit of other recent implicit and curvature evolution-based surface tension schemes [18,41]. We consider our scheme to be more focused on identifying and ameliorating the seeds of instability, hence more robust for application to surface tension modelling involving locally high curvatures or the use of relatively noisy interface representations (such as VOF data and inadequately smooth level-set data). [The other schemes are level-set-based and seem to rely on the preservation of initially smooth level-sets. Some degree of treating instabilities at their source is identified in [18] in introducing a diffusion operator to counter destabilization, while no obvious countering of instabilities is identified in [41].]

2.4. MGGMRES 3D elliptic solver

The issue of non-multigrid scaling of iteration count with problem size for discontinuous-coefficient linear elliptic problems is well known. Of specific interest in the current work is the observation that multigrid using bi-/tri-linear interpolation prolongation and restriction operators and the conventional Galerkin coarse-grid operator $A^{L-1} = RA^LP$ converges poorly for large linear Poisson systems [1,7,19,28]. The use of matrix-dependent prolongation and restriction stencils [1,7] has been shown to better preserve multigrid scaling, but its utility has generally been limited to cell-vertex multigrid. Cell-centered multigrid refers to a variation in the multigrid methodology, in which full nesting of coarse-grid cells within fine-grid cells is used; this methodology was popularized for use on 2^n staggered meshes in the CFD community through the work of Wesseling [19,45]). For cell-centered multigrid, the usual Algebraic Multigrid (AMG) paradigm of combining matrix-dependent prolongation and restriction with a Galerkin coarse-grid operator has not been pursued to any significant extent because of the lack of coincidence between fine-grid and coarse-grid mesh-points. Combining cell-centered multigrid with the popular conjugate gradient (CG) Krylov subspace accelerator through use as a preconditioner has also been problematic; the use of bi-/tri-linear interpolation and injection for prolongation and restriction (or vice versa) to satisfy the Order Rule and maintain compactness of Galerkin coarse-grid operator in the cell-centered multigrid (e.g. [37]) makes the preconditioning nonsymmetric.

The use of GMRES [39] in place of CG is effective in making cell-centered multigrid useable as a preconditioner. GMRES acceleration is effective in treating the symptoms of a large spectral radius in inadequately designed cell-centered multigrid that would otherwise decimate convergence of multigrid when it is used as a solver in its own right. However, GMRES acceleration is not sufficient in itself to cure observed problems of poor multigrid convergence. For discontinuous-coefficient elliptic problems, improvement requires reconsideration of aspects of the multigrid preconditioner used within GMRES. In the current work, we present a new variation of the cell-centered multigrid-preconditioned GMRES (MGGMRES) elliptic linear solver that is targeted at achieving robust multigrid-like convergence for the Helmholtz problem (Eq. (16)) and the related Poisson problem featuring large density discontinuities.

2.4.1. Multigrid preconditioner

With reference to the momentum equations in limited compressibility flow scenarios, continuity of the velocity field implies continuity of $\frac{1}{\rho}\nabla P$ across the interface: if the interface represents a discontinuity in ρ, then P is also discontinuous in order to preserve the continuity of $\frac{1}{\rho}\nabla P$. In cell-centered multigrid where coarse-grid mesh cell-centers do not coincide with fine-grid cell-centers, at least one of the multigrid operators (the restriction and prolongation transfer operators, and the coarse-grid operator) will sample data from both sides of the discontinuity. We note the sampling across the discontinuity by different multigrid operators to have different consequences on the approach of the fine-grid solution estimate to the converged solution.

- *Restriction operator* **R**. Applied to the residual $\mathbf{r} = \mathbf{S} - \mathbf{A}\boldsymbol{\phi}$, where the source term distribution $S = \nabla \cdot \mathbf{U}^*/\delta t$ on the fine-grid is continuous across the discontinuity, and $\mathbf{A}\boldsymbol{\phi}$ is continuous across the discontinuity in the limit of the converged solution. The consequences of sampling data from across the discontinuity, and of using regular (non-interface-cognizant) interpolation techniques, are relatively small.
- *Prolongation operator* **P**: Applied to the solution estimate for ϕ, there is scope for regular interpolation techniques to sample across the discontinuity. If ϕ on the coarse grid features a jump at the discontinuity representative of the jump in the converged fine-grid solution, and the fine- and coarse-grid points related through prolongation are on opposite sides of the discontinuity, then prolongation will introduce significant error at those points.
- *Coarse-grid operator* $\mathbf{A}^{L-1}$: Determines the coarse-grid solution that will be prolonged to fine-grid ϕ. If the coarse-grid operator does not result in the coarse-grid ϕ estimate featuring the discontinuity in ϕ of similar location and magnitude to that of converged ϕ on the finest grid, the prolonged solution will introduce significant error into fine-grid ϕ regardless of prolongation operator.

Based of this intuition, improvement of the coarse-grid operator is the top priority, with the intent being to make the $\mathbf{A}^{L-1}$ operator as representative as possible of the discontinuity in $\frac{1}{\rho}$ (that is assumed to be well-captured in the fine-grid operator $\mathbf{A}^L$). In cell-centered multigrid featuring nesting of fine-grid cells within coarse-grid cells, the Galerkin coarse-grid operator is not useful for such a purpose: matrix-dependent **R** and **P** operators are difficult to construct, and the use of at least one wide-stencilled transfer operator (**R** or **P**) to ensure compliance with the Order Rule will invariably smear the discontinuity representation embedded within $\mathbf{A}^{L-1}$. In the current work, we propose a more accurate preservation of the $\frac{1}{\rho}$ discontinuity (both in location and magnitude) be embedded into $\mathbf{A}^{L-1}$. In the context of PDE-based discretization of the coarse-grid operator [45], we discretize equation (16) on all coarse meshes, using densities prescribed by the distance function (RDF) level-set. The coarse-grid operator is

$$\mathbf{A}^{L-1} = \nabla^{L-1} \cdot \frac{1}{\rho^{L-1}} \nabla^{L-1} - \frac{1}{(\delta t \rho^2 c^2)^{L-1}}, \tag{38}$$

where ρ at coarse-grid mesh cell-center (I,J,K) is determined from the local point-wise value (i.e. not from the grid-based FV average) from the location of point (I,J,K) relative to the zero level in the stored finest-mesh distance function:

$$\rho^{L-1}_{I,J,K} = \begin{cases} \rho_G & \text{if } \mathbf{x}_{I,J,K} \text{ within phase } G, \\ \rho_L & \text{if } \mathbf{x}_{I,J,K} \text{ within phase } L. \end{cases} \tag{39}$$

The implementation of a discontinuity-cognizant cell-centered multigrid method is not "black box" in the nature of Algebraic Multigrid.

Discontinuity-cognizance in the coarse-grid operator is key to achieving iteration count scaling reminiscent of perfect multigrid, but it is not beneficial to extend this guideline to the restriction and prolongation operators. Engineering discontinuity-cognizance into the interpolation-based transfer operators presented for cell-centered multigrid in [45] has not been seen to result in any reliable improvement in MGGMRES iteration count. Injection (constant interpolation) is recommended for prolongation because it results in less sampling of data across the discontinuity than higher-order bi-/tri-linear interpolation (that uses a stencil that is twice as wide). Bi-/tri-linear interpolation is used for restriction: it allows the Order Rule for multigrid to be fulfilled, and is well suited to use in restriction because of the continuity of S and $A\phi$ in the converged-solution limit. This is similar to the set of restriction and prolongation operators used in [37], and highlights the fact that only a small modification to conventional documented cell-centered multigrid technologies is required to achieve robustness of multigrid convergence for the discontinuous-coefficient Poisson problem. The multigrid preconditioning is incorporated into the right-preconditioned GMRES algorithm outlined by Saad [38] in the form of Additive Schwarz preconditioning; in the context of parallel computation facilitated by domain decomposition, nesting of coarse-grid meshes is done on non-overlapping subdomains. Finally, Jacobi iterations are used for smoothing.

3. Numerical experiments

To highlight the improvement our new methods give over existing methods, we performed a number of tests to examine the flow solver and the curvature estimation achieved with the presented methods, and then demonstrate that these improvements result in tangible improvements to interfacial flow simulation. The testing is intended to show that volume tracking-based flow simulation that has proven to be useful for free surface flows and other large-scale flow simulations is applicable to surface tension-dominated small-scale flows. It is also intended to show that flow simulation based on this method is convergent, or alternatively degrades gracefully.

3.1. 3D compressible bubble oscillation

The compressible oscillation of a bubble is an important acoustics problem in engineering of broad interest. One interesting applications area from biomedical engineering is the fluid dynamics associated with Ultrasound Contrast Agents (UCA) and cavitation bubbles in the vasculature of live subjects. UCAs are coated microbubbles that are intravenously injected into

animal models or patients for purposes ranging from improving ultrasound imaging inside the body, to the removal of blood clots in stroke patients (sonothrombolysis [8,33]), to facilitating the delivery of drugs for the treatment of Alzheimer's disease by achieving sonoporation of the Blood–Brain Barrier (BBB) [36]. The ultrasound induces a first-order flow in the liquid directly associated with the resonance of the microbubbles [22,30], and may also induce a second-order acoustic microstreaming flow [44]. High-amplitude ultrasound applied to microbubbles can result in their destruction in events that may feature intense wrinkling of the microbubble surfaces and other substantial deviations from sphericity (examples have been captured by photography in [32,44]). The externally-applied ultrasound makes compressibility-induced pressure the driving force of the bubble oscillation, but the surface tension acts to maintain near-sphericity of the UCA microbubbles and also affects observed shape-mode oscillations.

Free small-amplitude oscillation of an overexpanded gas bubble in an unbounded fluid is described by the Rayleigh–Plesset equation [34]

$$R\frac{d^2R}{dt^2}+\frac{3}{2}\left(\frac{dR}{dt}\right)^2=\frac{1}{\rho}\left(p_g-p_\infty-\frac{2\sigma}{R}-4\frac{\mu}{R}\frac{dR}{dt}\right), \tag{40}$$

where R is the instantaneous bubble radius, $V=4/3\pi R^3$ is the instantaneous volume, ρ and μ are the surrounding liquid density and viscosity, σ is the surface tension, γ is the ratio of specific heats for the gas, p_∞ is the ambient pressure, and p_g is the pressure in the bubble that is related to the equilibrium bubble volume V_0 by

$$p_g=p_0\left(\frac{V_0}{V}\right)^\gamma. \tag{41}$$

The problem has previously been solved computationally using various approaches, such as boundary elements [3,20], front tracking-based finite-difference/finite-volume (FD/FV) computation [12], and level-set/VOF FD/FV computation [40]. Common to the bulk of these approaches is the use of an incompressible flow code that models the gas phase by assuming uniformly-distributed vapor pressure, i.e. no gas-sided flow solution.

In this test, the simulation setup is aimed at assessing the validity of the "limited compressibility" methodology introduced by Hirt and Nichols [15] that they proposed to be adequate for acoustics simulation. Specifically, we wish to determine whether the small modifications they prescribe to incompressible flow solver methodologies are adequate for acoustics simulation, or whether a more involved treatment of compressibility is required. The features of interest in the flow solver that affect performance when applied to compressible flows are the absence of the $\mathbf{U}\cdot\nabla P$ term, and the properties of the momentum advection scheme. Densities $\rho_G=1$ and $\rho_L=1000$ and viscosities $\mu_G=\mu_L=0$ are chosen to verify the flow solver to be robust for high density-ratio flows and not adversely affected by numerical dissipation. The small-amplitude volume oscillations for validity of the Rayleigh–Plesset solution involve constant curvature; for flow solver testing, we can hence ignore surface tension by setting $\sigma=0$, such that $P_0=P_\infty$. A bubble of equilibrium bubble radius $R_0=170$ μm is initially overexpanded to $R(0)=171.7$ μm, for a 3% change in mean gas density within the bubble over the oscillation. Based on $\gamma=1.4$, $p_\infty=101325$ Pa and polytropic compression, the initial pressure inside the bubble is set to $p_{g(0)}=97177.8$ Pa. One octant of the sphere was simulated by using symmetry boundary conditions at $x=0$, $y=0$ and $z=0$ in conjunction with pressure boundary conditions at the other domain boundaries. Mesh sizes of $76\times76\times76$ and $112\times112\times112$ were used to keep the computations feasible on the available computing resources. Uniform spacings – $\delta x=6.1111$ μm in the $76\times76\times76$ case and $\delta x=3.4375$ μm in the $112\times112\times112$ case – encompassed the entire bubble and immediate surrounds. Mesh coarsening away from the bubble towards the outer boundaries was used to minimize boundary effects and ensure free oscillation; the domain size spanned by the mesh was no smaller than $32R\times32R\times32R$.

Fig. 5 shows the volume of the bubble compared to the Rayleigh–Plesset solution. The simulated solution captures the periodic contraction and expansion realistically, with good quantitative correspondence achieved using meshes with $>10^6$ mesh cells. Richardson extrapolation of the $nT/16$ predictions by the simulated solutions at the two different mesh resolutions show close correspondence with the Rayleigh–Plesset solution. The result shows the feasibility of capturing the short timescale first-order flow induced by Ultrasound Contrast Agents on moderately-sized meshes. While generally validating the neglect of the $\mathbf{U}\cdot\nabla P$ term as done in the limited compressibility treatment, numerics improvements – such as including the $\mathbf{U}\cdot\nabla P$ term and imposing sharper discontinuity capturing – would be beneficial for converged resolution of the extrema in the oscillation without resort to $O(10^2)$ larger meshes or Adaptive Mesh Refinement.

3.2. 3D sphere curvature test

The sphere is the easiest test of curvature, and arguably the most important as the static equilibrium interface configuration. A sphere of radius 0.25 is initialized within a $1\times1\times1$ computational domain, and the curvature is computed using multiple variations of the height function curvature methodology:

- regular height function (C-based) (default scheme);
- orientation-independent scheme using the dot-product decision mechanism for best-candidate identification (first and second set of candidates only (since the mechanism does not distinguish between candidates from C-based or ϕ-based height functions));

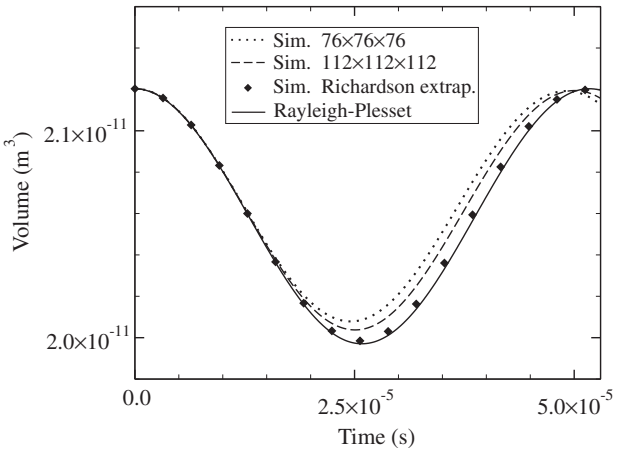

Fig. 5. History of oscillating bubble volume of an initially overexpanded bubble, as compared to the prediction by the Rayleigh–Plesset equation.

- orientation-independent scheme using the two-parameter decision mechanism (first, second and third set of candidates);
- orientation-independent scheme using the surrounding local average decision mechanism (first, second and third set of candidates).

The L_2 error norm is defined as

$$L_2 = \sqrt{\frac{\sum C(1-C)(\kappa - \kappa_{\text{exact}})^2}{\sum C(1-C)}}, \tag{42}$$

where summation occurs over the entire mesh. Along with the results from the different scheme variations listed above, results are also generated using the full pool of candidates (first, second, third and fourth set of values) and with choice of the best candidate based on *a priori* knowledge of the exact curvature. Those results are also tabulated, to identify the boundaries of the orientation-independent approach's potential for improving curvature estimation beyond the regular scheme.

Table 1 shows the errors in reproducing the curvature of a sphere. The regular height function curvature scheme tends towards second-order accuracy as expected. When the sample-based orientation-independent methods are used, error reductions are shown at all lower and medium resolutions regardless of the decision mechanism employed. The differences in performance on the sphere test using the different decision mechanisms are noteworthy. Specifically, the twofold/threefold error reductions using the surrounding local average and the dot-product mechanisms at low/medium resolutions are relatively spectacular compared with the more conservative 10–50% reductions achieved using the two-parameter mechanism. However, it is only the two-parameter mechanism that enables the sample-based orientation-independent curvature scheme to sustain tangible error reductions (relative to the regular height function curvature scheme) for ever-increasing mesh resolutions. At this stage, realizing both large reductions in error at lower resolutions and no deterioration in performance at high resolution requires use of a resolution cut-off based on the apparent radius of curvature, i.e. a $R_{curv}/\delta x = const.$ cut-off.

Table 1 also shows that better estimates of curvature are available across the board when diagonal-mesh candidates are made available. This represents the potential in the sample-based orientation-independent methodology that the decision mechanism for choosing the best candidate is trying to realize. Fig. 6 shows the curvature as a function of the maximum angular deviation away from a 45 degree interface normal orientation: an angle of 45 degrees indicates that the interface normal coincides with a Cartesian axis direction, while an angle of 0 represents an interface normal oriented at 45 degrees from all Cartesian axis directions. The regular height function curvature scheme (shown in Fig. 6(a)) results in multiple

Table 1
L_2 errors in curvature estimation on a 3D sphere using the regular height function scheme and the new orientation-independent height function scheme.

	Regular height function	Orientation-independent height function (best candidate)	Orientation-independent height function (dot-product criterion)	Orientation-independent height function (2-parameter criterion	Orientation-independent height function (surrounding local average)
$24 \times 24 \times 24$	2.81×10^{-1}	1.08×10^{-1}	2.12×10^{-1}	2.47×10^{-1}	1.48×10^{-1}
$48 \times 48 \times 48$	7.26×10^{-2}	1.80×10^{-2}	3.71×10^{-2}	5.40×10^{-2}	2.63×10^{-2}
$96 \times 96 \times 96$	1.74×10^{-2}	4.81×10^{-3}	1.71×10^{-2}	1.14×10^{-2}	6.54×10^{-3}
$192 \times 192 \times 192$	5.35×10^{-3}	1.75×10^{-3}	1.30×10^{-2}	3.98×10^{-3}	2.80×10^{-3}

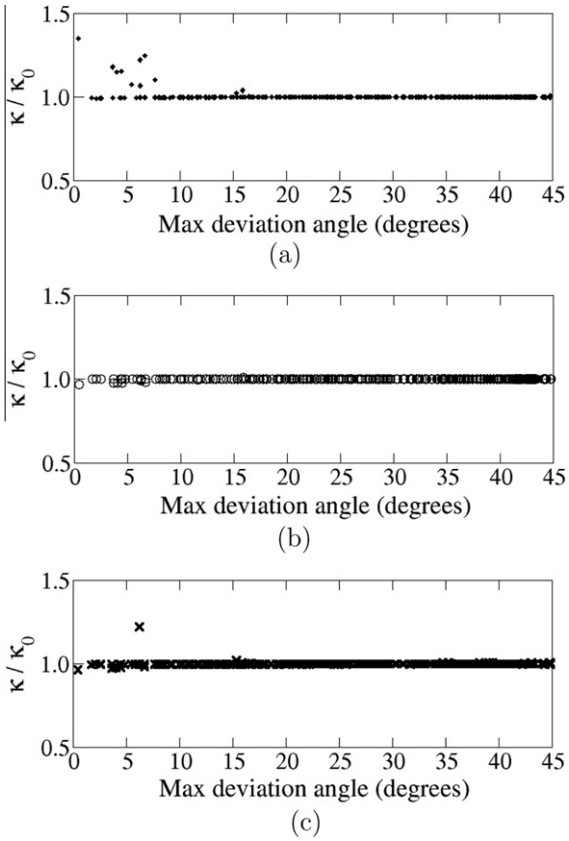

Fig. 6. Curvature estimates from the 3D sphere test on a 192 × 192 × 192 mesh as a function of interface orientation, where the orientation is described as the maximum deviation from a 45 degree orientation between Cartesian axis directions: (a) regular height function curvature scheme; (b) best candidate chosen from sample of candidates; (c) result from orientation-independent scheme.

substantial deviations away from the exact curvature for candidates from near-45 degree interface orientations. Fig. 6(b) shows that a far better estimate is available at each locality, that is able to essentially eliminate the locally high error. The current implementation of the sample-based orientation-independent curvature scheme is shown in Fig. 6(c) to be generally effectively in identifying the superior estimates, albeit not universally infallible. The error reductions that can be achieved across all resolutions through improvement in the decision mechanism are significant, and represent significant motivation for pursuing improved decision mechanisms in future work.

3.3. 3D cosine wave

In the cosine wave test [6], the curvature distribution is not spatially uniform and changes relative to mesh size. In the current work, the test is made 3D by initializing a 2D cosine wave in a plane that does not coincide with those of the mutually orthogonal (3D Cartesian) coordinate axes; in this case, the cosine wave is propagated in the $\mathbf{i}+\mathbf{j}$ direction. For any (x,y) coordinate, the z-coordinate of the cosine wave is initialized as

$$z = A - B\cos\left(\frac{2n\pi w}{L}\right), \tag{43}$$

where w is the projection of the 2D coordinate vector $x\mathbf{i}+y\mathbf{j}$ onto the propagation direction $\mathbf{p}=\mathbf{i}+\mathbf{j}$. The cosine wave using $L=8$, $A=4$, $B=1$ and $n=4$ in Eq. (43) is shown in Fig. 7. The exact curvature corresponding to a 2D cosine wave propagated in the direction of increasing w is

$$\kappa_{\text{exact}} = -\frac{\frac{d^2z}{dw^2}}{\left(1+\left(\frac{dz}{dw}\right)^2\right)^{3/2}} = -\frac{\left(\frac{2n\pi}{L}\right)^2\cos\left(\frac{2n\pi w}{L}\right)}{\left(1+\left(\frac{2n\pi}{L}\right)^2\sin^2\left(\frac{2n\pi w}{L}\right)\right)^{3/2}}. \tag{44}$$

The wave initialization features a mix of interface normals oriented well away from coordinate axis directions, as well as low-curvature and high-curvature regions. The configuration choices used in the current work represent a relatively simple modification of the 2D initialization in [6], designed to highlight the handling of the 45 degree interface orientation that can

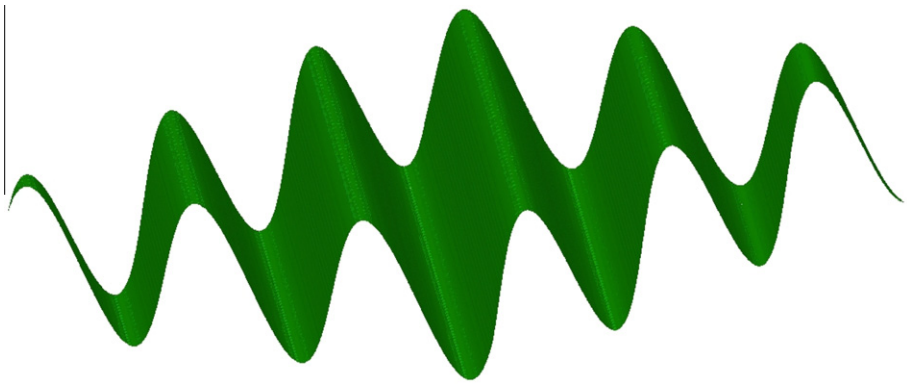

Fig. 7. Initialization of the 3D cosine wave test.

be problematic for the regular height function curvature scheme. In this test, the regular and orientation-independent schemes are compared, with only the two-parameter and surrounding local average decision mechanisms considered for the orientation-independent scheme. [The dot-product is not considered because of the results from the sphere test showing the other options to be superior.] An additional set of results is included, in which the two-parameter decision mechanism is applied to the regular height function curvature scheme. For this set of results, the decision mechanism is only applied to the first set of curvature candidates; this demonstrates that aspects of the decision mechanisms presented in this work can be applied independently of the diagonally-oriented mesh estimates.

Table 2 shows observed proportions of the populations of mixed cells ($0 < C < 1$) that feature curvatures from diagonally-oriented C-based height functions that are better than curvatures from regular height functions. The results show that the curvature candidates from diagonally-oriented height functions are often superior to those from regularly-oriented height functions, and will generally be the majority in areas of the interface oriented away from the coordinate axis directions. The proportions differ depending on whether the C distribution on the diagonally-oriented mesh is obtained through decomposition/reconstitution as is necessary for multi-material flow simulation, or through exact imposition as done when initializing fluid topologies using sub-cell refinement or Monte Carlo methods. Significant improvements in accuracy of curvature estimation are therefore achievable, but dependent on an effective means of identifying traditional height function estimates that are inferior and hence to be swapped with curvature estimates from diagonally-oriented height functions.

While Table 3 confirms the convergence of the regular height function scheme for curvature evaluation to be second-order, its observation with the mesh resolutions considered in the current work is the result of larger curvature errors at the low-resolution end of the spectrum. The key comparison to be made in Table 3 is in the errors between the different col-

Table 2
Proportion of interface cells for which at least one curvature estimate from height functions generated from diagonally-oriented C distributions is superior to the traditional curvature estimate from height functions defined in coordinate axis directions. (second column from right) C distribution on diagonally-oriented meshes imposed through initialization processes on the diagonally-oriented mesh. (Rightmost column) C distribution on diagonally-oriented meshes reconstituted from regular-mesh C distribution using PLIC-VOF interface reconstruction and geometric toolbox to partition truncated volumes of C-fluid into triangular prisms, then re-assembling prisms into diagonally-oriented mesh cells.

Mesh	No. interface cells	Exact diagonal C imposed		Diagonal C reconstituted	
		Cells	%	Cells	%
$24 \times 24 \times 24$	1880	1028	55	802	43
$48 \times 48 \times 48$	8013	3302	41	2657	33
$96 \times 96 \times 96$	31931	8383	26	564	2

Table 3
L_2 errors in curvature estimation on a 3D cosine wave using the regular height function scheme and the new orientation-independent height function scheme.

	Regular height function	Orientation-independent height function (best candidate)	Orientation-independent height function (two-parameter criterion)	Orientation-independent height function (two-parameter criterion	Orientation-independent height function (surrounding local average)
$24 \times 24 \times 24$	1.11×10^{0}	2.90×10^{-1}	7.75×10^{-1}	1.11×10^{0}	1.09×10^{0}
$48 \times 48 \times 48$	1.24×10^{0}	1.47×10^{-1}	8.33×10^{-1}	1.24×10^{0}	1.64×10^{0}
$96 \times 96 \times 96$	2.87×10^{-1}	6.99×10^{-2}	1.58×10^{-1}	2.78×10^{-1}	1.36×10^{0}
$192 \times 192 \times 192$	7.65×10^{-2}	4.14×10^{-2}	7.07×10^{-2}	6.89×10^{-2}	6.97×10^{-2}

umns, and especially at the low-resolution end of the spectrum. When all candidates for the curvature are considered and the best candidate based on *a priori* knowledge of the exact curvature is chosen, the reduction is spectacular at low-resolutions while relatively insignificant at high resolutions; this is reflected in the proportions of the interface-cell population featuring better diagonal candidates shown in Table 2.

Without *a priori* knowledge of the exact curvature, the two-parameter decision mechanism in the orientation-independent scheme is the only one that can (i) tap into the pool of superior curvature estimates from diagonally-oriented meshes at coarser mesh resolutions; (ii) sustain improvement over the regular height function approach at finer resolutions. As shown however by the results of the case where the two-parameter model is applied to the regular height function scheme, the decision mechanism only contributes to low-resolution improvements when there is the pool of candidates available from the diagonally-oriented meshes. At high mesh resolutions, the difference between the regular height function and the orientation-independent curvature schemes diminishes. Indeed at the high resolutions, it is interesting to note that the surrounding local average decision mechanism becomes effective again because $R_{curv}/\delta x$ has increased, and also that the two-parameter decision mechanism is effective even on the regular height function curvature method. Overall, the results support the idea that the orientation-independent scheme is useful in getting the best possible curvature estimates out of compact interface stencils and relatively poor quality interface representations, thus making it useful for real transient multi-material CFD.

3.4. 3D static inviscid drop

The 3D inviscid static-drop was simulated here using the same initialization as used by Francois et al. [9] – a drop of radius 1 and density 1 initialized to be surrounded by fluid of density *0.1* in an $8 \times 8 \times 8$ computational domain on a uniform $40 \times 40 \times 40$ mesh. The first test using this configuration involved simulating static equilibrium of the drop when the analytical curvature is imposed; Francois et al. used this test to determine whether the surface force model discretization and the overall multi-material flow solver was a balanced-force implementation that would preserve static equilibrium. Non-negligible deviations from static equilibrium imply that the flow solver discretization is not balanced-force and hence a source of parasitic modes in itself. In the current work using the MFVOF-3D solver, simulation to $t = 4.0 \times 10^{-3}$ with five timesteps using the momentum formulation of the momentum equations (discretization of Eq. (11)) results in a maximum velocity component of $|u_i|_{\max} = 7.16 \times 10^{-16}$: this magnitude is of the order of machine accuracy, and verifies the flow solver to feature a balanced-force discretization of the surface force. Use of the velocity formulation as the basis of discretization (Eq. (11) divided through by ρ prior to discretization) did not preserve static equilibrium.

In the second test, the curvature was calculated using both the regular height function curvature scheme, and the orientation-independent height function scheme. The surrounding local average decision mechanism was used here, because the testing presented previously in the curvature tests shows it to be the best mechanism for application to low-curvature problems. Table 4 shows that the orientation-independent height function more than halves the intensity of parasitic currents introduced into simulated static-drop solutions due to curvature errors. The numbers in Table 4 also compare favorably to those presented in [9]. This result shows that the improvements in curvature estimation have tangible benefits on overall flow solver accuracy.

Timesteps of the second test were also used for recording CPU timings, to provide an indication of the actual computational effort associated with the new orientation-independent curvature scheme; the results are shown in Table 5. The orientation-independent scheme itself is inexpensive relative to the other key modules in the solution algorithm; if the interface indicator data such as the C distribution or distance function level-set is already available, then the extra computational cost associated with upgrading to the orientation-independent surface tension scheme is small. In the case of the diagonal-mesh C distributions also needing to be generated, the expense of the module for generating the diagonal-mesh C distributions from the regular C distribution is 50–75% on top of the VOF module that features the objects required to facilitate the computation. The numbers are similar if the RDF generation scheme is also used, but with poorer scaling of computational effort for RDF generation than for diagonal-mesh C generation. In any case, solving the Poisson problem remains the most expensive part of the solution algorithm. As well as remaining more expensive than each and all components required for the orientation-independent surface tension module, the Poisson solver also scales worse than the surface tension module numerics. In summary, the timings prove use of the orientation-independent surface tension module to not be computationally burdensome, hence making it feasible for use in generic interfacial flow simulation.

Table 4
Maximum cell-face-velocity component in 3D static-drop test after the first timestep and after 50 timesteps, where $\delta t = 10^{-3}$, using the Sharp Surface Force method [9] and the regular and orientation-independent height function schemes.

Time	Regular height function	Orientation-independent height function
Δt	3.72×10^{-3}	2.30×10^{-3}
$50\Delta t$	6.98×10^{-2}	3.35×10^{-2}

Table 5
CPU timings for various numerical schemes within a timestep of the MFVOF-3D solution algorithm applied to the static-drop test. The timed components include the orientation-independent surface tension scheme, the schemes for generating diagonally-oriented C distributions and RDFs, the 3D PLIC-VOF interface tracking module, and a standard Incomplete Cholesky conjugate gradient ICCG(0) Poisson solver.

Module	$40 \times 40 \times 40$		$80 \times 80 \times 80$	
	CPU time (s)	CPU time (% of timestep)	CPU time (s)	CPU time (% of timestep)
Full timestep	6.56	100.0	47.46	100.0
Surface tension core scheme	0.48	7.3	1.87	3.9
Diagonal-mesh C generation	1.03	15.7	4.01	8.4
RDF generation	0.55	8.4	5.11	10.8
VOF	1.42	21.6	7.32	15.4
Poisson solver	2.56	39.0	26.32	55.5

3.5. 3D incompressible drop oscillation

The incompressible drop oscillation problem is used here as a test of the timestep constraint relaxation achieved using the timescale-separated slope limiting for curvature evolution scheme. If a drop is initialized to be distorted away from a spherical shape by distortion in one axis direction relative to the other two axis directions, the drop will oscillate with an angular frequency [21]

$$\omega = \sqrt{\frac{\sigma n(n+1)(n-1)(n+2)}{[(n+1)\rho_1 + n\rho_2]R^3}}, \tag{45}$$

where $n = 2$ is the first mode of oscillation, σ is the surface tension, and R is the radius of the drop in static equilibrium. In the problem setup, a drop is initially distorted away from an equilibrium shape according to the function [29]

$$R = R_0[1 + \eta P_2(\cos\phi)], \tag{46}$$

where R_0 is the equilibrium drop radius, η is the amplitude of the disturbance away from sphericity, ϕ is the polar angle, and P_2 is the Legendre polynomial of order 2. Additionally for the viscous case, the amplitude of the oscillation is [21,29]

$$a(t) = \eta \exp\left(-\frac{5\mu_1}{\rho_1 R_0^2} t\right) \cos(\omega t). \tag{47}$$

The same setup parameters as used in [29] are used: drop density $\rho_1 = 1$, drop viscosity $\mu_1 = 10^{-2}$, surrounding fluid density $\rho_2 = 10^{-2}$, surrounding fluid viscosity $\mu_1 = 10^{-4}$, surface tension 1, $R_0 = 1$, $\eta = 10^{-2}$. The drop is initialized in a domain with free-slip boundary conditions located $2R$ from the center of the drop, which is adequate for negating wall effects. In the test, a resolution of $\delta x = 4/64$ and the standard height function scheme are used.

Fig. 8 shows that the timescale-separated slope limiting for curvature evolution is able to maintain stability when the explicit capillary timestep condition is violated. In contrast, using $f_{cap} = 1.5$ in an explicit update resulted in the simulation being destroyed by instability. We note that the timestep prescribed by Eq. (37) is the bottleneck on timestep size in this simulation, with the viscous and CFL timestep conditions being one and two orders of magnitude larger, respectively. Lopez et al. [29] noted the capillary condition to be the most restrictive constraint on timestep in their simulation, and used $f_{cap} = 0.35$ in Eq. (37) in their work for stability. Compared to the analytical solution for the decay in oscillation amplitude

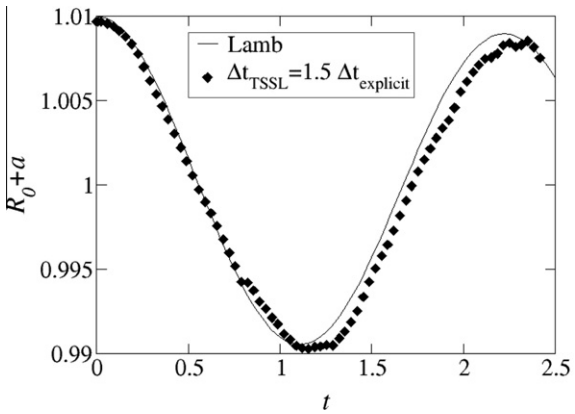

Fig. 8. Amplitude of 3D incompressible drop oscillation generated using $f_{exp} = 1.5$ and the timescale-separated slope limiting for curvature evolution.

(Eq. (47)), the $f_{cap} = 1.5$ result compares well over the majority of the oscillation period when the interface is advancing or receding. Fidelity of the peaks and troughs in the oscillation where flow reversal occurs requires proper resolution in the time integration (noted for compressible bubble oscillation in [12]). During such stages of the oscillation, reducing f_{cap} to levels used in explicit updates is warranted.

3.6. 3D bubble rise

3.6.1. Larger bubble

As a first case study, we consider an experiment performed by Hnat and Buckmeister [16] that was subsequently simulated by Sussman and Ohta [41]. The setup involves a bubble of equivalent-sphere diameter $d_e = 1.215$ cm, density $\rho = 1.262$ kg/m^3 and viscosity 1.80×10^{-5} N s/m^2 rising under buoyancy (the result of gravitational acceleration $g_z = -9.81$ m/s) through a liquid of density $\rho = 876$ kg/m^3 and viscosity 1.18×10^{-1} N s/m^2. The surface tension of the bubble is $\sigma = 3.22 \times 10^{-2}$ N/m. The terminal velocity measured in [16] is $U_{\text{exp}} = 0.215$ m/s, resulting in Reynolds number $Re_{\text{exp}} = 19.4$ and Eötvös number $Eo_{\text{exp}} = 3.93$. The bubble is moderately large, and its rise is within the spherical cap regime [5]. The largeness of the bubble means inertia is significant relative to surface tension, obviously making the advective timestep the bottleneck on timestep size and hence reducing the motivation for capillary timestep relaxation. On the other hand, the known bubble shape of terminal rise from the experiment features locally high curvatures and inversion in bubble curvature from convex to concave, and so is more of a challenge to curvature spatial discretizations.

The simulation presented here is of a single quadrant in the xy-plane: the $(x,y) \geqslant 0$ quadrant, with the bubble centered at $(0,0)$ and gravity acting in the z-direction. Symmetry conditions were used at $x = 0$ and $y = 0$. The simulation to terminal rise was performed on a $48 \times 48 \times 96$ non-uniform mesh, with multiple restarts of the simulation involving transferring the flow field around the bubble from the upper half of the domain to the lower half. In this manner, the progression of the bubble to its final shape and terminal velocity is completed more rapidly. The non-uniformity of the mesh in the (x,y) plane featured uniform cells of breadth $\delta x = \delta y = 3.5 \times 10^{-4}$ m in the vicinity of the bubble, that broadened away from the bubble to be $\delta x = 6.6 \times 10^{-3}$ m adjacent to the $x = X$ and $y = Y$ boundaries. The mesh spacing in the z-direction was uniform with $\delta z = 3.5 \times 10^{-4}$ m. The domain for zeroing in on terminal velocity was of dimensions 4.86×10^{-2} m $\times$ 4.86×10^{-2} m $\times$ 6.72×10^{-2} m ($8r_e \times 8r_e \times 11r_e$), and the mesh size was $48 \times 48 \times 192$.

Fig. 9 shows the simulated 3D bubble surface at an instant in time after terminal velocity is reached. The bubble profile obtained here from simulating in the $(x,y) \geqslant 0$ quadrant corresponds qualitatively in shape and terminal velocity with full-bubble (all-quadrant) simulation performed using lower resolution. The use of the non-uniform mesh and simulation only in the $(x,y) \geqslant 0$ quadrant is beneficial in helping adequately resolve the high curvature near the rim. The profile also shows strong correspondence with the experimental profile from [16] and the numerical simulation profile obtained in [41]. The terminal velocity of the simulated rising bubble is 0.212 m/s, representing good agreement with the experimental terminal velocity 0.215 m/s.

3.6.2. Smaller bubble

As a second case study, we consider the case of a $d_e = 3$ mm air bubble bursting at a water surface, featuring an Eötvös number $Eo = 1.19$ and a Morton number $M = 10^{-10.59}$. Given bubble size is much smaller and surface tension is larger, the relative contribution of inertia is smaller, and the time integration of the surface force contribution is a greater challenge to the stability of the multi-material flow solver. The rise of a 3 mm bubble resides in the ellipsoidal regime [5], hence bubble shape features less spatial variation in curvature. As such, this smaller bubble is more of a challenge to the timestep relaxation achieved using the timescale-separated slope limiting for curvature evolution than the previous larger bubble case.

Bubble rise in the z-direction is simulated using the first quadrant in the xy-plane on a non-uniform $32 \times 32 \times 512$ mesh until statistically steady-state bubble rise is achieved. Being in the ellipsoidal regime, bubble shape and rise velocity can be expected to feature small fluctuations about the time-averages. The non-uniformity of the mesh in the xy-plane is such that the finest-mesh spacing $\delta x_{finest} = \delta y_{finest} = \delta z_{finest} = 150$ μm is used to resolve the bubble and surroundings. Coarsening away

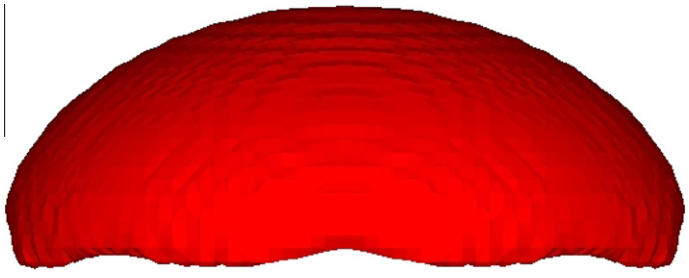

Fig. 9. Bubble profiles at terminal velocity, for the simulation of the $d_e = 1.215$ cm bubble rise experiment of Hnat and Buckmeister [16] in a $8r_e \times 8r_e \times 11r_e$ domain on a non-uniform $48 \times 48 \times 192$ mesh.

from the bubble rise core of the domain ensures that the overall domain is sufficiently large to minimize wall effects; the domain size in the *xy*-plane is $10d_e \times 10d_e$.

Fig. 10(a) shows bubble profiles at 0.01 s intervals generated by the MFVOF-3D code. Once upward motion of the bubble is established, the transformation of bubble shape to ellipsoidal is relatively rapid, occurring in about 0.07 s over a distance of about $4d_e$. With regards to the experimental terminal velocity datasets for air/water systems collated in graphical form by Clift et al. [5], it is worth noting the envelope of terminal velocities that exists for the ellipsoidal regime inclusive of the 3 mm diameter considered in the current work. For any bubble diameter on their terminal velocity plot, the velocities on the upper side of the envelope spanning the ellipsoidal regime correspond to relatively pure air/water systems, while the lower side of the envelope represents impure systems in which surfactants in the experimental system accumulate at the interface. In impure systems, the surfactant acts to immobilize the interface of the air bubble in water, thereby slowing internal circulation within the air bubble and hence reducing terminal velocity. On the experimental datasets graph in [5], the data most representative of pure systems are those aligned with the upper-envelope curve and featuring the highest terminal velocities; this data also contains visible scatter, which Alves et al. [2] identify to coincide with issues such as the use of different water purification techniques and different rates of bubble dissolution. Fig. 10(b) shows the rise velocity to stabilize at a terminal velocity of $U_T \approx 29.0$ cm/s. This predicted terminal velocity compares very well with specific experimental data points in the experimental data set collated by Clift et al. [5]. We note the possible slight overprediction of terminal velocity (compared to the oft-quoted $25 - 28$ cm/s figures for clean systems) due to the use of computation-saving vertical symmetry planes removing the scope for translational horizontal modes of motion such as the zig-zag rise trajectories. In any case, the prediction is well within the experimental error associated with the scatter seen in collated experimental data sets (e.g. in [5,2]). With regards to the aspect ratio from simulation shown in Fig. 10(b), the averaged value corresponding to terminal rise of 0.46 is also well within the experimental data range presented in [5]. The replication of experimentally-observed terminal velocity and aspect ratio validates the multi-material flow solver in general, and the surface tension scheme to accurately and robustly model interfacial flows even with relatively low-resolution.

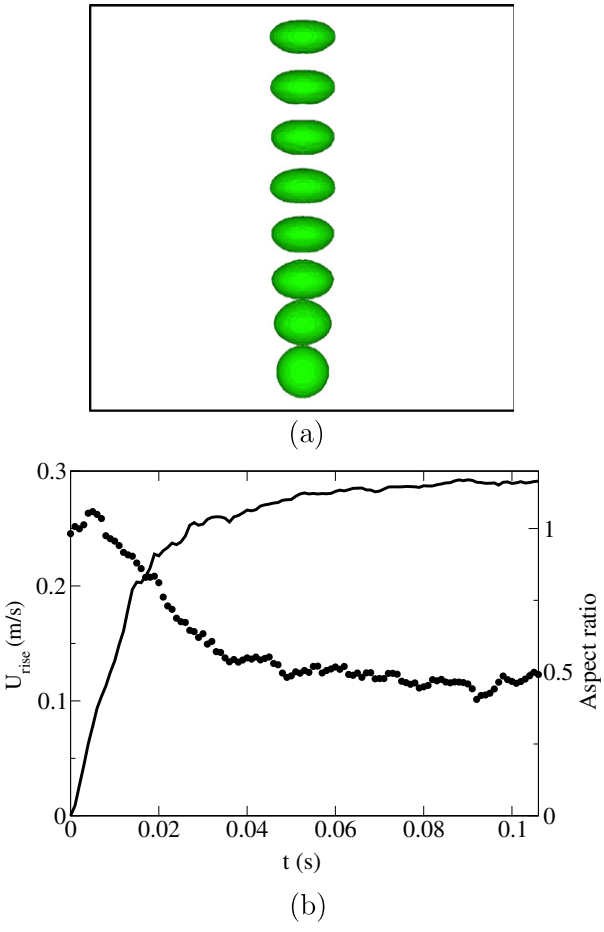

Fig. 10. (a) Profiles during the rise of a 3 mm air bubble in water, showing evolution to a terminal rise shape; (b) time histories of rise velocity and aspect ratio captured by simulation with the MFVOF-3D code.

3.7. MGGMRES solver

During the course of the 3D compressible oscillating bubble and incompressible oscillating drop simulations, MGGMRES solver performance was recorded to demonstrate the efficiency and robustness of discontinuity-cognizant cell-centered MGGMRES. Fig. 11(a) shows the timings for this method compared to cell-centered MGGMRES using the standard Galerkin coarse-grid operator, generated on the compressible oscillating bubble problem. The discontinuity-cognizant method performs far better than the standard Galerkin cell-centered multigrid method, with the main saving being in not having to construct that Galerkin coarse-grid operator. Fig. 11(a) shows little variation in CPU times for convergence over the course of the oscillation, and Fig. 11(b) from the incompressible oscillating drop simulation shows convergence with multigrid-like V-cycle counts is achieved for discontinuous-coefficient problems. The reduced CPU times for convergence also make multigrid useful at lower resolutions, because the crossover at which the multigrid-based solver becomes more efficient than non-multigrid Krylov subspace Poisson/Helmholtz solvers (e.g. ICCG(0)) occurs at a lower problem size.

Given the new numerical methods for curvature discretization help enable the generation of high-fidelity solutions with moderate computing requirements, the bubble rise test problems treated in the current work were not large-mesh computations and hence not a major test of the MGGMRES solver on large elliptic problems in multi-material flow CFD. A better indication of the performance of the MGGMRES solver is provided by considering a flow problem that requires large meshes even for basic resolution requirements. One such problem is 3D coastal wave breaking – a flow that has been solved previously using the MFVOF-3D code (refer to [25] for visualizations of wave breaking simulated by the code). As a test of the elliptic solver, wave-propagation towards breaking at an upward-sloping sea bed by a cnoidal wave is simulated. The domain size 26.9 m $\times$ 0.4 m $\times$ 1.25 m is resolved by a uniform 1024 $\times$ 48 $\times$ 192 mesh, using a 64 CPU computation based on a 16 $\times$ 1 $\times$ 4 domain decomposition. In the problem setup, a flat free surface separating the air and water phases is initialized to connect (x,z) coordinates $(0,0.646)$ and $(22.6,0)$, and gravitational acceleration acts in the direction $(g_x,g_y,g_z) = (-0.28,0,-9.80)$. Wave propagation is driven by a cnoidal wave generator [11], which imposes fixed inflow/outflow

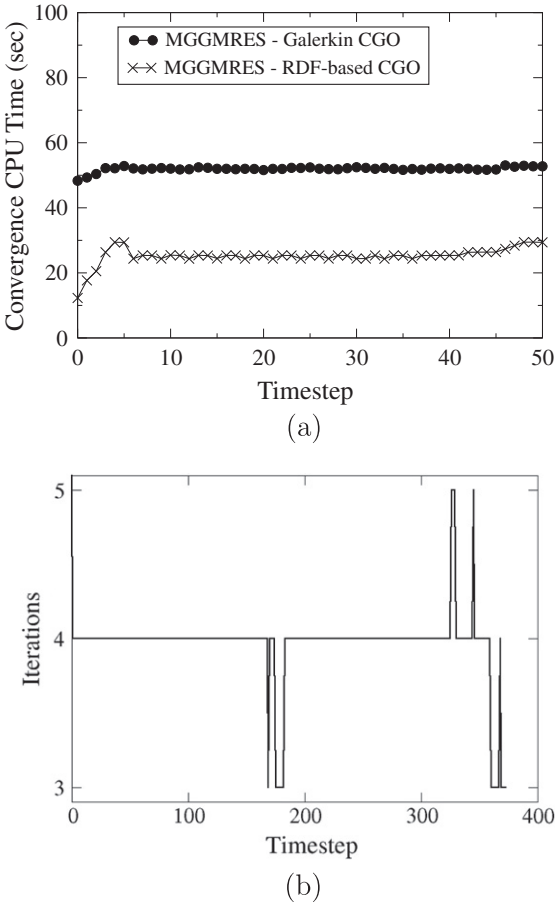

Fig. 11. Convergence histories for the multigrid-preconditioned GMRES solver on the oscillating bubble problems presented in the paper: (a) history of CPU time for multigrid-preconditioned GMRES solver convergence in the compressible oscillating bubble problem using cell-centered multigrid featuring different coarse-grid operators; (b) history of iteration count for multigrid-preconditioned GMRES solver convergence in the incompressible oscillating drop problem using the discontinuity-cognizant coarse-grid operator in cell-centered multigrid preconditioning of the MGGMRES solver.

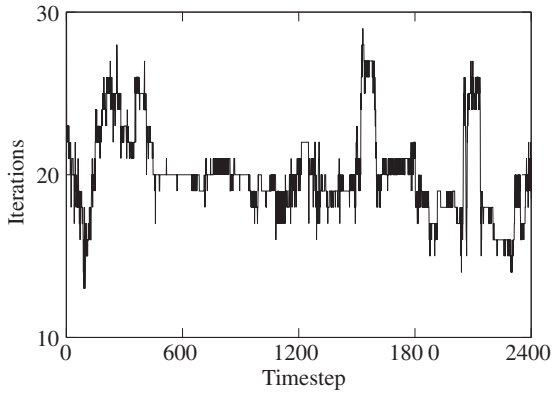

Fig. 12. Convergence histories (in terms of multigrid V-cycle iteration count) for a large-scale wave-propagation/wave-breaking problem simulated using the MFVOF-3D code, using the discontinuity-cognizant coarse-grid operator in cell-centered multigrid preconditioning of the MGGMRES solver.

boundary conditions at the offshore $x = 0$ boundary; the corresponding open boundary is located at $x = 26.9$ m. The MFVOF-3D code solves this as an incompressible flow problem, resulting in solution of a Poisson equation for the pressure correction. Fig. 12 shows the multigrid V-cycle count for this problem converged to a 10^{-5} criterion at each timestep. The iteration counts represent the $O(10)$ counts expected of well-performed multigrid solvers, and this performance is sustained over many thousands of timesteps – a reflection of MGGMRES solver robustness on large free surface flow problems. This performance is transferable across the range of multi-material flow problems, from large-scale free surface flows to small-scale surface tension-dominated interfacial flows. In contrast, thousands of ICCG(0) iterations are required to converge the Poisson problem in this simulation.

4. Conclusion

This paper presents a new scheme for curvature computation in surface tension models, based on addressing the relatively poor quality curvature estimates generated by past height function approaches when the interface normal is oriented away from coordinate axis directions. The use of height functions defined in diagonal directions between the orthogonal coordinate axis directions ensures at least one candidate for the interface curvature at any point is based on a height function that is oriented in a similar direction to the interface normal. The use of a dot-product criterion is primarily responsible for correcting bad regular curvature estimates with better candidates from diagonally-oriented height functions, and additional operations such as outlier filtering can further help in zeroing in on a "best" estimate. The Reconstructed Distance Function level-set is also useful for facilitating feasible 3D simulation by enabling effective multigrid coarse-grid operators to be constructed with less computation, thereby reducing the wall time required to converge elliptic problems in transient multi-fluid solution algorithms. This reduction in per-timestep computation is complemented by the reduction in the number of timesteps required to complete time integration that is attained by using curvature evolution to ameliorate the transfer of errors in curvature into the flow solution. Simulations of dynamic flow scenarios were able to remain stable even though the explicit capillary timestep restriction was violated; this verifies the intuition of our methodology in detecting and eliminating spurious modes in curvature evolution. The improved methods presented in this paper are practical and are useful for incremental retrofitting of existing interface tracking-based flow solvers to handle 3D interfacial flow problems.

Acknowledgments

This work was supported by the NCI National Facility at the Australian National University www.nci.org.au. PL and RM acknowledge the support of the Preventative Health Flagship of the Commonwealth Scientific and Industrial Research Organization (CSIRO). MF acknowledges the support of the US Department of Energy ASC and LDRD programs at Los Alamos National Laboratory under contract DE-AC52-06NA25396 with Los Alamos National Security, LLC for the National Nuclear Security Administration.

Appendix A. Two-parameter method as a best-candidate decision mechanism

For use as decision mechanisms for best-candidate identification out of samples of curvature candidates obtained using different height functions, Eqs. (25) and (27) individually feature significant shortcomings. It is proposed and tested in the current work, however, that use of Eqs. (25) and (27) together can yield good performance – sizeable reductions in error over the entire range of mesh resolutions. Below is the algorithm (with concise explanation) for the two-parameter decision mechanism.

(0) Set up indexing l for entries into sample of candidates for curvature:
normal C-based candidates: $1 \leqslant l \leqslant 6$;
diagonally-oriented mesh C-based candidates: $7 \leqslant l \leqslant 18$;
normal ϕ-based candidates: $19 \leqslant l \leqslant 24$;
diagonally-oriented mesh ϕ-based candidates: $25 \leqslant l \leqslant 36$.
(1) Compute dot-product s_l for all candidates, i.e. s_1 to s_{36}.
(2) Disqualify all candidates featuring $s_l > 0.25$, i.e. set $\kappa_l = large$.
(3) Compute second-derivative variation statistic P for remaining candidates.
(4) Sweep through all remaining candidates:
Store data (κ, s, P) for each pool of candidates that have:
(i) highest value of s;
ii) lowest value of P;
Values stored are:

$$s^{Cnorm}_{s\text{-}crit}, s^{\phi norm}_{s\text{-}crit}, s^{Cdiag}_{s\text{-}crit}, s^{\phi diag}_{s\text{-}crit}, P^{Cnorm}_{s\text{-}crit}, P^{\phi norm}_{s\text{-}crit}, P^{Cdiag}_{s\text{-}crit}, P^{\phi diag}_{s\text{-}crit},$$
$$s^{Cnorm}_{P\text{-}crit}, s^{\phi norm}_{P\text{-}crit}, s^{Cdiag}_{P\text{-}crit}, s^{\phi diag}_{P\text{-}crit}, P^{Cnorm}_{P\text{-}crit}, P^{\phi norm}_{P\text{-}crit}, P^{Cdiag}_{P\text{-}crit}, P^{\phi diag}_{P\text{-}crit}$$

(5) Sweep through all remaining candidates:
Disqualify candidate l if:
(a) $S_l < S^{Cnorm}_{P\text{-}crit}$ and $P_l > P^{Cnorm}_{P\text{-}crit}$;
(b) $S_l < S^{Cnorm}_{s\text{-}crit}$ and $P_l > P^{Cnorm}_{P\text{-}crit}$;
(c) $P_l > aP^{Cnorm}_{s\text{-}crit}$;
(d) $P_l < bP^{Cnorm}_{s\text{-}crit}$;
(e) $|\kappa_l| > c|\kappa^{Cnorm}_{s\text{-}crit}$;
(f) $\kappa_l \kappa^{Cnorm}_{s\text{-}crit} < 0$ and $|\kappa_l| > d|\kappa^{Cnorm}_{s\text{-}crit}$.
(6) Compute *samplesize* of valid entries.
(7) if *samplesize* $\geqslant 2$ then: (multiple valid alternative candidates)
sweep through candidate other than from *Cnorm* pool
if $S_l > S^{Cnorm}_{s\text{-}crit}$ and $P_l < P^{Cnorm}_{s\text{-}crit}/e$
$diffmin = min(diffmin, |\kappa_l - \kappa|)$
if $(diffmin = |\kappa_l - \kappa|)\kappa_{diffmin} = \kappa_l$
end if
end sweep
$\kappa = \kappa_{diffmin}$
end if
if *samplesize* = 1 then: (one valid alternative candidate)
if $S_l > S^{Cnorm}_{s\text{-}crit} + f$ and $P_l < P^{Cnorm}_{s\text{-}crit}/g$
$\kappa = \kappa_l$
end if
$\kappa = \kappa_{diffmin}$
end if
if *samplesize* = 0 then: (all valid alternative candidates eliminated – revisit disqualified list)
$diffmin = min(|\kappa_l - \kappa^{Cdiag}_{s\text{-}crit}|, |\kappa_l - \kappa^{\phi norm}_{s\text{-}crit}|, |\kappa_l - \kappa^{\phi diag}_{s\text{-}crit}|)$
if $diffmin = |\kappa_l - \kappa^{Cdiag}_{s\text{-}crit}| \kappa_{diffmin} = \kappa^{Cdiag}_{s\text{-}crit}$
if $diffmin = |\kappa_l - \kappa^{\phi norm}_{s\text{-}crit}|\ \kappa_{diffmin} = \kappa^{\phi norm}_{s\text{-}crit}$
if $diffmin = |\kappa_l - \kappa^{\phi diag}_{s\text{-}crit}|\ \kappa_{diffmin} = \kappa^{\phi diag}_{s\text{-}crit}$
if $|\kappa_{diffmin}| > h|\kappa|$ and $|\kappa_{diffmin}| < i|\kappa|$
$\kappa = \kappa_l$
end if
$\kappa = \kappa_{diffmin}$
end if
where $a = 2.5$, $b = 0.1$, $c = 1.5$, $d = 0.5$, $e = 2.5$, $f = 0.15$, $g = 1.5$, $h = 0.5$, $i = 1.5$

References

[1] R.E. Alcouffe, A. Brandt, J.E. Dendy Jr., J.W. Painter, The multi-grid method for the diffusion equation with strongly discontinuous coefficients, SIAM J. Sci. Stat. Comput. 2 (1981) 430–454.
[2] S.S. Alves, S.P. Orvalho, J.M.T. Vasconcelos, Effect of bubble contamination on rise velocity and mass transfer, Chem. Eng. Sci. 60 (2005) 1–9.
[3] J.R. Blake, M.C. Hooton, P.B. Robinson, R.P. Tong, Collapsing cavities, toroidal bubbles and jet impact, Philos. Trans. Roy. Soc. Ser. – A 355 (1997) 537–550.
[4] J.U. Brackbill, C. Zemach, D.B. Kothe, A continuum method for modeling surface tension, J. Comput. Phys. 100 (1992) 335–354.
[5] R. Clift, J.R. Grace, M.E. Weber, Bubbles, Drops and Particles, Academic Press, 1978.
[6] S.J. Cummins, M.M. Francois, D.B. Kothe, Estimating curvature from volume fractions, Comput. Struct. 83 (2005) 425–434.

[7] J.E. Dendy Jr., Black box multigrid, J. Comput. Phys. 48 (1982) 366–386.
[8] J. Eggers, G. Seidel, B. Koch, I.R. König, Sonothrombolysis in acute ischemic stroke for patients ineligible for rt-PA, Neurology 64 (2005) 1052–1054.
[9] 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 (2006) 141–173.
[10] M.M. Francois, B.K. Swartz, Interface curvature via volume fractions, heights, and mean values on nonuniform rectangular meshes, J. Comput. Phys. 229 (2010) 527–540.
[11] D.G. Goring, Tsunamis – the propagation of long waves onto a shelf, Ph.D. Dissertation, California Institute of Technology, 1979.
[12] Y. Hao, A. Prosperetti, A numerical method for three-dimensional gas–liquid flow computations, J. Comput. Phys. 196 (2004) 126–144.
[13] J. Helmsen, P. Colella, E.G. Puckett, Non-convex profile evolution in two dimensions using volume of fluids, Technical Report LBNL-40693, Lawrence Berkeley National Laboratory, 1997.
[14] C.W. Hirt, B.D. Nichols, Adding limited compressibility to incompressible hydrocodes, J. Comput. Phys. 34 (1980) 390–400.
[15] C.W. Hirt, B.D. Nichols, Volume of fluid (VOF) method for the dynamics of free boundaries, J. Comput. Phys. 39 (1981) 205–226.
[16] J.G. Hnat, J.D. Buckmeister, Spherical cap bubbles and skirt formation, Phys. Fluids 19 (1976) 182–194.
[17] J.I. Hochstein, T.L. Williams, An implicit surface tension model, in: Proceedings of the 34th Aerospace Sciences Meeting and Exhibit, Reno, NV, January 15–18, AIAA 96-0599, 1996.
[18] S. Hysing, A new implicit surface tension implementation for interfacial flows, Int. J. Numer. Methods Fluids 51 (2005) 659–672.
[19] M. Khalil, P. Wesseling, Vertex-centered and cell-centered multigrid for interface problems, J. Comput. Phys. 98 (1992) 1–10.
[20] E. Klaseboer, B.C. Khoo, Boundary integral equations as applied to an oscillating bubble near a fluid–fluid interface, Comput. Mech. 33 (2004) 129–138.
[21] H. Lamb, Hydrodynamics, Cambridge University Press, 1932.
[22] T.G. Leighton, The Acoustic Bubble, Academic Press, 1994.
[23] P. Liovic, D. Lakehal, Large Eddy Simulation of steep water waves, IUTAM Symposium on ‘‘Computational approaches to disperse multiphase flow, Argonne National Laboratory, Argonne, Illinois, October 4–7, 2004.
[24] P. Liovic, D. Lakehal, Interface–turbulence interactions in large-scale bubbling processes, Int. J. Heat Fluid Flow 28 (2007) 127–144.
[25] P. Liovic, D. Lakehal, Multi-physics treatment in the vicinity of arbitrarily deformable gas–liquid interfaces, J. Comput. Phys. 222 (2007) 504–535.
[26] P. Liovic, D. Lakehal, A Newton–Krylov solver for remapping-based Volume-of-Fluid methods, SIAM J. Sci. Comput. 31 (2008) 865–889.
[27] P. Liovic, M. Rudman, J.-L. Liow, D. Lakehal, D.B. Kothe, A 3D unsplit-advection volume tracking algorithm with planarity-preserving interface reconstruction, Comput. Fluids 35 (2006) 1011–1032.
[28] C. Liu, Z. Liu, S. McCormick, An efficient multigrid scheme for elliptic equations with discontinuous coefficients, Commun. Appl. Numer. Methods 8 (1992) 621–631.
[29] J. Lopez, C. Zanzi, P. Gomez, R. Zamora, F. Faura, J. Hernandez, An improved height function technique for computing interface curvature from volume fractions, Comput. Methods Appl. Mech. Eng. 198 (2009) 2555–2564.
[30] R. Manasseh, A. Ooi, Frequencies of acoustically interacting bubbles, Bubble Sci. Eng. Technol. 1 (2009) 58–74.
[31] 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 (2007) 949–974.
[32] S.M. van der Meer, B. Dollet, D.E. Goertz, N. de Jong, M. Versluis, D. Lohse, Surface modes of ultrasound contrast agent microbubbles, IEEE Ultrason. Symp. (2006) 112–115.
[33] F. Perren, J. Loulidi, D. Poglia, T. Landis, R. Sztajzel, Microbubble potentiated transcranial duplex ultrasound enhances IV thrombolysis in acute stroke, J. Thromb. Thrombolys. 25 (2008) 219–223.
[34] M. S Plesset, A. Prosperetti, Bubble dynamics and cavitation, Ann. Rev. Fluid Mech. 9 (1977) 145–185.
[35] S. Popinet, An accurate adaptive solver for surface-tension-driven interfacial flows, J. Comput. Phys. 228 (2009) 9293–9302.
[36] S.B. Raymond, L.H. Treat, J.D. Dewey, N.J. McDannold, K. Hynynen, B.J. Bacskai, Ultrasound enhanced delivery of molecular imaging and therapeutic agents in Alzheimer's Disease mouse models, PLoS ONE 3 (2008) e2175.
[37] M. Rudman, A volume-tracking method for incompressible multi-fluid flows with large density variations, Int. J. Numer. Methods Fluids 28 (1998) 357–378.
[38] Y. Saad, Iterative Methods for Sparse Linear Systems, PWS, 1996.
[39] Y. Saad, M.H. Schultz, GMRES: a generalized minimal residual method for solving nonsymmetric linear systems, SIAM J. Sci. Stat. Comput. 7 (1986) 856–869.
[40] M. Sussman, A second order coupled level set and volume-of-fluid method for computing growth and collapse of vapor bubbles, J. Comput. Phys. 187 (2003) 110–136.
[41] M. Sussman, M. Ohta, A stable and efficient method for treating surface tension in incompressible two-phase flow, SIAM J. Sci. Comput. 31 (2009) 2447–2471.
[42] M. Sussman, M. Ohta, High-order techniques for calculating surface tension forces, Int. Ser. Numer. Math. 154 (2009) 425–434.
[43] M. Sussman, E.G. Puckett, A coupled Level Set and Volume-of-Fluid method for computing 3D and axisymmetric incompressible two-phase flows, J. Comput. Phys. 162 (2000) 301–337.
[44] P. Tho, R. Manasseh, A. Ooi, Cavitation microstreaming in single and multiple bubble systems, J. Fluid Mech. 576 (2007) 191–223.
[45] P. Wesseling, An Introduction to Multigrid Methods, Wiley, 1992.
[46] M.W. Williams, E.G. Puckett, D.B. Kothe Convergence and accuracy of continuum surface tension models, in: W. Shyy, R. Narayanan (Eds.), Fluid Dynamics at Interfaces, 1999, pp. 294–305.