OpenFOAM® Journal

Volume 2 [16th OpenFOAM® Workshop - 2021], Pages 31–47
ISSN: 2753-8168

ARTIFICIAL COMPRESSIBILITY WITH RIEMANN SOLVERS: CONVERGENCE OF LIMITERS ON UNSTRUCTURED MESHES

SHANNON LEAKEY, VASSILIS GLENIS*, AND CASPAR J. M. HEWETT

School of Engineering, Newcastle University, United Kingdom
Email address: s.c.leakey2@newcastle.ac.uk, vassilis.glenis@newcastle.ac.uk, caspar.hewett@newcastle.ac.uk

DOI: 10.51560/ofj.v2.49
Version(s): OpenFOAM® v1906
Repo: -

Abstract. Free-surface flows and other variable density incompressible flows have numerous important applications in engineering. One way such flows can be modelled is to extend established numerical methods for compressible flows to incompressible flows using the method of artificial compressibility. Artificial compressibility introduces a pseudo-time derivative for pressure and, in each real-time step, the solution advances in pseudo-time until convergence to an incompressible limit—a fundamentally different approach than SIMPLE, PISO, and PIMPLE, the standard methods used in OpenFOAM® . Although the artificial compressibility method is widespread in the literature, its application to free-surface flows is not. In this paper, we apply the method to variable density flows on 3D unstructured meshes, implementing a Godunov-type scheme with MUSCL reconstruction and Riemann solvers, where the free surface gets captured automatically by the contact wave in the Riemann solver. The critical problem in this implementation lies in the slope limiters used in the MUSCL reconstruction step. It is well-known that slope limiters can inhibit convergence to steady state on unstructured meshes; the problem is exacerbated here as convergence in pseudo-time is required not just once, but at every real-time step. We compare the limited gradient schemes included in OpenFOAM® with an improved limiter from the literature, testing the solver against dam-break and hydrostatic pressure benchmarks. This work opens OpenFOAM® up to the method of artificial compressibility, breaking the mould of PIMPLE and harnessing high-resolution shock-capturing schemes that can scale better in parallel.

1. Introduction

Variable density incompressible flows are of much practical interest in engineering. One important class of variable density incompressible flows is free-surface flows of water and air, for example, dam breaks [1], hydraulic jumps [2], hydraulic structures [3], and nature-based flood defences, both coastal [4] and fluvial [5]. These flows in these papers [1, 2, 3, 4, 5] were all modelled using the volume-of-fluid (VOF) solver `interFoam`. As with most OpenFOAM® solvers, `interFoam` uses the PIMPLE algorithm to update the pressure field. In this paper, we harness the flexibility of OpenFOAM® to implement an alternative free-surface solver based on the method of artificial compressibility [6], which has a key computational advantage compared to PIMPLE.

Both artificial compressibility and PIMPLE address the problem of the incompressible Navier-Stokes equations having no equation for pressure. In artificial compressibility, the idea is to add a pseudo-time derivative for pressure to the incompressibility constraint, and the solution advances in this pseudo-time until convergence to an incompressible limit [7, 8, 9]. This makes the governing equations hyperbolic, and thus solvable with the wealth of methods developed for compressible flows, for example, Godunov-type schemes involving MUSCL (monotonic upstream-centered scheme for conservation laws) reconstruction and Riemann solvers. While the method was originally developed for steady-state cases, it can be generalised to transient cases by employing dual time stepping, where the solution converges to the incompressible limit each real-time step.

This is a fundamentally different approach to the PIMPLE algorithm. PIMPLE, a combination of SIMPLE (semi-implicit method for pressure-linked equations) [10] and PISO (pressure-implicit with splitting of operators) [11], is a predictor-corrector method that relies on matrix inversion. While such methods require fewer total iterations than artificial compressibility and therefore would be more efficient in serial

* Corresponding author
Received: 1 July 2021, Accepted: 8 February 2022, Published: 4 March 2022

[12], artificial compressibility is easier and more efficient to parallelise [12, 13, 14] and can lead to almost linear speed-up [15]. This is because solving local wave propagation problems requires less communication between processors than matrix inversion [12]. Therefore, it is worthwhile to depart from the convention of using PIMPLE in OpenFOAM® and investigate implementing artificial compressibility instead because of its scaling potential on the massively parallel architectures of today.

Although the artificial compressibility method is widespread, its application to free-surface flows is not [16]. Kelecy and Pletcher [17] pioneered an approach, modelling the flows as incompressible with variable density and then the air-water interface gets captured automatically by the contact wave in the Riemann solver. As such, no special treatment like in VOF [18] is required, making the surface capturing simpler computationally. Until the present paper, numerical methods to solve the variable-density artificial-compressibility equations have not been implemented on 3D unstructured meshes. They have been implemented on 2D structured meshes [19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36], 3D structured meshes [17, 37, 38, 39], and 2D unstructured meshes [40]. While most of these papers use the Cartesian cut-cell method to model simple obstacles, an implementation on 3D unstructured meshes is needed for more complex geometries. This is lacking in the current literature, but other multiphase artificial compressibility methods have been implemented on unstructured meshes, both 2D [41, 42, 43, 44] and 3D [45, 46, 47, 15, 48], as well as structured meshes [13, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60]. Consequently, there is room for further developing multiphase artificial compressibility methods on unstructured meshes, especially for investigating the limiter convergence problem only briefly hinted at by [48].

Capitalising on the powerful tools provided by OpenFOAM® , we generalise the specific variable-density artificial-compressibility scheme in [22] to 3D unstructured meshes, testing the solver against dam-break and hydrostatic pressure benchmarks. Even though the new solver does not use much of the numerics in OpenFOAM® , it is useful to develop the solver in OpenFOAM® because the low-level structure does not have to be built from scratch. A downside is that the form of parallelisation in OpenFOAM® , domain decomposition, is not optimal for artificial compressibility as the waves propagate locally [12]. However, we focus on the numerics in this paper, leaving the implementation of efficient parallelisation to future work.

2. Implementation

The variable-density artificial-compressibility equations are

$$\frac{\partial \rho}{\partial \tau} + \frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0 \tag{1}$$

$$\frac{\partial}{\partial \tau}(\rho \mathbf{u}) + \frac{\partial}{\partial t}(\rho \mathbf{u}) + \nabla \cdot (\rho \mathbf{u} \otimes \mathbf{u}) = -\nabla p + \nabla \cdot \left(\mu \left(\nabla \mathbf{u} + \nabla \mathbf{u}^T\right)\right) + \rho \mathbf{g} \tag{2}$$

$$\frac{1}{\beta}\frac{\partial p}{\partial \tau} + \nabla \cdot \mathbf{u} = 0 \tag{3}$$

where ρ is density, $\mathbf{u}$ velocity, p pressure, μ dynamic viscosity, $\mathbf{g}$ gravity, β the artificial compressibility coefficient, t real-time, and τ pseudo-time. Each real-time step requires convergence in pseudo-time, and so two nested time loops are needed for the dual time stepping, as well as a Runge-Kutta loop. If the solution converges in pseudo-time, then $\partial p/\partial \tau \approx 0$ and the artificial compressibility coefficient β should not impact the solution. However, when the maximum density is around $\rho = 10^3$ as for water, setting $\beta < 10^3$ can result in instability and setting $\beta > 10^4$ can inhibit convergence, meaning $\partial p/\partial \tau \approx 0$ is not achieved [17]. Therefore some trial and error is required to pick a suitable value of β. In practice, we have found that setting $\beta = 1100$ is just large enough to avoid instability, and so will have minimal impact on convergence.

We solve the governing equations (1)–(3) by implementing in OpenFOAM® the numerical scheme described in detail and implemented on 2D Cartesian meshes by [22]. In that study, a Godunov-type scheme was developed that was explicit in pseudo-time and point-implicit in real-time, especially easy to parallelise, and had a low memory footprint. Specifically, the explicit Runge-Kutta pseudo-time update

equation is given by

$$\mathbf{Q}_i^{n+1,m+1,0} = \mathbf{Q}_i^{n+1,m} \tag{4}$$

$$\mathbf{Q}_i^{n+1,m+1,s} = \mathbf{Q}_i^{n+1,m+1,0} + \frac{\alpha_s \Delta\tau}{\alpha_{PI}} \left(-\frac{1}{|\Omega|} \sum_{f \in \partial\Omega} \mathbf{F}_f^{n+1,m+1,s-1} \cdot \mathbf{S}_f - \mathbf{I}_C \frac{\partial}{\partial t}(\mathbf{Q}_i^{n+1,m+1,0}) + \mathbf{B}_i^{n+1,m+1,0} \right) \quad \text{for } s = 1, ..., s_{max} \tag{5}$$

$$\mathbf{Q}_i^{n+1,m+1} = \mathbf{Q}_i^{n+1,m+1,s_{max}} \tag{6}$$

where $\mathbf{Q}$ is the state vector, Ω the computational cell, $\mathbf{F}$ the flux vector, $\mathbf{I}_C$ the cancellation matrix, $\mathbf{B}$ the body force term, α_S the Runge-Kutta coefficient, s_{max} the total number of Runge-Kutta stages, α_{PI} the point-implicit scaling coefficient, n the current real-time iteration, m the current pseudo-time iteration, and s the current Runge-Kutta stage. The real-time derivative is treated as a source term, calculated point-implicitly with a backward-differencing scheme, and the fluxes are approximated using a Riemann solver. For full details, see [22]. Note that we consider the inviscid flux only (i.e. we set $\mu = 0$), but the viscous flux could be added easily using standard finite difference methods [17, 43].

The present paper goes further than [22], implementing the numerical scheme (4)–(6) in OpenFOAM® so that it can be used on 3D unstructured meshes and in parallel. The new implementation uses the Roe Riemann solver and the new pressure gradient calculation developed in [22]. It has a structure inspired by foam-extend's `dbnsFoam` [61]. However, it was written from scratch to take into account the different governing equations and the fact that several cell-limited gradient schemes were added to OpenFOAM® since `dbnsFoam` was released. Therefore, it has the potential to be forward-compatible with any limiters added to OpenFOAM® in the future.

Switching to an unstructured mesh does require slightly changing the scheme from [22]. The difference is in the MUSCL reconstruction step, which is where the primitive variables are extrapolated to cell faces to provide the left and right states for the Riemann solver. This extrapolation relies upon a limited gradient to avoid the introduction of new extrema, Gibbs-type oscillations. Unstructured meshes require a different approach than structured meshes to do MUSCL reconstruction, and there are two components to this difference: calculating the gradient and limiting the gradient.

First, the gradient calculation for Cartesian meshes is very simple, but unstructured meshes require a method such as Green-Gauss or least-squares. These well-known standard methods are already implemented in OpenFOAM® , so using them is simply a case of writing `fvc::grad()` in the code and specifying `Gauss` or `leastSquares` in `fvSchemes`. The gradients can be limited by specifying a limiter alongside `Gauss` or `leastSquares` in `fvSchemes`. Note that this use of `fvc::grad()` is the only time we use OpenFOAM® numerics here, but it is not the gradient calculation itself which is problematic.

The problem lies in the limiters, a problem that did not exist in the implementation for 2D Cartesian meshes in [22]. The difference is that, while the limiter calculation for structured meshes is straightforward, involving constructing left- and right-sided gradients at each cell and applying a standard limiter to these, it is not obvious what the left- and right-sided gradients would be on an unstructured mesh [62]. The standard way to overcome this problem is the framework of Barth and Jespersen [63], which is the method implemented in the cell-limited gradient schemes in OpenFOAM® .

2.1. **Barth and Jespersen.** Let W be a scalar variable or one of the components of a vector variable. In the context of a Godunov-type scheme, the idea is to construct a scalar limiter function $\Phi \in [0, 1]$ to be used as

$$W_{f,i} = W_i + \Phi_i \nabla W_i \cdot (\mathbf{x}_f - \mathbf{x}_i) \tag{7}$$

where ∇W_i is gradient of the variable before limiting and $W_{f,i}$ is the Riemann state at face f on the side belonging to cell i. Then the Riemann states either side of the face can be put into the Riemann solver to get the numerical flux. Barth and Jespersen [63] introduced a method that ensures the interpolated variable $W_{f,i}$ does not exceed that of the neighbouring cells in magnitude. Note that this means all the neighbouring cells of cell i, not just the cell on the other side of face f.

Practically, the calculation loops over all the cell interfaces twice [64]. There is a loop to calculate the maximum and minimum values in the neighbouring cells,

$$W_{i,\max} = \max\left(W_i, \max_{j \in \text{nei}} W_j\right) \tag{8}$$

$$W_{i,\min} = \min\left(W_i, \min_{j \in \text{nei}} W_j\right), \tag{9}$$

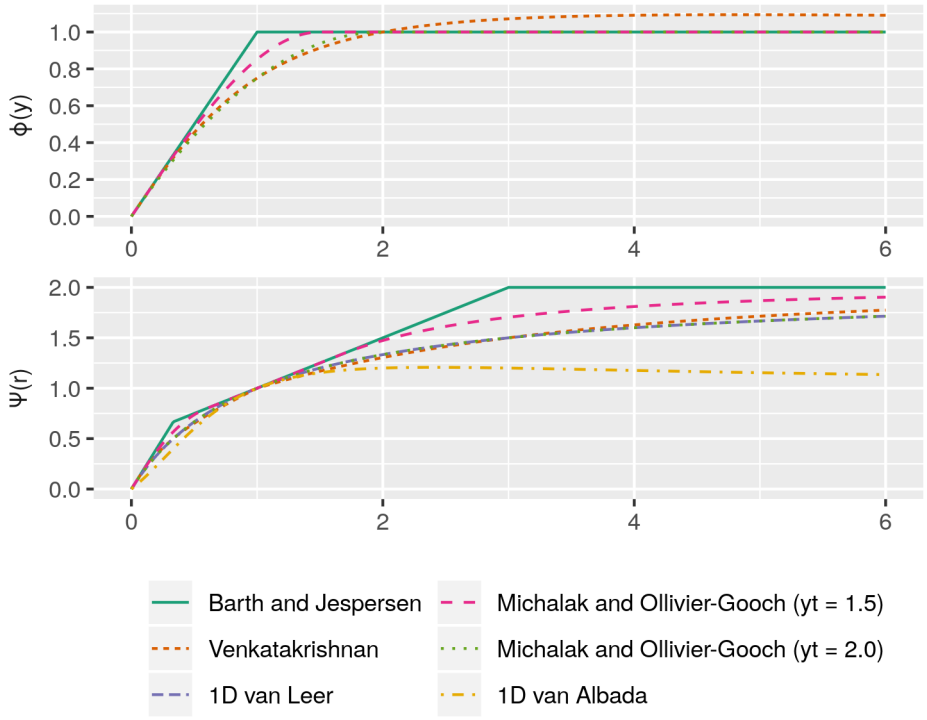

FIGURE 1. Comparison of ϕ (used in unstructured limiter Φ) and ψ (used in Spekreijse's 1D formulation) between different limiters.

and then a loop to determine how much limiting is required,

$$\Phi_i = \min_f \begin{cases} \phi\left(\frac{\Delta_{i,\max}}{\Delta_{i,f}}\right) & \text{if } \Delta_{i,f} > 0 \\ \phi\left(\frac{\Delta_{i,\min}}{\Delta_{i,f}}\right) & \text{if } \Delta_{i,f} < 0 \\ 1 & \text{if } \Delta_{i,f} = 0 \end{cases} \tag{10}$$

where

$$\phi(y) = \min(1, y) \tag{11}$$

and

$$\Delta_{i,\max} = W_{i,\max} - W_i \tag{12}$$
$$\Delta_{i,\min} = W_{i,\min} - W_i \tag{13}$$
$$\Delta_{i,f} = \nabla W_i \cdot \mathbf{d}_{i,f} \tag{14}$$
$$\mathbf{d}_{i,f} = \mathbf{x}_f - \mathbf{x}_i. \tag{15}$$

Thus, the quantity $\Delta_{i,f}$ is a measure of how much W changes between the centre of cell i and the centre of the cell interface f. The function ϕ clips $\Delta_{i,f}$ so that it does not exceed $\Delta_{i,\min}$ or $\Delta_{i,\max}$ in magnitude.

In the uniform 1D case, the Barth-Jespersen limiter can be recast into the Spekreijse [65] form

$$W^L_{i+1/2} = W_i + \frac{1}{2}\psi(r_i)(W_i - W_{i-1}) \tag{16}$$

where the limiter is

$$\psi(r) = \frac{1}{2}(r+1)\min\left(\phi\left(\frac{4r}{r+1}\right), \phi\left(\frac{4}{r+1}\right)\right) \tag{17}$$

and it is applied to the ratio

$$r = \frac{W_{i+1} - W_i}{W_i - W_{i-1}}, \tag{18}$$

as shown in [66]. Both ϕ from (11) and ψ from (17) are plotted in Figure 1.

The practical problem with this method is that the non-differentiability of (11) can inhibit convergence in steady-state cases, instead leading to bounded odd-even modes. The simulation is not unstable; it does not crash or blow up, but neither does it make any progress towards convergence. It would continue

indefinitely if it was not stopped. This is a well-known problem [48, 64, 66, 67, 68, 69, 70, 71, 72, 73]. Fortunately, there are ways to overcome it, including replacing the non-differentiable function (11) with a differentiable alternative and switching off the limiter in areas of relatively uniform flow. In the present paper, we investigate these techniques in the context of artificial compressibility, where pseudo-time convergence is required not just once, but at every real-time step, and thus the limiters must converge more reliably than for a simple steady-state simulation.

2.2. **Venkatakrishnan.** Venkatakrishnan [66] addressed the convergence problem by replacing the non-differentiable function (11) with the differentiable function

$$\phi(y) = \frac{y^2 + 2y}{y^2 + y + 2}. \tag{19}$$

As shown in Figure 1, this function is much smoother than (11), and the fact that $\phi > 1$ for $y > 2$ does not affect ψ, which is bounded by the Barth-Jespersen limiter for all r.

A slight modification, not plotted in Figure 1, ensures that the limiter is also not activated in smooth regions:

$$\Phi_i = \min_f \begin{cases} \frac{1}{\Delta_{i,f}} \left(\frac{(\Delta^2_{i,\max} + \epsilon_i^2)\Delta_{i,f} + 2\Delta^2_{i,f}\Delta_{i,\max}}{\Delta^2_{i,\max} + 2\Delta^2_{i,f} + \Delta_{i,\max}\Delta_{i,f} + \epsilon_i^2} \right) & \text{if } \Delta_{i,f} > 0 \\ \frac{1}{\Delta_{i,f}} \left(\frac{(\Delta^2_{i,\min} + \epsilon_i^2)\Delta_{i,f} + 2\Delta^2_{i,f}\Delta_{i,\min}}{\Delta^2_{i,\min} + 2\Delta^2_{i,f} + \Delta_{i,\min}\Delta_{i,f} + \epsilon_i^2} \right) & \text{if } \Delta_{i,f} < 0 \\ 1 & \text{if } \Delta_{i,f} = 0 \end{cases} \tag{20}$$

where

$$\epsilon_i^2 = K^3 V_i \tag{21}$$

and V_i is the cell volume [64]. The parameter K is a threshold that marks the largest size of oscillations untouched by the limiter.

If the gradient is small, then $\epsilon_i^2 \gg \Delta^2_{i,\min}, \Delta^2_{i,\max}, \Delta_{i,\min}\Delta_{i,\max}$ and so $\Phi_i \to 1$, which means there is no limiting. This stops residuals stalling due to numerical noise. By a similar argument, increasing K increases Φ_i and so decreases the amount of limiting, and while this is more conducive to convergence to a steady-state, it also increases the potential for Gibbs-type oscillations and thus instability. The more convergent options are more unstable, that is, there is a trade-off between convergence and stability.

2.3. **Michalak and Ollivier-Gooch.** While Barth-Jespersen and Venkatakrishnan are the two standard limiters [64], there are more options in the literature. For example, Michalak and Ollivier-Gooch [67] replaced the function (11) with

$$\phi(y) = \begin{cases} P(y) & \text{if } y < y_t \\ 1 & \text{if } y \geq y_t \end{cases} \tag{22}$$

where $P(y)$ is the cubic polynomial with

$$P\big|_0 = 0 \tag{23}$$

$$P\big|_{y_t} = 1 \tag{24}$$

$$\frac{dP}{dy}\Big|_0 = 1 \tag{25}$$

$$\frac{dP}{dy}\Big|_{y_t} = 0 \tag{26}$$

and $1 \leq y_t \leq 2$ is a threshold. The polynomial itself is not explicitly stated in [67], but a simple derivation gives

$$P(y) = ay^3 + by^2 + y \tag{27}$$

$$a = \frac{1}{y_t^2} - \frac{2}{y_t^3} \tag{28}$$

$$b = -\frac{3}{2} a y_t - \frac{1}{2y_t}. \tag{29}$$

As shown in Figure 1, the function ϕ is differentiable everywhere and does not exceed 1.

Like Venkatakrishnan, Michalak and Ollivier-Gooch also proposed switching off the limiter in uniform regions of flow. However, they used a different method for this, smoothly transitioning to switching off the limiter when

$$(\Delta_{i,\max} - \Delta_{i,\min})^2 < K^3 V_i. \tag{30}$$

This is done by defining

$$\tilde{\phi}_i = \sigma_i + (1 - \sigma_i)\phi \tag{31}$$

with ϕ from (22) and $\tilde{\phi}_i$ now plugged into (10). The indicator-type function is given by

$$\sigma_i = \begin{cases} 1 & \text{if } (\Delta_{i,\max} - \Delta_{i,\min})^2 \leq K^3 V_i \\ s\left(\frac{(\Delta_{i,\max} - \Delta_{i,\min})^2 - K^3 V_i}{K^3 V_i}\right) & \text{if } K^3 V_i < (\Delta_{i,\max} - \Delta_{i,\min})^2 < 2K^3 V_i \\ 0 & \text{if } (\Delta_{i,\max} - \Delta_{i,\min})^2 \geq 2K^3 V_i \end{cases} \tag{32}$$

with the smooth transition function s given by

$$s(y) = 2y^3 - 3y^2 + 1. \tag{33}$$

2.4. **Cell-limited schemes in OpenFOAM®.** The above three limiters are already implemented in OpenFOAM® , and can be easily accessed by specifying `cellLimited`, `cellLimited<Venkatakrishnan>`, or `cellLimited<cubic>` respectively alongside the gradient calculation in `fvSchemes`. However, there are some major problems with the limiters as implemented in the OpenFOAM® source code.

First, `cellLimited<Venkatakrishnan>` is not the same as the original limiter put forward by Venkatakrishnan, and so it cannot be expected to have the same convergence properties shown in the original study [66]. One difference is that it uses the unmodified version (19) instead of (20), and so may still be active in uniform regions of flow. Another difference is that, regardless of the limiter, OpenFOAM® always clips Φ so that it never exceeds 1. As indeed noted in the source code documentation, this clipping makes this particular limiter non-differentiable, and so it "no longer conforms to the basic principles of this kind of limiter function" (`VenkatakrishnanGradientLimiter.H`, lines 53–54). The whole reason Venkatakrishnan developed the limiter was so it could be differentiable, therefore it seems that `cellLimited<Venkatakrishnan>` is of little practical use.

Second, `cellLimited<cubic>` is also not the same as the original limiter put forward by Michalak and Ollivier-Gooch, and so cannot be expected to have the same convergence properties shown in the original study [67] either. One difference is that it does not use (31), and therefore does not stop activation in uniform regions of flow. Another difference is that the limiter uses an incorrect cubic polynomial, one with a very large slope discontinuity at $\phi(y_t)$, not the one stated in (27)–(29).*

In this study, we do not attempt to fix `cellLimited<Venkatakrishnan>`; the automatic clipping of Φ makes this too problematic. However, we do implement an improved version of `cellLimited<cubic>` that corresponds to the limiter in the original paper [67], both by using the correct cubic polynomial and by stopping activation in uniform regions of flow. This improved version is called `cellLimited<Michalak>`.†

Note that there are three other limited gradient schemes in OpenFOAM® : `cellMDLimited`, `faceLimited`, and `faceMDLimited`. In the `cellLimited` limiters outlined above, one scalar limiter is applied to the x, y, and z components of the gradient equally, irrespective of which neighbouring cells have the minimum and maximum values. This can lead to excessive limiting. The `cellMDLimited` limiter takes an alternative approach. Consider a cell with faces $f = 1, 2, ..., F$. Set $(\nabla W)_{i,0} = (\nabla W)_i$ to be the gradient before limiting. Loop through the faces f, calculating the extrapolate

$$\Delta_{i,f} = (\nabla W)_{i,f-1} \cdot \mathbf{d}_{i,f} \tag{34}$$

and then clipping the gradient in the direction from the cell centre to that face centre:

$$(\nabla W)_{i,f} = \begin{cases} (\nabla W)_{i,f-1} + \mathbf{d}_{i,f} \frac{\Delta_{\mathrm{i,max}} - \Delta_{i,f}}{|\mathbf{d}_{i,f}|^2} & \text{if } \Delta_{i,f} > \Delta_{\mathrm{i,max}} \\ (\nabla W)_{i,f-1} + \mathbf{d}_{i,f} \frac{\Delta_{\mathrm{i,min}} - \Delta_{i,f}}{|\mathbf{d}_{i,f}|^2} & \text{if } \Delta_{i,f} < \Delta_{\mathrm{i,min}} \\ (\nabla W)_{i,f-1} & \text{otherwise} \end{cases} \tag{35}$$

for $f = 1, 2, ..., F$. The final iteration gives us the `cellMDLimited` gradient. Therefore, the limiter "is applied to the gradient in each face direction separately" (`cellMDLimitedGrad.H`, lines 36–38) not, as is sometimes suggested, to each coordinate direction separately. Meanwhile, the `faceLimited` and `faceMDLimited` limiters are like the `cellLimited` and `cellMDLimited` limiters but clip the gradient between the face-neighbour values rather than the cell-neighbour values. None of these allow differentiable

*As of June 2021, the cubic polynomial has been fixed in the official development branches due to the work carried out in the present paper. See the bug reports at `https://develop.openfoam.com/Development/openfoam/-/issues/2113` [Accessed: 10 February 2022] and `https://bugs.openfoam.org/view.php?id=3684` [Accessed: 10 February 2022] for details. However, these bug fixes do not stop activation in uniform regions of flow.

†Thus `cellLimited<Michalak>` with $K = 0$ is equivalent to `cellLimited<cubic>` in the official development branches from June 2021.

functions as a run-time selectable option like `cellLimited` does. However, simulations using them are included for comparison purposes in the following section, alongside `cellLimited`, `cellLimited<Venkatakrishnan>`, `cellLimited<cubic>`, and the newly implemented `cellLimited<Michalak>`.

3. Application

The new solver was put through three benchmark tests: a simple dam break to illustrate the importance of limiter choice, a more complicated dam break to compare the new solver against `interFoam`, and a hydrostatic case to demonstrate that the solver can be run on arbitrary 3D unstructured meshes.

3.1. **Convergence of limiters.** First, we replicated from [22] the first real-time step of a dam break on a uniform 2D Cartesian mesh. The mesh was a metre length in each direction, divided into 20×20 cells. Despite its simplicity, the mesh had to be stored as a 3D unstructured mesh in OpenFOAM® . However, due to its simplicity, the mesh was useful for isolating the effect of different limiters on convergence. Initial conditions were set to be $\rho = 1000$ in the water and $\rho = 1$ in the air, with $\mathbf{u} = 0$ and $p = 0$ everywhere. The solver was run for one real-time step of size $\Delta t = 0.01$ seconds with the following settings: three Runge-Kutta stages, an artificial compressibility coefficient of $\beta = 1100$, and a Courant number for pseudo-time of 0.48 to satisfy the explicit stability constraint. The Green-Gauss gradient calculation was chosen for ρ and $\mathbf{u}$, and the corresponding limiter changed in each simulation to show its effect on convergence.

Recall that the solution at each real-time step is reached when the solution converges in pseudo-time. This is measured with residuals, here calculated as the maximum absolute difference between a conserved variable in the current and previous pseudo-time steps. Figure 2 shows that, of all the limiters available in OpenFOAM® as standard, only `faceLimited` converged in this case, and not very smoothly. It might seem like the convergence problem was because all the OpenFOAM® limiters are non-differentiable, as discussed in Section 2.4. However, Figure 2 shows that, despite being differentiable, `cellLimited<Michalak>` ($K = 0$) did not converge either. It was only when the limiter was switched off in areas of uniform flow that convergence was achieved ($K = 1$). Clearly, when $K = 1$, the newly implemented limiter `cellLimited<Michalak>` has improved convergence properties compared to the options already present in OpenFOAM® .

It is important to note that, although the residuals stalled for nearly all of the other limiters, there was no instability and therefore the simulations did not blow up. Only a few cells failed to converge, and they switched between very similar states, which meant the final results did not differ substantially between the limiters. Consider Figure 3. The residuals only stalled in the top-left corner above the water column. Once K was changed to 1 for `cellLimited<Michalak>`, the limiter switched off in this area of uniform flow.

3.2. **Comparison with `interFoam`.** The new solver was tested with the `damBreak` tutorial to compare its performance with `interFoam`. Initial conditions were set to be $\rho = 1000$ in the water and $\rho = 1$ in the air, with $\mathbf{u} = 0$ and $p = 0$ everywhere. The simulations were run until $t = 1.0$ or until the residuals stalled, whichever was sooner, with the following settings: one Runge-Kutta stage, an artificial compressibility coefficient of $\beta = 1100$, and a Courant number for pseudo-time of 0.48 to satisfy the explicit stability constraint. At each real-time step, the absolute tolerances for pseudo-time convergence were 0.0001 for ρ, 0.01 for $\rho\mathbf{u}$, and 0.01 for p. The Green-Gauss gradient calculation was chosen for ρ and $\mathbf{u}$, and the corresponding limiter changed along with the Courant number for real-time. Recall that real-time is treated point-implicitly, and so there is no explicit stability constraint for the real-time Courant number.

Figure 4 shows how far each simulation could go. The residuals were prone to stalling when the tail of the column of water hit either the obstacle or the far wall and, as expected, `cellLimited<Michalak>` converged better than `cellLimited`. However, sometimes `cellLimited<Michalak>` stalled, and not always with the same parameters. In this particular benchmark, when the real-time Courant number was 1, the choice $y_t = 1.5$ converged but $y_t = 2.0$ stalled, and it was the other way around if the Courant number was 2 or 5. Both choices of y_t stalled if the Courant number was 0.5, perhaps due to the increased number of real-time steps meaning there were more chances to stall. Moreover, the effect of switching off the limiter in uniform regions of flow (that is, setting $K > 0$) was not noticeable. This suggests that the standard slope limiters for unstructured meshes are not sufficiently robust for this solver, whether the limiters already available in OpenFOAM® or the limiter implemented in this paper. Again, this is nothing to do with stability. At no point did the simulations blow up or crash; they simply stopped making progress towards convergence.

Despite this lack of convergence, we could use parameters that worked for this particular benchmark to compare the new solver, as it stands, with `interFoam`. Figures 5–6 show the results for

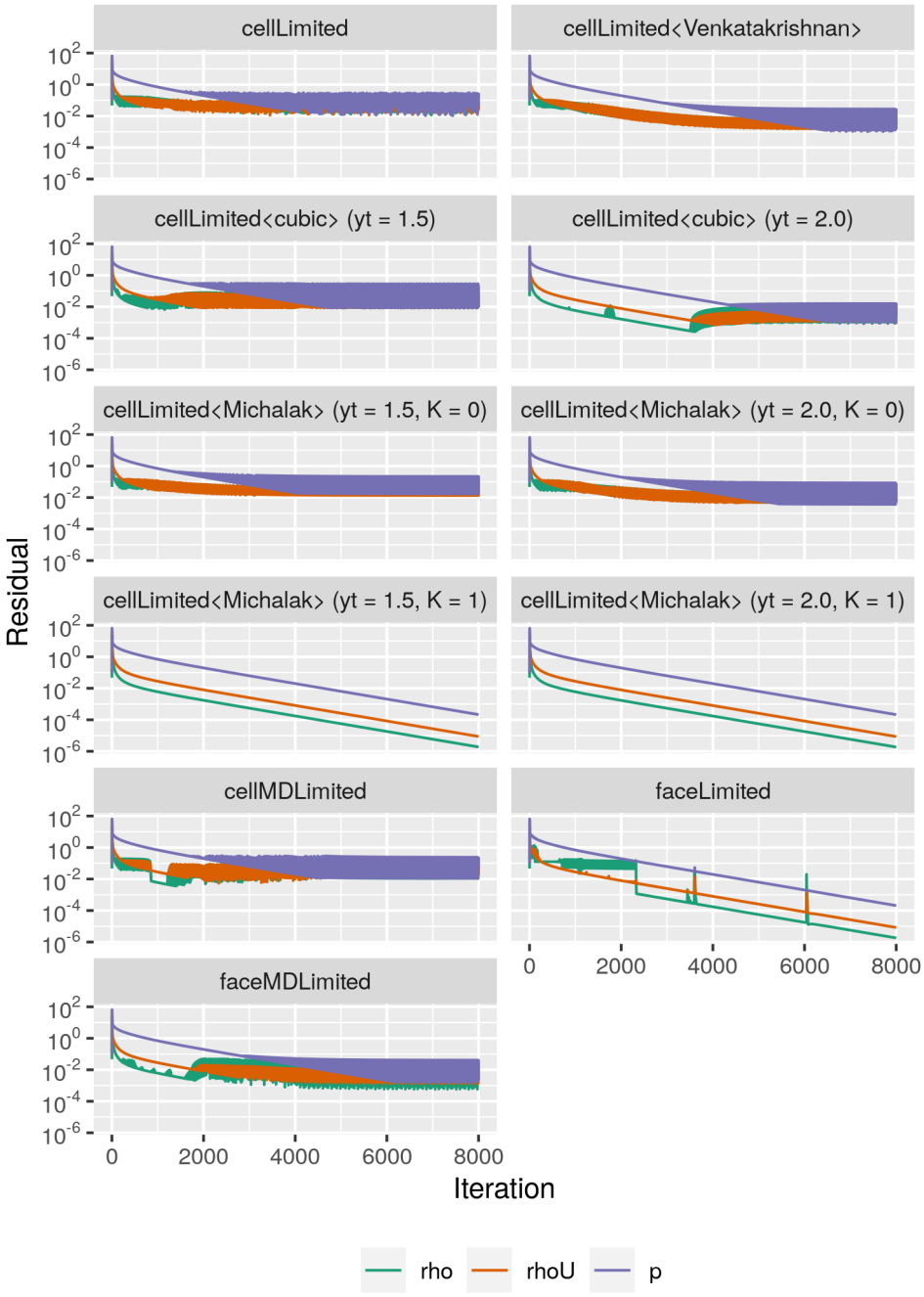

FIGURE 2. Convergence history for different gradient limiters.

`cellLimited<Michalak>` with $y_t = 2.0$, $K = 1$, and real-time Courant numbers of 2 and 5. Figure 5 includes results for the `interFoam` simulations, where the settings were as in the standard laminar `damBreak` tutorial, but with zero viscosity and surface tension, and the `atmosphere` patch a wall. Now, `interFoam` keeps the free surface sharp using a compression term[‡] with coefficient `cAlpha`. Although the compression term has a physical basis [74, p.117], in practice `cAlpha` is usually arbitrarily set to 1. Setting `cAlpha` to 0 is equivalent to removing the interface compression term—not a practice acceptable for practical simulations but useful nonetheless for this comparison. Figure 5 shows that the new solver with a real-time Courant number of 2 had very similar behaviour to `interFoam` with a `cAlpha` of 0. Surprisingly, when the real-time Courant number was increased to 5, the new solver still managed to capture the highly transient behaviour of the dam break, with only slightly less detail. This is encouraging

[‡]Often called artificial compression, completely unrelated to artificial compressibility.

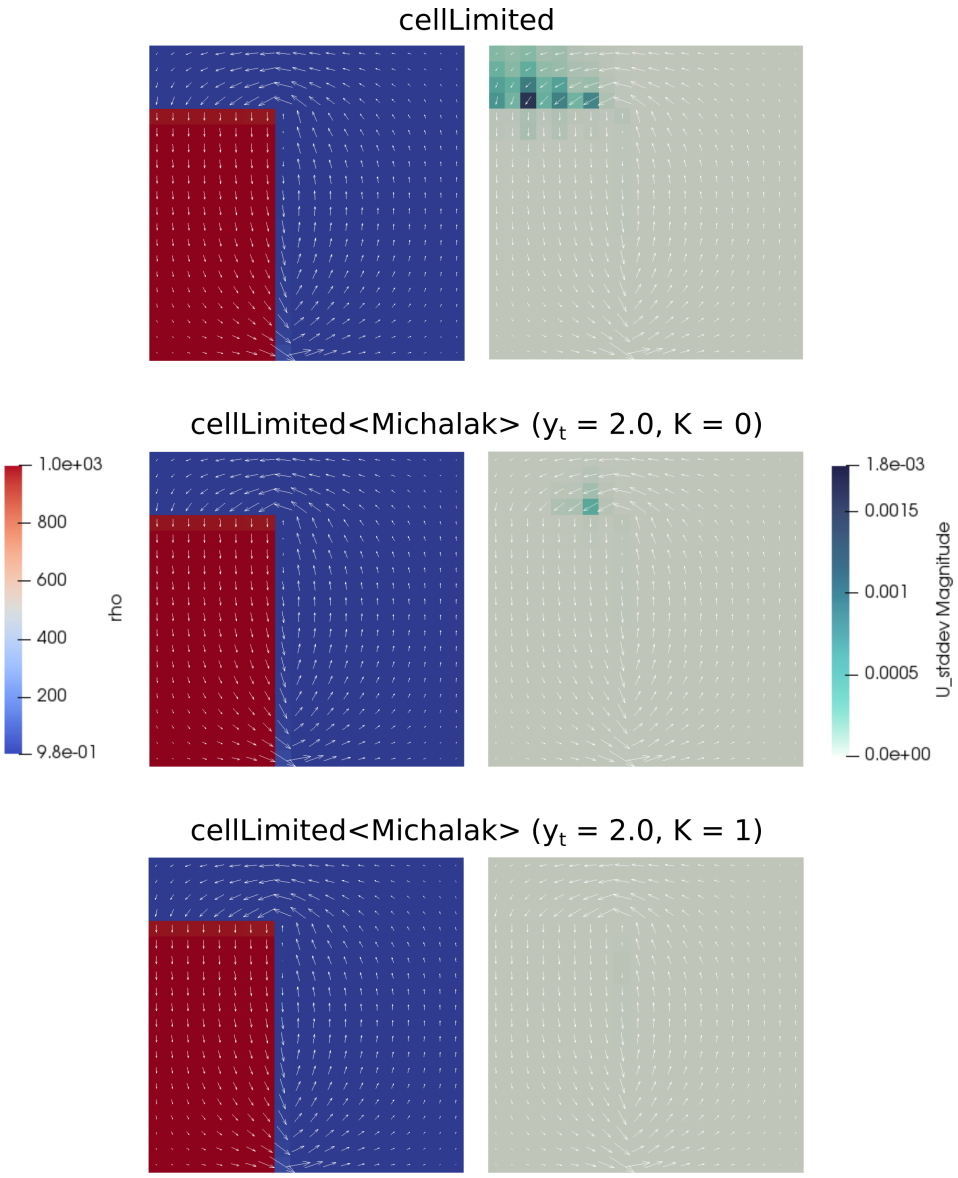

FIGURE 3. Density field after 8000 pseudo-time iterations, and standard deviation of velocity magnitude over pseudo-time iterations 7000–8000 (results saved every 100 iterations).

because fewer total iterations are required for this larger real-time Courant number, as shown by Figure 6.

While the solver has potential, Figure 5 shows that the use of a high-resolution Godunov-type scheme is not sufficient by itself to keep the interface sharp. Activating the interface compression term does indeed keep the interface sharp when using `interFoam`, but this comes at the cost of a deformed free surface (see the $t = 0.6$ frame). It was hoped that a Godunov-type scheme might circumvent this problem, but clearly that is not the case, and something like the interface compression term in `interFoam` is required to keep the interface sharp. This has been done before in the context of Godunov-type schemes for artificial compressibility coupled with VOF [57, 55], and would be a good next step once the convergence problem has been solved.

3.3. **Mesh from `snappyHexMesh`.** The meshes in the previous benchmarks appeared structured, even if they were not stored that way in OpenFOAM® . Therefore, to demonstrate that the solver can be run on arbitrary meshes, it was also tested on a much less uniform mesh generated by `snappyHexMesh`. A simple hydrostatic case was chosen to avoid the convergence problem and because the solution is known, allowing us to test the impact of the pressure gradient calculation on the simulation.

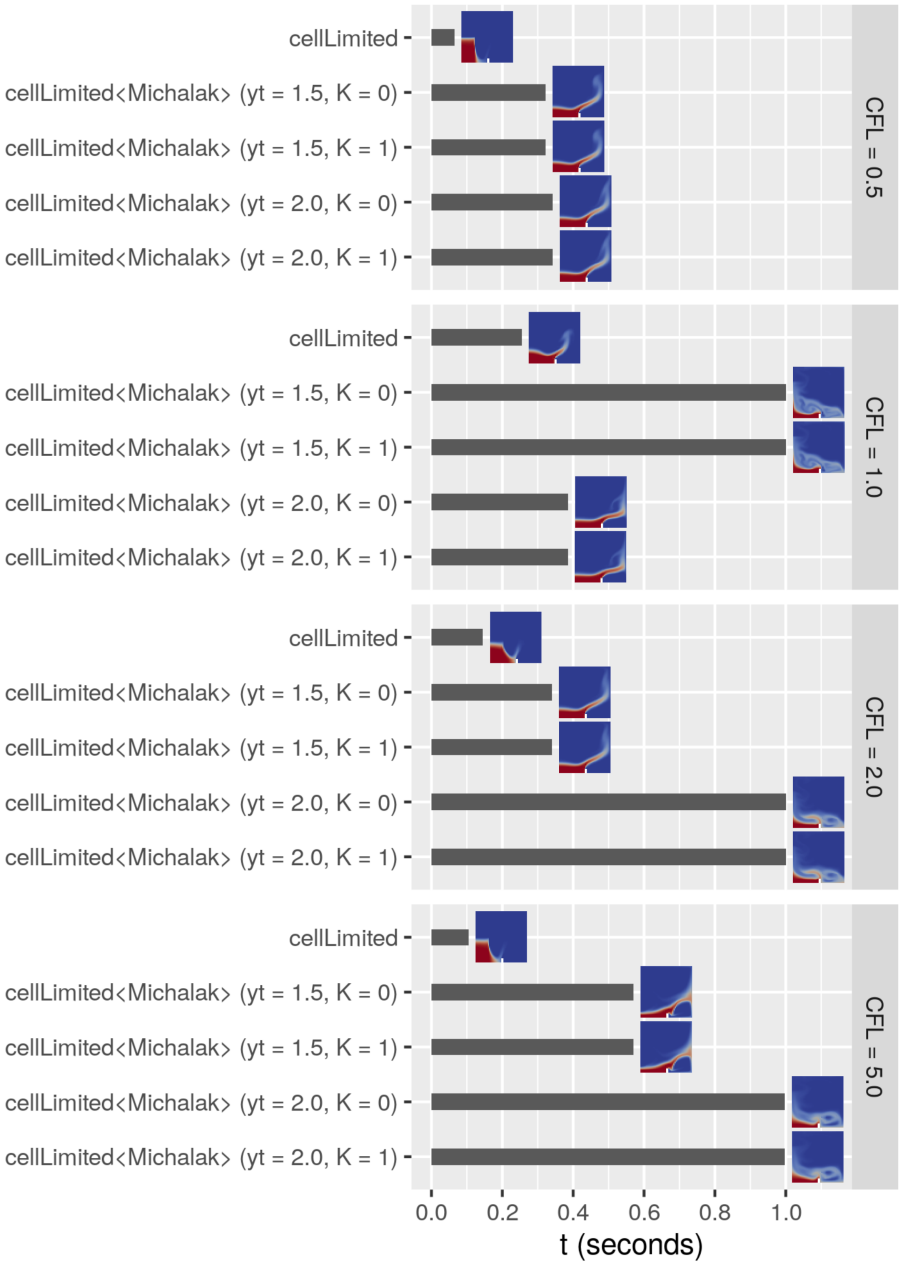

FIGURE 4. Effect of limiter and real-time Courant number on how far the simulation gets and, if convergence is not achieved, when the stall occurs in real-time. Density field shown to highlight problematic moments of the simulation.

The mesh from the `iglooWithFridges` tutorial was used, with all the boundary conditions changed to walls, and the domain filled with water up to $z = 2$, as shown in Figure 7. As before, the initial conditions for velocity and pressure were set to zero. The solver was run for one real-time step of size $\Delta t = 0.01$ seconds with the following settings: one Runge-Kutta stage, an artificial compressibility coefficient of $\beta = 1100$, and a Courant number for pseudo-time of 0.48 to satisfy the explicit stability constraint. The absolute tolerances for pseudo-time convergence were 0.01 for ρ, 0.01 for $\rho\mathbf{u}$, and 0.01 for p. The Green-Gauss gradient calculation was chosen for ρ and $\mathbf{u}$, and the corresponding limiter set to `cellLimited<Michalak>` with $y_t = 1.5$ and $K = 1$.

Recall that the pressure gradient calculation used throughout this paper is from [22]. It is based on a rearrangement of the momentum equation, and so it is automatically well-balanced, that is, is balances exactly with the gravity source term when the velocities are zero. While there are other well-balanced methods [19, 44, 75, 76, 77], this one is very easy to implement on unstructured meshes. Therefore, we

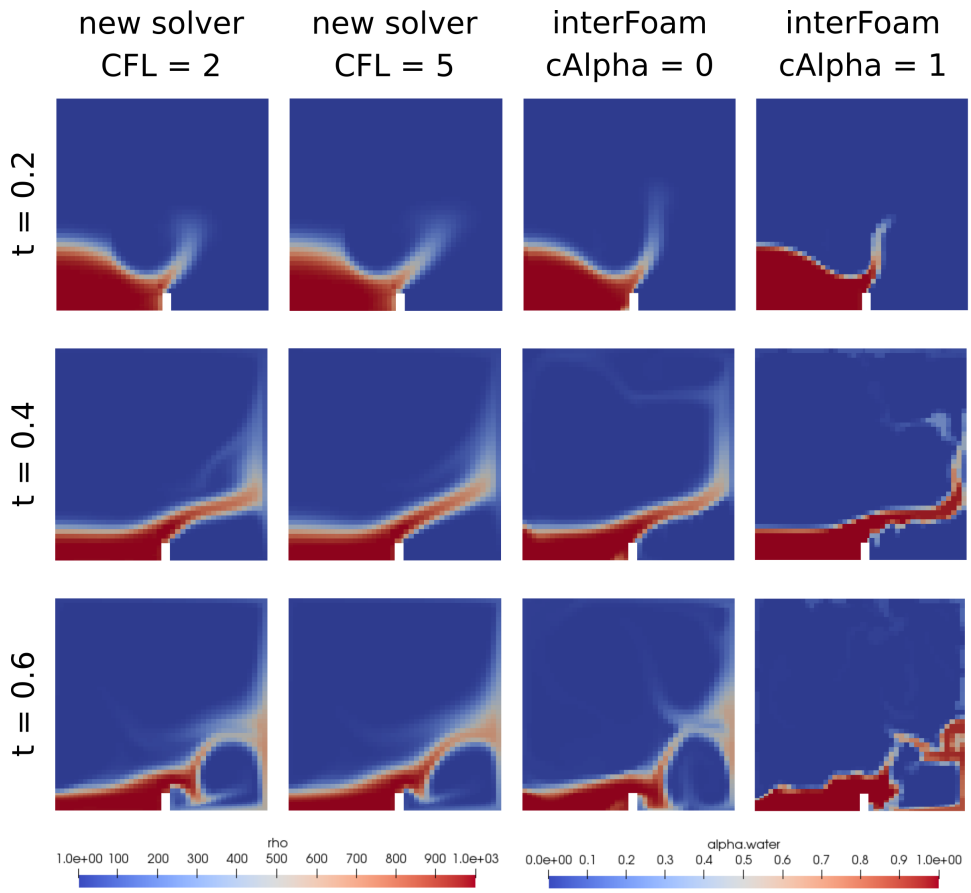

FIGURE 5. Density field for new solver compared with water phase field for `interFoam` in `damBreak` case. The Courant number is for real time.

investigate its effectiveness on unstructured meshes by comparing it to when the pressure gradient is calculated instead by `cellLimited<Michalak>` with $y_t = 1.5$ and $K = 1$.

Figure 8 shows that the method from [22] indeed performs better along the direction of gravity than the limiter `cellLimited<Michalak>`, which is too limiting. The resulting difference in the pressure and velocity in the direction of gravity is notable. The small but non-zero velocities in the other directions can be attributed to the free surface not starting out entirely flat due to the irregular shape of the mesh. Figure 9 shows that the improved performance of the well-balanced limiter comes at the cost of slower convergence, but this is primarily due to the fact that it takes longer for the pressure field to converge to the correct magnitude from $p = 0$. In any case, this test demonstrated that the new solver can be run on 3D unstructured meshes and that the pressure gradient calculation of [22] retains its well-balancedness, although not as cleanly as on a 2D Cartesian mesh.

4. CONCLUSION

In OpenFOAM® , the traditional way to model free-surface flows is with the VOF solver `interFoam`, which uses the PIMPLE algorithm based on matrix inversion. However, there are other ways to deal with the lack of a pressure equation in the incompressible Navier-Stokes equations. In this paper, we focus on the method of artificial compressibility, which scales better than matrix inversion as it requires less communication between processors. Indeed, although artificial compressibility is not new, it is currently very relevant due to its suitability for implementation on massively parallel architectures [12, 13, 14].

Despite its potential for efficient parallelisation, artificial compressibility has not been investigated extensively for variable density flows (including free-surface flows), having only been implemented on 2D and structured meshes so far [17, 19, 20, 22]. The present study harnesses OpenFOAM® to implement artificial compressibility for variable density incompressible flows on 3D unstructured meshes. We found that, since pseudo-time convergence is required at every real-time step, the slope limiter used in the MUSCL reconstruction step is critical, and the limiters currently in OpenFOAM® are not fit for this purpose. Therefore, we implemented Michalak and Ollivier-Gooch's [67] limiter fully to avail of its

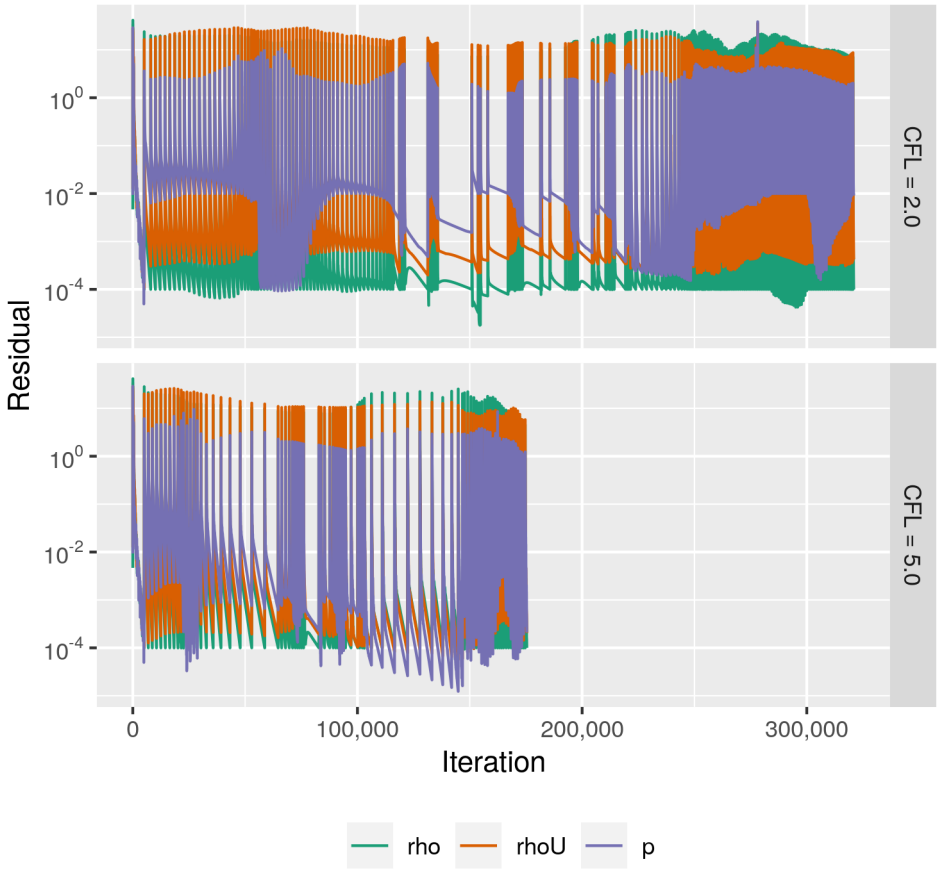

FIGURE 6. Residuals for each pseudo-time iteration required throughout full `damBreak` simulation to $t = 1$.

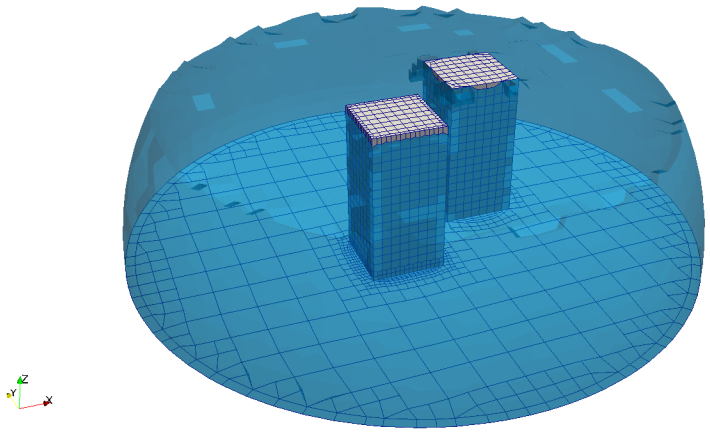

FIGURE 7. Water partially filling the igloo (converged solution with ∇p from [22]).

improved convergence properties. When the results converge, they both compare well with `interFoam` and are well-balanced, but even the improved limiter is not sufficiently robust to ensure convergence. While the solver requires a mechanism to keep the free surface sharp as in `interFoam`, the convergence problem is more urgent. Without convergence, there is no solution.

Consequently, further research should first focus on alternative routes to second-order accuracy than Barth and Jespersen's [63] framework for MUSCL. This could involve smoothing out the multidimensional

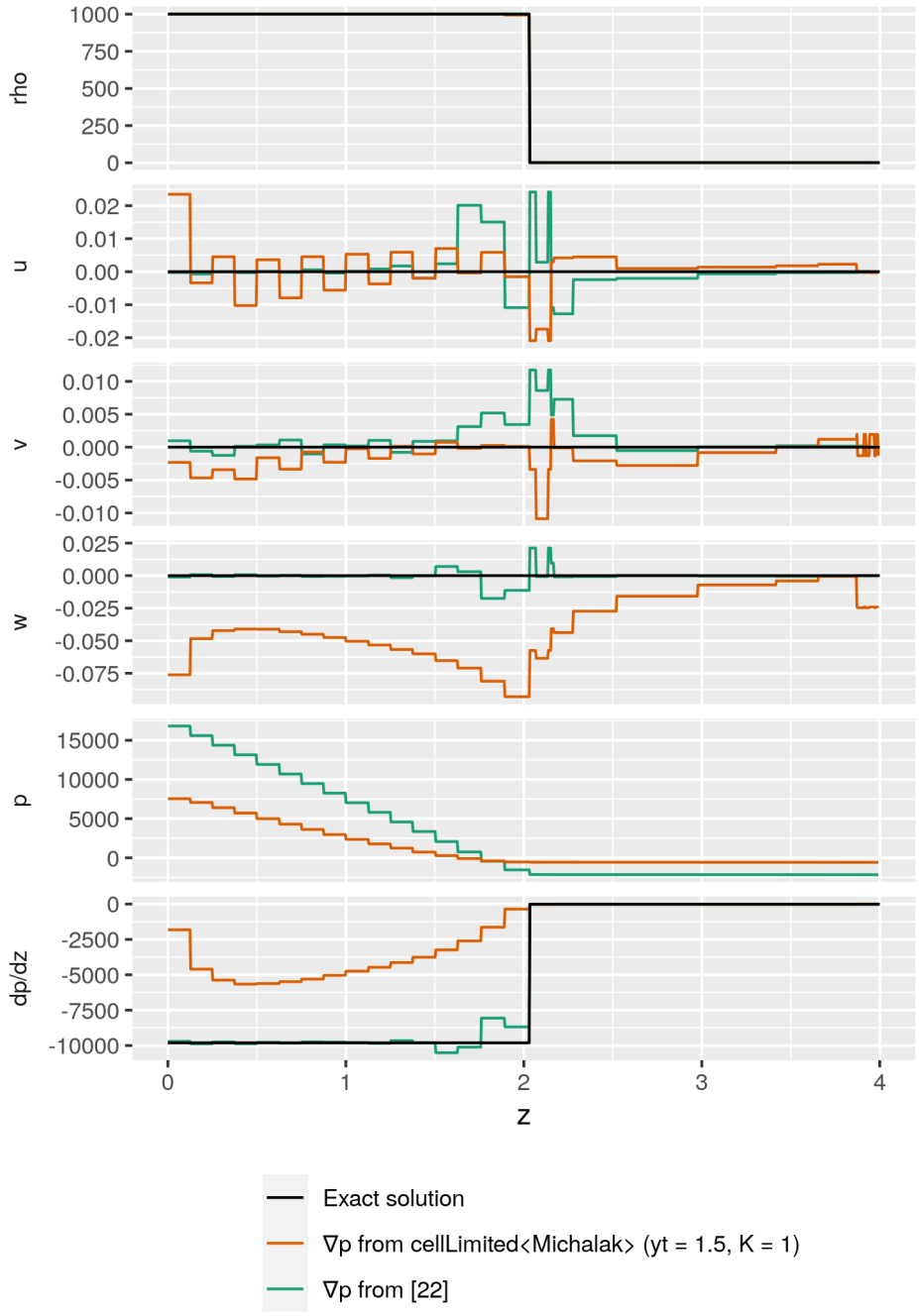

FIGURE 8. Cross section of cell-centred values along the z-axis near the centre of the igloo at $x = 3.2, y = 3$ for different ∇p calculations.

limiter `cellMDLimited` rather than `cellLimited`, or implementing the WENO (weighted essentially non-oscillatory) [78] method. Of course, convergence is necessary but not sufficient for a good solution, and so more work would need to be done after achieving robust convergence: a mechanism could be employed to keep the interface sharp, the solver could be compared thoroughly with `interFoam`, and it could be applied to practical cases. Ultimately, this will allow us to use Riemann solvers to capture the free surface automatically and in a way that is particularly suited to the massively parallel architectures of today.

ACKNOWLEDGEMENTS

The authors would like to thank the two anonymous reviewers for their thoughtful and thorough feedback. This research was funded by the Engineering and Physical Sciences Research Council grant number EP/R51309X/1.

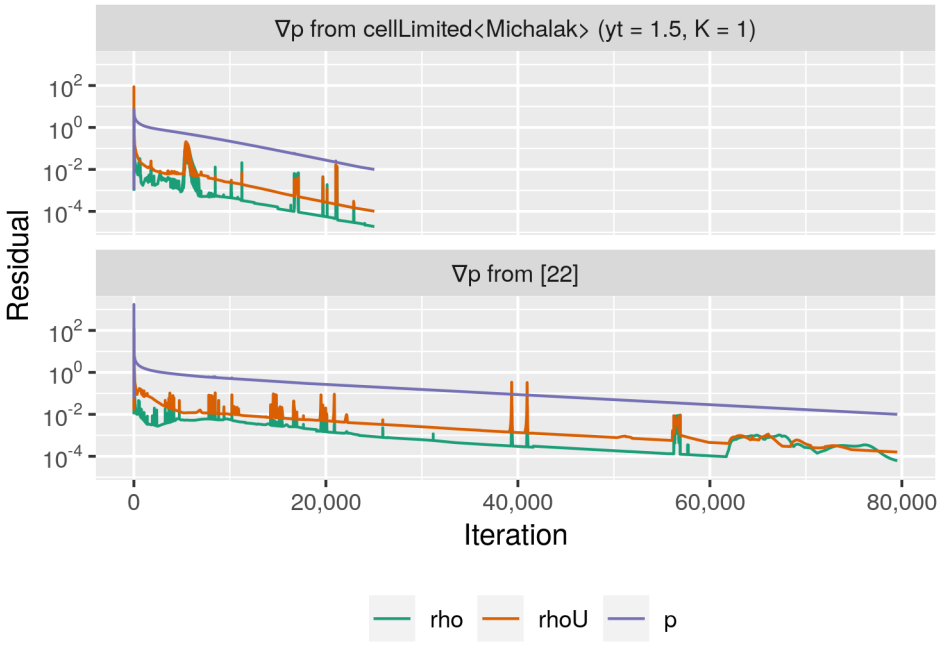

FIGURE 9. Residuals for each pseudo-time iteration required throughout hydrostatic simulation.

Author Contributions: Conceptualisation, V.G.; methodology, S.L. and V.G.; software, S.L.; validation, S.L.; formal analysis, S.L.; investigation, S.L.; resources, V.G.; data curation, S.L.; writing—original draft preparation, S.L.; writing—review and editing, V.G. and C.J.M.H.; visualisation, S.L.; supervision, V.G. and C.J.M.H.; project administration, S.L. and C.J.M.H.; funding acquisition, C.J.M.H. All authors have read and agreed to the published version of the manuscript.

REFERENCES

[1] A. Zhainakov and A. Kurbanaliev, "Verification of the open package OpenFOAM on dam break problems," *Thermophysics and Aeromechanics*, vol. 20, no. 4, pp. 451–461, 2013. [Online]. Available: https://doi.org/10.1134/S0869864313040082

[2] A. Bayon-Barrachina and P. A. Lopez-Jimenez, "Numerical analysis of hydraulic jumps using OpenFOAM," *Journal of Hydroinformatics*, vol. 17, no. 4, pp. 662–678, 03 2015. [Online]. Available: https://doi.org/10.2166/hydro.2015.041

[3] A. Patil, "Numerical investigation of the effect of nappe non-aeration on caisson sliding force during tsunami breakwater over-topping using OpenFOAM," 2018, TU Delft. [Online]. Available: http://resolver.tudelft.nl/uuid:65aff400-8610-4867-92fa-ca33a9e54a01

[4] M. Maza, J. L. Lara, and I. J. Losada, "Tsunami wave interaction with mangrove forests: A 3-D numerical approach," *Coastal Engineering*, vol. 98, pp. 33–54, 2015. [Online]. Available: https://doi.org/10.1016/j.coastaleng.2015.01.002

[5] S. Leakey, C. J. M. Hewett, V. Glenis, and P. F. Quinn, "Investigating the behaviour of leaky barriers with flume experiments and 3D modelling," in *Proceedings of SimHydro 2021: Models for complex and global water issues*, 2021.

[6] A. J. Chorin, "A numerical method for solving incompressible viscous flow problems," *Journal of Computational Physics*, vol. 135, no. 2, pp. 118–125, 1997. [Online]. Available: https://doi.org/10.1006/jcph.1997.5716

[7] D. D. Drikakis and W. Rider, *High-resolution Methods for Incompressible and Low-Speed Flows*, ser. Computational fluid and solid mechanics. New York: Springer-Verlag, 2005. [Online]. Available: https://doi.org/10.1007/b137615

[8] E. F. Toro, *Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction*. New York: Springer, 2009. [Online]. Available: https://doi.org/10.1007/b79761

[9] D. Kwak and C. Kiris, *Computation of Viscous Incompressible Flows*, ser. Scientific Computation. New York: Springer, 2011. [Online]. Available: https://doi.org/10.1007/978-94-007-0193-9

[10] S. Patankar and D. Spalding, "A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows," *International Journal of Heat and Mass Transfer*, vol. 15, no. 10, pp. 1787–1806, 1972. [Online]. Available: https://doi.org/10.1016/0017-9310(72)90054-3

[11] R. Issa, "Solution of the implicitly discretised fluid flow equations by operator-splitting," *Journal of Computational Physics*, vol. 62, no. 1, pp. 40–65, 1986. [Online]. Available: https://doi.org/10.1016/0021-9991(86)90099-9

[12] B. R. Hodges, "An artificial compressibility method for 1D simulation of open-channel and pressurized-pipe flow," *Water*, vol. 12, no. 6, p. 1727, 2020. [Online]. Available: https://doi.org/10.3390/w12061727

[13] R. Nourgaliev, T. Dinh, and T. Theofanous, "A pseudocompressibility method for the numerical simulation of incompressible multifluid flows," *International Journal of Multiphase Flow*, vol. 30, no. 7, pp. 901–937, 2004. [Online]. Available: https://doi.org/10.1016/j.ijmultiphaseflow.2004.03.010

[14] N. A. Loppi, "High-order incompressible computational fluid dynamics on modern hardware architectures," Ph.D. dissertation, Imperial College London, 2019. [Online]. Available: https://doi.org/10.25560/73888
[15] O. F. Oxtoby, A. G. Malan, and J. A. Heyns, "A computationally efficient 3D finite-volume scheme for violent liquid–gas sloshing," *International Journal for Numerical Methods in Fluids*, vol. 79, no. 6, pp. 306–321, 2015. [Online]. Available: https://doi.org/10.1002/fld.4055
[16] E. Shapiro and D. Drikakis, "Artificial compressibility, characteristics-based schemes for variable density, incompressible, multi-species flows. Part I. Derivation of different formulations and constant density limit," *Journal of Computational Physics*, vol. 210, no. 2, pp. 584–607, 2005. [Online]. Available: https://doi.org/10.1016/j.jcp.2005.05.001
[17] F. Kelecy and R. Pletcher, "The development of a free surface capturing approach for multidimensional free surface flows in closed containers," *Journal of Computational Physics*, vol. 138, no. 2, pp. 939–980, 1997. [Online]. Available: https://doi.org/10.1006/jcph.1997.5847
[18] C. W. Hirt and B. D. Nichols, "Volume of fluid (VOF) method for the dynamics of free boundaries," *Journal of Computational Physics*, vol. 39, no. 1, pp. 201–225, 1981. [Online]. Available: https://doi.org/10.1016/0021-9991(81)90145-5
[19] L. Qian, D. M. Causon, C. G. Mingham, and D. M. Ingram, "A free-surface capturing method for two fluid flows with moving bodies," *Proceedings: Mathematical, Physical and Engineering Sciences*, vol. 462, no. 2065, pp. 21–42, 2006. [Online]. Available: https://doi.org/10.1098/rspa.2005.1528
[20] F. Bassi, F. Massa, L. Botti, and A. Colombo, "Artificial compressibility Godunov fluxes for variable density incompressible flows," *Computers and Fluids*, vol. 169, pp. 186–200, 2018. [Online]. Available: https://doi.org/10.1016/j.compfluid.2017.09.010
[21] F. C. Massa, F. Bassi, L. Botti, and A. Colombo, "An implicit high-order discontinuous Galerkin approach for variable density incompressible flows," in *Droplet Interactions and Spray Processes*, ser. Fluid Mechanics and Its Applications. Cham: Springer International Publishing, 2020, vol. 121, pp. 191–202.
[22] S. Leakey, V. Glenis, and C. J. M. Hewett, "Riemann solvers and pressure gradients in Godunov-type schemes for variable density incompressible flows," 2021. [Online]. Available: https://arxiv.org/abs/2108.08769
[23] F. Gao, D. M. Ingram, D. M. Causon, and C. G. Mingham, "The development of a Cartesian cut cell method for incompressible viscous flows," *International Journal for Numerical Methods in Fluids*, vol. 54, no. 9, pp. 1033–1053, 2007. [Online]. Available: https://doi.org/10.1002/fld.1409
[24] ——, "Numerical modelling of wave interaction with porous structures," in *New Trends in Fluid Mechanics Research*, F. G. Zhuang and J. C. Li, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 514–517. [Online]. Available: https://doi.org/10.1007/978-3-540-75995-9_171
[25] D. Ingram, F. Gao, D. Causon, C. Mingham, and P. Troch, "Numerical investigations of wave overtopping at coastal structures," *Coastal Engineering*, vol. 56, no. 2, pp. 190–202, 2009. [Online]. Available: https://doi.org/10.1016/j.coastaleng.2008.03.010
[26] L. Qian, D. M. Causon, D. M. Ingram, and C. G. Mingham, "A two-fluid solver for hydraulic applications," in *Hydraulic Information Management*, C. A. Brebbia and W. R. Blain, Eds. WIT Press, 2002. [Online]. Available: https://www.witpress.com/elibrary/wit-transactions-on-ecology-and-the-environment/52/466
[27] D. Causon, C. Mingham, D. Ingram, and L. Qian, "Numerical experiments on slamming of rigid wedge-shaped bodies," ser. International Ocean and Polar Engineering Conference, 2004. [Online]. Available: https://onepetro.org/ISOPEIOPEC/proceedings-pdf/ISOPE04/All-ISOPE04/ISOPE-I-04-438/1852536/isope-i-04-438.pdf
[28] Z. Ma, L. Qian, D. Causon, and C. Mingham, "Simulation of solitary breaking waves using a two-fluid hybrid turbulence approach," ser. International Ocean and Polar Engineering Conference, 2011. [Online]. Available: https://onepetro.org/ISOPEIOPEC/proceedings-pdf/ISOPE11/All-ISOPE11/ISOPE-I-11-070/1668843/isope-i-11-070.pdf
[29] W.-H. Wang and Y.-Y. Wang, "An improved free surface capturing method based on Cartesian cut cell mesh for water-entry and -exit problems," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 465, no. 2106, pp. 1843–1868, 2009. [Online]. Available: https://doi.org/10.1098/rspa.2008.0410
[30] ——, "An essential solution of water entry problems and its engineering applications," *Journal of Marine Science and Application*, vol. 9, pp. 268–273, 2010. [Online]. Available: https://doi.org/10.1007/s11804-010-1006-5
[31] ——, "Calculation of water entry problem for free-falling bodies using a developed Cartesian cut cell mesh," *AIP Conference Proceedings*, vol. 1233, no. 1, pp. 590–595, 2010. [Online]. Available: https://doi.org/10.1063/1.3452239
[32] ——, "Analysis of mechanical energy transport on free-falling wedge during water-entry phase," *Journal of Applied Mathematics*, 2012. [Online]. Available: https://doi.org/10.1155/2012/738082
[33] Y. Wu, W.-H. Wang, Y. Huang, and Y.-Y. Wang, "Further application of surface capturing method and Cartesian cut cell mesh on hydroelastic water-entry problems of free-falling elastic wedge," *Mathematical Problems in Engineering*, 2014. [Online]. Available: https://doi.org/10.1155/2014/545642
[34] S. Shin, S. Y. Bae, I. C. Kim, Y. J. Kim, and H. K. Yoon, "Simulation of free surface flows using the flux-difference splitting scheme on the hybrid Cartesian/immersed boundary method," *International Journal for Numerical Methods in Fluids*, vol. 68, no. 3, pp. 360–376, 2012. [Online]. Available: https://doi.org/10.1002/fld.2519
[35] S. Shin, "Simulation of two-dimensional internal waves generated by a translating and pitching foil," *Ocean Engineering*, vol. 72, pp. 77–86, 2013. [Online]. Available: https://doi.org/10.1016/j.oceaneng.2013.06.011
[36] G. Y. Evtushok, A. V. Boiko, S. N. Yakovenko, E. E. Takovenko, and K. C. Chang, "Modification and verification of numerical algorithms for dam-break flow over a horizontal bed," *Journal of Applied Mechanics and Technical Physics*, vol. 62, pp. 255–261, 2021. [Online]. Available: https://doi.org/10.1134/S0021894421020097
[37] E. Shapiro and D. Drikakis, "Artificial compressibility, characteristics-based schemes for variable-density, incompressible, multispecies flows: Part II. Multigrid implementation and numerical tests," *Journal of Computational Physics*, vol. 210, no. 2, pp. 608–631, 2005. [Online]. Available: https://doi.org/10.1016/j.jcp.2005.05.002
[38] Z. Z. Hu, D. M. Causon, C. G. Mingham, and L. Qian, "A Cartesian cut cell free surface capturing method for 3D water impact problems," *International Journal for Numerical Methods in Fluids*, vol. 71, no. 10, pp. 1238–1259, 2013. [Online]. Available: https://doi.org/10.1002/fld.3708

[39] J. Manzanero, G. Rubio, D. A. Kopriva, E. Ferrer, and E. Valero, "An entropy–stable discontinuous Galerkin approximation for the incompressible Navier–Stokes equations with variable density and artificial compressibility," *Journal of Computational Physics*, vol. 408, p. 109241, May 2020. [Online]. Available: http://doi.org/10.1016/j.jcp.2020.109241

[40] W.-H. Wang, Y.-Z. Du, L.-L. Wang, Y.-Y. Wang, and Y. Huang, "Novel numerical method to simulate hydrodynamic characteristic of moving body through free surface and stratified-fluid interface," *Ocean Engineering*, vol. 205, p. 107234, 2020. [Online]. Available: https://doi.org/10.1016/j.oceaneng.2020.107234

[41] Y. Zhao, H. Hui Tan, and B. Zhang, "A high-resolution characteristics-based implicit dual time-stepping VOF method for free surface flow simulation on unstructured grids," *Journal of Computational Physics*, vol. 183, no. 1, pp. 233–273, 2002. [Online]. Available: https://doi.org/10.1006/jcph.2002.7196

[42] A. N. Sambe, F. Golay, D. Sous, P. Fraunié, V. Rey, R. Marcer, and C. De Jouette, "Two-phase-flow unstructured grid solver: Application to tsunami wave impact," *International Journal of Offshore and Polar Engineering*, vol. 21, no. 03, 09 2011. [Online]. Available: https://onepetro.org/IJOPE/article-pdf/2189573/isope-11-21-3-186.pdf

[43] S. Bhat and J. C. Mandal, "Contact preserving Riemann solver for incompressible two-phase flows," *Journal of Computational Physics*, vol. 379, pp. 173–191, 2019. [Online]. Available: https://doi.org/10.1016/j.jcp.2018.10.039

[44] D. Ntouras and G. Papadakis, "A coupled artificial compressibility method for free surface flows," *Journal of Marine Science and Engineering*, vol. 8, no. 8, p. 590, 2020. [Online]. Available: https://doi.org/10.3390/jmse8080590

[45] T. Hino, "An interface capturing method for free surface flow computations on unstructured grids," *Journal of The Society of Naval Architects of Japan*, vol. 1999, no. 186, pp. 177–183, 1999. [Online]. Available: https://doi.org/10.2534/jjasnaoe1968.1999.186_177

[46] N. Evstigneev, *Advances in Fluid Mechanics VII*, ser. WIT Transactions on Engineering Sciences. WIT, 2008, ch. Integration of 3D incompressible free surface Navier-Stokes equations on unstructured tetrahedral grid using distributed computation on TCP/IP networks, pp. 65–77. [Online]. Available: https://books.google.co.uk/books?id=DeOgAQAAQBAJ

[47] X. Lv, Q. Zou, D. Reeve, and Z. Wang, "An Unstructured 3D LES Solver For Free Surface Flow And Breaking Waves," ser. International Ocean and Polar Engineering Conference, 2008. [Online]. Available: https://onepetro.org/ISOPEIOPEC/proceedings-pdf/ISOPE08/All-ISOPE08/ISOPE-I-08-087/1803502/isope-i-08-087.pdf

[48] T. D. Economon, "Simulation and adjoint-based design for variable density incompressible flows with heat transfer," *AIAA Journal*, vol. 58, no. 2, pp. 757–769, 2020. [Online]. Available: https://doi.org/10.2514/1.J058222

[49] J. Wackers and B. Koren, "A surface capturing method for the efficient computation of steady water waves," *Journal of Computational and Applied Mathematics*, vol. 215, no. 2, pp. 618–625, 2008, proceedings of the Third International Conference on Advanced Computational Methods in Engineering (ACOMEN 2005). [Online]. Available: https://doi.org/10.1016/j.cam.2006.03.056

[50] S. N. Yakovenko and K. C. Chang, "Volume fraction flux approximation in a two-fluid flow," *Thermophysics and Aeromechanics*, vol. 15, pp. 169–186, 2008. [Online]. Available: https://doi.org/10.1134/S0869864308020017

[51] H. Gough, A. L. Gaitonde, and D. P. Jones, "A dual-time central-difference interface-capturing finite volume scheme applied to cavitation modelling," *International Journal for Numerical Methods in Fluids*, vol. 66, no. 4, pp. 452–485, 2011. [Online]. Available: https://doi.org/10.1002/fld.2265

[52] Y. G. Chen, W. G. Price, and P. Temarel, "An anti-diffusive volume of fluid method for interfacial fluid flows," *International Journal for Numerical Methods in Fluids*, vol. 68, no. 3, pp. 341–359, 2012. [Online]. Available: https://doi.org/10.1002/fld.2509

[53] A. Yang, S. Chen, L. Yang, and X. Yang, "An upwind finite volume method for incompressible inviscid free surface flows," *Computers and Fluids*, vol. 101, pp. 170–182, 2014. [Online]. Available: https://doi.org/10.1016/j.compfluid.2014.06.013

[54] M. Mortezazadeh and K. Hejranfar, "Simulation of incompressible multiphase flows using the artificial compressibility method," ser. Fluids Engineering Division Summer Meeting, vol. 2, 2018, v002T09A001. [Online]. Available: https://doi.org/10.1115/FEDSM2018-83013

[55] S. Bhat and J. C. Mandal, "Modified HLLC-VOF solver for incompressible two-phase fluid flows," 2019. [Online]. Available: https://arxiv.org/abs/1911.11234

[56] S. Parameswaran and J. Mandal, "A novel Roe solver for incompressible two-phase flow problems," *Journal of Computational Physics*, vol. 390, pp. 405–424, 2019. [Online]. Available: https://doi.org/10.1016/j.jcp.2019.04.012

[57] D. Pan and C. Chang, "The capturing of free surfaces in incompressible multi-fluid flows," *International Journal for Numerical Methods in Fluids*, vol. 33, no. 2, pp. 203–222, 2000. [Online]. Available: https://doi.org/10.1002/(SICI)1097-0363(20000530)33:2%3C203::AID-FLD9%3E3.0.CO;2-F

[58] C. de Jouëtte, O. Laget, J.-M. Le Gouez, and H. Viviand, "A dual time stepping method for fluid–structure interaction problems," *Computers and Fluids*, vol. 31, no. 4-7, pp. 509–537, 2002. [Online]. Available: http://doi.org/10.1016/S0045-7930(01)00068-8

[59] A. Kajzer and J. Pozorski, "A weakly compressible, diffuse-interface model for two-phase flows," *Flow, Turbulence and Combustion*, vol. 105, no. 2, pp. 299–333, 2020. [Online]. Available: https://doi.org/10.1007/s10494-020-00164-8

[60] R. Yang, H. Li, and A. Yang, "A HLLC-type finite volume method for incompressible two-phase flows," *Computers and Fluids*, vol. 213, p. 104715, 2020. [Online]. Available: https://doi.org/10.1016/j.compfluid.2020.104715

[61] H. Jasak, "dbnsfoam, foam-extend 4.0," 2014. [Online]. Available: http://www.foam-extend.org

[62] M. Darwish and F. Moukalled, "TVD schemes for unstructured grids," *International Journal of Heat and Mass Transfer*, vol. 46, no. 4, pp. 599–611, 2003. [Online]. Available: https://doi.org/10.1016/S0017-9310(02)00330-7

[63] T. Barth and D. Jespersen, "The design and application of upwind schemes on unstructured meshes," in *27th Aerospace Sciences Meeting*, 1989. [Online]. Available: https://doi.org/10.2514/6.1989-366

[64] J. Blazek, *Computational Fluid Dynamics: Principles and Applications*, 2nd ed. Elsevier, 2005. [Online]. Available: https://doi.org/10.1016/C2013-0-19038-1

[65] S. Spekreijse, "Multigrid solution of monotone second-order discretizations of hyperbolic conservation laws," *Mathematics of Computation*, vol. 49, no. 179, pp. 135–155, 1987. [Online]. Available: https://doi.org/10.2307/2008254

[66] V. Venkatakrishnan, "Convergence to steady state solutions of the Euler equations on unstructured grids with limiters," *Journal of Computational Physics*, vol. 118, no. 1, pp. 120–130, 1995. [Online]. Available: https://doi.org/10.1006/jcph.1995.1084

[67] K. Michalak and C. Ollivier-Gooch, "Limiters for unstructured higher-order accurate solutions of the Euler equations," in *46th AIAA Aerospace Sciences Meeting and Exhibit*, 2008. [Online]. Available: https://doi.org/10.2514/6.2008-776

[68] M. Aftosmis, D. Gaitonde, and T. S. Tavares, "Behavior of linear reconstruction techniques on unstructured meshes," *AIAA Journal*, vol. 33, no. 11, pp. 2038–2049, 1995. [Online]. Available: https://doi.org/10.2514/3.12945

[69] W. K. Anderson, J. L. Thomas, and B. Van Leer, "Comparison of finite volume flux vector splittings for the Euler equations," *AIAA Journal*, vol. 24, no. 9, pp. 1453–1460, 1986. [Online]. Available: https://doi.org/10.2514/3.9465

[70] M. Berger, M. Aftosmis, and S. Muman, "Analysis of slope limiters on irregular grids," in *43rd AIAA Aerospace Sciences Meeting and Exhibit*, 2005. [Online]. Available: https://arc.aiaa.org/doi/abs/10.2514/6.2005-490

[71] G. Li, D. Bhatia, and J. Wang, "Compressive properties of min-mod-type limiters in modelling shockwave-containing flows," *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, vol. 42, no. 6, p. 290, 2020. [Online]. Available: https://doi.org/10.1007/s40430-020-02374-7

[72] J. S. Park and C. Kim, "Multi-dimensional limiting process for finite volume methods on unstructured grids," *Computers and Fluids*, vol. 65, pp. 8–24, 2012. [Online]. Available: https://doi.org/10.1016/j.compfluid.2012.04.015

[73] J. S. Park, S.-H. Yoon, and C. Kim, "Multi-dimensional limiting process for hyperbolic conservation laws on unstructured grids," *Journal of Computational Physics*, vol. 229, no. 3, pp. 788–812, 2010. [Online]. Available: https://doi.org/10.1016/j.jcp.2009.10.011

[74] H. Rusche, "Computational fluid dynamics of dispersed two-phase flows at high phase fractions," Ph.D. dissertation, Imperial College London, 2002. [Online]. Available: http://hdl.handle.net/10044/1/8110

[75] A. Kruisbrink, F. Pearce, T. Yue, and H. Morvan, "An SPH multi-fluid model based on quasi buoyancy for interface stabilization up to high density ratios and realistic wave speed ratios," *International Journal for Numerical Methods in Fluids*, vol. 87, no. 10, pp. 487–507, 2018. [Online]. Available: https://doi.org/10.1002/fld.4498

[76] V. Vukčević, J. Roenby, I. Gatin, and H. Jasak, "A sharp free surface finite volume method applied to gravity wave flows," 2018. [Online]. Available: https://arxiv.org/abs/1804.01130

[77] P. Queutey and M. Visonneau, "An interface capturing method for free-surface hydrodynamic flows," *Computers and Fluids*, vol. 36, no. 9, pp. 1481–1510, 2007. [Online]. Available: https://doi.org/10.1016/j.compfluid.2006.11.007

[78] X.-D. Liu, S. Osher, and T. Chan, "Weighted essentially non-oscillatory schemes," *Journal of Computational Physics*, vol. 115, no. 1, pp. 200–212, 1994. [Online]. Available: https://doi.org/10.1006/jcph.1994.1187