PDE-based methods for multiscale gas dynamics simulations

By

RAAGHAV RAMANI

DISSERTATION

Submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

APPLIED MATHEMATICS

in the

OFFICE OF GRADUATE STUDIES

of the

UNIVERSITY OF CALIFORNIA

DAVIS

Approved:

Steve Shkoller, Chair

John Hunter

Becca Thomases

Committee in Charge

2023

To my parents

Contents

PDE-based methods for multiscale gas dynamics simulations

Abstract

In this dissertation, we develop a numerical simulation framework for multiscale fluid flows in multiple space dimensions. We couple a robust front-capturing approach with an efficient front-tracking scheme; adaptivity is incorporated via a novel adaptive meshing procedure. The work presented is comprised of four articles published in the Journal of Computational Physics. We begin with a summary of the research performed, introducing the problems under consideration and an overview of our framework and its principal components. We summarize the new methods developed, their mathematical formulations and associated algorithmic implementations, and the results obtained. Each of the three subsequent chapters then describes in detail a particular component of the multiscale framework.

Acknowledgments

This thesis would not have been possible without my advisor, Steve Shkoller. I would like to thank him for the suggestions of interesting problems to study, for his novel insights into how to solve them, and for his patience while I figured out how to do so.

I am also thankful for the other members of my committee, John Hunter and Becca Thomases. A special thanks to Becca for also chairing my qualifying exam committee and for all her help as I navigate the next stages of my career.

Many thanks to Jon Reisner at Los Alamos National Lab for funding my work and for providing me with numerous opportunities for research and career development at the lab.

I would like to express my gratitude to the mathematics department staff; specifically, Tina Denena, Sarah Driver (former), Victoria Whistler (former), Vanessa Bravo (former), Diana Bond, and Matt Silver.

Thanks to my friends, old and new, near and far, for the good times passed and the good times to come.

And finally, thank you to my family: to my sister, Ranjani, and my brother-in-law, Arvind, for all their support; and to my parents, for everything and more.

CHAPTER 1

Introduction

1. Multiscale fluid flows

The focus of this dissertation is the development of robust numerical algorithms for simulating multiscale flows in gas dynamics.

1.1. Two motivating experiments

An example of the type of problem we are interested in is shown in Figure 1. This image is from an experiment conducted in 1959 by D. W. Boyer [3]. The experimental setup is as follows: a 3 ft diameter spherical "shock chamber" contains a 2 in diameter sphere made of soda-lime glass and filled with pressurized air. The sphere is hand-blown and fitted with a small hollow stem that is then cemented to a pipe (shown at the top of the figure), suspending the sphere in the center of the shock chamber. Air is pumped into the glass sphere through the pipe until the pressure inside reaches 400 psi, roughly the same as the pressure at a depth of 275 m underwater. The experiment is initiated when a spring-loaded mallet, visible at the bottom of the figure, strikes the glass sphere, shattering it. The resulting gas flow, driven by the escaping pressurized air, is visualized along a two-dimensional slice using shadowgraph photography.

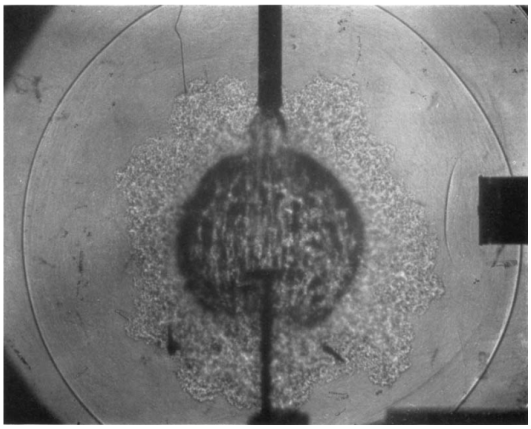


Figure 1: Shadowgraph taken 300 μsec after initiation in the exploding glass sphere experiment. Figure taken from [3].

The *multiscale* nature of the flow is clearly demonstrated in Figure 1. The shattering of the

glass sphere generates a spherical *shock wave*: a thin[1] adjustment layer across which certain fluid properties (such as pressure, density, temperature, flow velocity, etc.) vary rapidly. This shock wave then propagates radially outwards into the surrounding air. Meanwhile, the compressed air in the sphere is expanded through a spherical *rarefaction wave* that propagates radially inwards. Lying between the two waves, separating the air compressed by the shock from the expanded sphere gas, is another wave: a *contact discontinuity*. The contact discontinuity is inherently *unstable*; the initial spherically symmetric flow quickly develops asymmetries, visualized in Figure 1 as the small whorls and eddies of the contact surface. The instability is enhanced by the asymmetries introduced by the experimental apparatus: the pipe, the cemented hollow stem, and the fragments of shattered glass flowing through the air. Eventually, we observe the emergence of a *turbulent mixing region* containing flow structures across a wide range of spatial scales.

In 1960, less than a year after Boyer's experiments, the laser was invented. It was quickly realized that this new technology appeared to be the perfect mechanism for driving *inertial fusion* devices, a concept proposed in 1957 by J. Nuckolls. In a 1972 paper, Nuckolls introduced laser-based *inertial confinement fusion* (ICF) [20]. The numerical simulation in Figure 2 is taken from [6] and is one of several described therein, with each simulation representing an experiment that might be conducted at the National Ignition Facility (NIF), the world's largest and most powerful ICF device [10].

The NIF is the size of 3 football fields; the bulk of the facility is occupied by an array of 192 beamlines forming a laser delivery system. The beamlines are connected to a 10 m diameter spherical target chamber which contains, at its center, a 1 cm cylindrical cavity made of gold and depleted uranium known as a *hohlraum*. Inside the hohlraum is a 2 mm diameter spherical *fuel capsule* composed of a thin layer of frozen Deuterium-Tritium (DT) ice surrounded by a diamond *ablator*. The fuel capsule is fitted with a *fill tube* through which DT gas is pumped, then suspended at the center of the hohlraum via a *capsule support tent*.

An experiment at NIF begins with the passage of the 192 laser beams through the beamlines, where they are energized to 30 quadrillion watts. The lasers are then focused into the center of the target chamber, where they enter the hohlraum and reflect off the internal walls, creating an x-ray bath that turns the surface of the fuel capsule into a superheated plasma. The surface explodes

[1] Shock waves in air have been measured to have a width of approximately 200 nm, which is roughly 4 times the mean free path of air molecules and 100 times smaller than the width of a human hair.

outwards, which, in turn, causes the DT gas to *implode* inwards: the gas is compressed down to a core half the width of a human hair. At peak compression, the pressure/temperature of the DT gas is more than two times (respectively, five times) the pressure/temperature at the center of the sun. Large enough compression of the gas at high enough temperatures for a sufficiently long amount of time then results in *ignition*: a series of self-sustaining fusion reactions and a thermonuclear burn wave that replicate, in a laboratory on Earth, the energy production method that powers the stars.

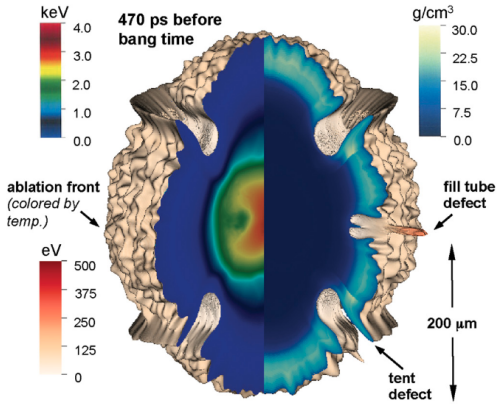


Figure 2: Numerical simulation of an ICF experiment at the NIF. Figure taken from [6].

The key to achieving ignition is maintaining a highly uniform compression of the DT gas; controlling the instability at the ablation front is the main difficulty in doing so. As shown in Figure 2, asymmetries introduced by the fill tube and the capsule support tent cause the ablation front to penetrate into the interior "hot spot", which is then cooled down and cannot ignite. Enormous resources are invested into removing the asymmetries seeding the instability: the capsule support tent is manufactured to be just 12 nm thick, the diameter of the fill tube is just 2 μm, and the surface of the ablator is meticulously smoothed to remove tiny non-uniformities in the capsule. In addition, the lasers are focused into the hohlraum in a carefully designed sequence of pulses, which, in turn, generates a set of converging shock waves that help suppress the instability [6, 30]. The purpose of numerical simulations like the one shown in Figure 2 is to help inform the specific design choices for an ignition experiment; these simulations are incredibly computationally expensive, requiring roughly one month to complete while running on six thousand processors [6].

In December 2022, NIF conducted experiment N221204, in which ignition was achieved for the first time. Separating N221204 and Boyer's experiments are six decades of advances in theoretical, computational, and experimental physics; mathematics; computer science and hardware; mechanical engineering; materials science; laser science, and numerous other scientific fields. And yet, remarkably, the two experiments share many of the same qualitative features.

Multiscale fluid flows are ubiquitous. They are observed in: atmospheric flows, such as cyclones, jet streams, and wildfires [31, 16]; astrophysical phenomena, such as relativistic jets, star/galaxy

formation, and black hole accretion disks [11, 12, 13, 32, 9]; and industrial applications, such as fusion reactors, combustion engines, and wind flow around turbines [17, 2, 15, 22]. Already, this short and entirely non-exhaustive list of applications indicates the necessity of having robust numerical algorithms for simulating such flows. However, capturing their wide range of dynamically evolving and interacting spatial and temporal scales efficiently and accurately is a formidable problem that usually requires massive supercomputing resources. Even with such resources, it is the case that many, if not most, problems of interest remain out of reach of traditional numerical algorithms [18].

1.2. A PDE-based simulation framework

Mathematically, the dynamics of fluids are described by *nonlinear hyperbolic systems of conservation laws*; the prototypical example is the compressible Euler system of gas dynamics, which we write in conservation law form as

$$\partial_t \rho + \operatorname{div}(\rho u) = 0\,, \tag{1a}$$

$$\partial_t (\rho u) + \operatorname{div}(\rho u \otimes u) + \nabla p = 0\,, \tag{1b}$$

$$\partial_t E + \operatorname{div}(u(E+p)) = 0\,, \tag{1c}$$

where $\otimes$ denotes the tensor product, and $\operatorname{div} M$ denotes the row-wise divergence of a matrix M. The velocity vector is $u = \left(u^1, \ldots, u^d\right)$, $\rho > 0$ is the fluid density (assumed strictly positive), E denotes the energy, and p is the pressure defined by an equation of state.

The nonlinear nature of wave transport in the Euler system is the key mechanism by which multiple spatial and temporal scales develop in the flow. For instance, steepening of sound waves is the mechanism by which shock formation occurs [4, 19]. Interactions between vorticity waves lead to the onset of shearing instabilities and the generation of small-scale vortical structures [1]. Additional complex physics, such as combustion chemistry, often introduce further small-scale solution structures, such as reaction zones, cellular detonation structures, and combustion instabilities [8]. The Euler system already captures many of the fundamental difficulties with simulating solutions to multidimensional nonlinear conservation laws, and we shall focus our attention on this system henceforth.

1.2.1. Designing SAFE numerical algorithms. Before we describe our multiscale simulation framework, we first provide some guiding design goals.

1. First, we require our algorithms to be **accurate**. The numerical solutions constructed must exhibit: sharp, but non-oscillatory, representation of shocks and contact discontinuities; small errors and high-order accuracy in smooth regions of the flow; and convergence towards the correct entropy solution as the mesh is refined. This can be readily tested in $1D$ due to the availability of exact solutions to Riemann problems; in multiple space dimensions, we instead rely upon more qualitative comparisons with the existing literature, as well as experimental data when available. Of central importance is the requirement that our simulations accurately reproduce the dynamics of highly singular phenomena relevant in practical applications; examples include strong blast waves, unstable contacts, self-similar implosions, and shock reflection.

2. Another key design goal is **efficiency**; specifically, we want our simulations to run in real-time with serial implementations on standard computer hardware. Our algorithms should also be *scalable*; that is, parallelizable and of optimal complexity. Algorithmic efficiency is benchmarked against comparable state-of-the-art methods.

3. An important consideration when developing an efficient simulation framework is the **simplicity** of the code. For instance, we avoid time-dependent data structures and external routines. More generally, we develop a code with a "hierarchical" structure in which well-tested portions of the code are re-used and "bootstrapped" to form the full algorithm. From the mathematical point of view, we would like our formulations to have simple interpretations (often of a geometric nature) with physical justifications.

4. To keep our code simple, we will generally work in $2D$ rectangular geometry using finite-difference methods; on the other hand, it is essential that the algorithms we develop are **flexible** and generalizable to more complex physics, geometries, and numerical discretization strategies.

To summarize, the goal of our framework is to provide a set of SAFE (simple, accurate, flexible, and efficient) numerical algorithms for capturing multiscale flows.

1.2.2. PDE-based modifications. To construct an algorithm satisfying the requirements described above, we introduce a set of PDE-based modifications to the underlying compressible Euler system (1). As a result, the methodology is independent of problem-specific geometry, numerical discretization strategy, and often even the particular physics under consideration.

Specifically, we develop a hybrid Eulerian-Lagrangian scheme set in a unified Arbitrary Lagrangian-Eulerian (ALE) framework; a schematic is provided in Figure 3.

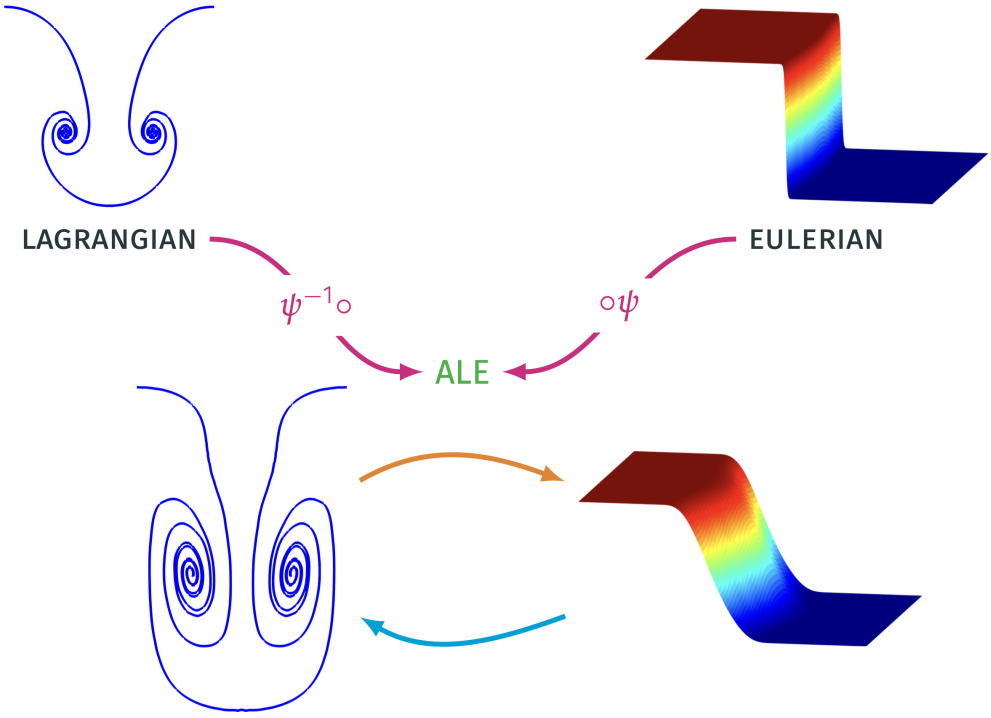


Figure 3: Front-capturing and front-tracking with adaptivity: a schematic of the numerical framework. The Eulerian (or front-capturing) part of the scheme is the C-method; the Lagrangian (or front-tracking) part of the scheme is the z-model, which is coupled to the C-method via a multiscale flow decomposition. Adaptivity is incorporated via a Smooth Adaptive Meshing algorithm, which allows us to recast the coupled Lagrangian-Eulerian scheme in Arbitrary Lagrangian-Eulerian coordinates.

- The Eulerian part of the scheme is a space-time smooth nonlinear artificial viscosity method for capturing shocks and contacts, which we refer to as the **C-method** [26, 27].

- The Lagrangian part of the scheme is based on a multiscale compressible-incompressible decomposition, together with an asymptotic model for incompressible vortex sheet evolution; this part of the algorithm is referred to as the **multiscale decomposition & z-model** for contacts [24].

- Unifying the two is a **smooth adaptive meshing (SAM)** algorithm [25] that allows us to recast our PDE-based hybrid Lagrangian-Eulerian scheme in ALE coordinates via the chain rule.

In the following three sections, we briefly describe each of these three components.

2. Space-time smooth nonlinear artificial viscosity: the C-method

2.1. Smooth front-tracking and stabilization

The central feature of the C-method is the construction of so-called C-functions for smoothly tracking the location and geometry of propagating fronts. These C-functions are found as solutions to auxiliary scalar reaction-diffusion equations of the form

$$\partial_t C + \frac{1}{\varepsilon} C - \kappa \Delta C = F \,,$$

where F is a user-defined forcing function, and ε and κ are parameters controlling the localization and smoothness of C, respectively. A suitable choice of F then forces the associated C-function to track specific fronts that require explicit stabilization.

The C-functions allow us to implement directionally isotropic and anisotropic artificial viscosity schemes for the stabilization of shocks and contacts, respectively. Based on the original idea of von Neumann & Richtmyer [33], we add to the right-hand side of the momentum equation (1b) artificial viscosity terms of the form

$$\begin{aligned} &\beta\, \partial_i(\rho\, C\, \partial_i u), && \text{for } \textit{isotropic} \text{ stabilization of shocks;} \\ &\mu\, \partial_i(\rho\, \tau^i \tau^j\, \partial_j u), && \text{for } \textit{anisotropic} \text{ stabilization of contacts}\,. \end{aligned}$$

Here, C is a smooth shock-tracking function; τ is a vector-valued function tracking the geometry of the contact; and β and μ are user-defined parameters.

We emphasize here the importance of using *anisotropic* artificial viscosity for contact discontinuities: on the one hand, the propagation of an unstable contact in directions not aligned with the underlying grid produces spurious small-scale Kelvin-Helmholtz (KH) structures that quickly

corrupt the numerical solution; on the other hand, isotropic stabilization (as in the classical artificial viscosity method) produces a *smeared* density profile with incorrect bubble/spike growth rates and prevents the development of the (physical) roll-up structure.Our anisotropic C-method smooths the contact only in directions tangential to the evolving interface, thereby preventing the growth of spurious small-scale KH structures, while preserving the classical "mushroom" roll-up (see Figure 4).

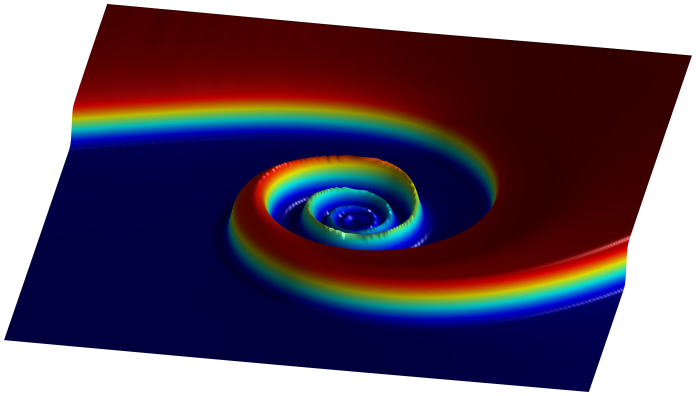


Figure 4: Roll-up of the contact in the RT instability. Video available at [23].

2.2. Numerical implementation and applications

Our numerical implementation of the C-method for the Euler system is based on a simplified 5th order WENO-type solver (with no characteristic decompositions or Riemann solvers) for advection, central differencing for the artificial viscosity terms, and 3rd order Runge-Kutta temporal integration. Meanwhile, the C-equations are efficiently solved using the classical alternating direction implicit scheme. Extensive error analysis and convergence tests show that the resulting WENO-C scheme is high-order accurate and produces smaller errors and better convergence rates than both the classical WENO scheme, as well as the classical artificial viscosity scheme of von Neumann & Richtmyer.

The C-method allows us to effectively treat the singular phenomenon of shock-wall collision and bounce-back. To further dampen the high-frequency oscillations (or *noise*) that occurs during shock-collision, we also develop a wavelet-based noise-detection scheme and an associated noise-removal procedure based on a localized heat equation solver. Shown in Figure 5 is the C-method applied to a problem involving an initially radially symmetric shock colliding with the walls of a square shock chamber, with numerous shock-wall, shock-shock, and shock-contact collisions occurring thereafter;

the C-method produces oscillation-free numerical solutions and is able to capture the intricate small-scale structure produced by the various wave interactions. Furthermore, we show that the C-method is able to simulate extremely challenging shock problems, such as the LeBlanc problem and the Noh implosion; to the best of our knowledge, no other numerical scheme is able to predict the correct shock speed and location for these two problems while preventing anomalous "wall-heating".

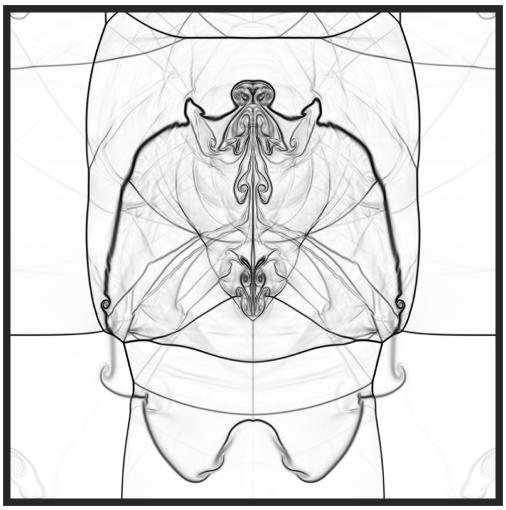


Figure 5: The C-method applied to a shock reflection problem. Video available at [23].

3. Asymptotic models for fluid instabilities: the z-model and multiscale algorithm

It is extremely difficult to capture the small-scale geometry associated with unstable Rayleigh-Taylor (RT) and Richtmyer-Meshkov (RM) interfaces using a traditional Eulerian grid. Instead, we use a Lagrangian method in which the contact discontinuity is described by a parametrized co-dimension one hypersurface. As described below, the Lagrangian method is then coupled to the Eulerian C-method to form an efficient hybrid Eulerian-Lagrangian scheme.

3.1. The z-model for incompressible & irrotational dynamics

We begin with a description of our method for simulating vortex sheets in *incompressible & irrotational flows*; we will justify this simplification of the dynamics in the next subsection. Specifically, we assume that the vorticity is a measure supported on the interface and rewrite the $2D$ Euler system using the Birkhoff-Rott (BR) formulation of vortex sheet evolution, which reduces the problem

to $1D$ evolution of the parametrization $z(\alpha,t) = (z^1, z^2)$ and the amplitude of vorticity $\varpi(\alpha,t)$.

The dimensional reduction provided by the BR formulation allows for high-resolution representation of the contact; on the other hand, it is well-known that the nonlinear and non-local ϖ equation is difficult to simulate efficiently and accurately. As such, we instead develop a reduced-order asymptotic model for the BR system, which we refer to as the z-model. In contrast to traditional reduced-order models for BR dynamics (that rely upon smallness of slope or curvature), our z-model is instead derived using small *non-locality* in the asymptotic parameter.

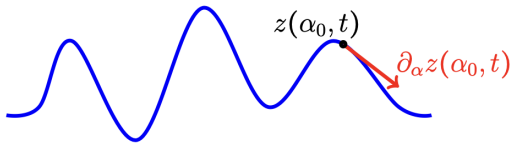


Figure 6: The parametrized curve in the z-model.

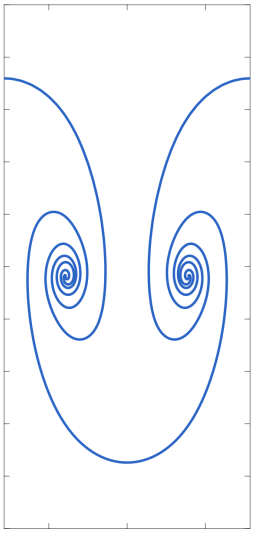
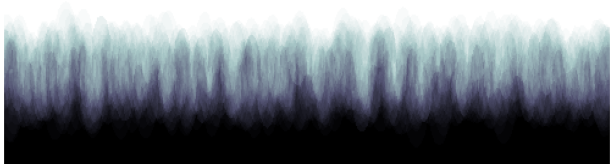


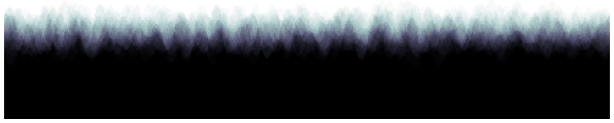
Figure 7: z-model simulation of the RT instability.

The z-model allows for interface turn-over and roll-up, and yields a significant simplification for the ϖ equation. Our numerical implementation of the z-model is based upon a hybrid scheme: the ϖ equation is efficiently solved in Fourier space with a stabilized spectral method, while the z equations are solved in physical space using the classical vortex-blob method [14]. The z-model is generalized to $3D$ flows in [21].

Our z-model is validated via comparison with experimental data for both single-mode and multi-mode RT problems. In particular, the z-model solutions are shown to predict the correct growth rates for the bubble/spike amplitudes and mixing layer, while efficiently capturing the small-scale vortical structures of the flow. The z-model solutions are also shown to be in excellent agreement with "reference" solutions calculated using a numerical method for the full BR dynamics, while running approximately 75 times faster. We perform a number of convergence tests and demonstrate the convergence of "large-scale" solution structures (such as bubble/spike location and radius of the roll-up region) as the interfacial mesh is refined. Moreover, we show how the z-model can be used to simulate diffusion and mixing layer growth [28, 34] via ensemble averaging of randomly initialized z-model runs (see Figure 8).



(c) $t = 0.15$



(b) $t = 0.10$

(a) $t = 0.05$

Figure 8: Mixing layer growth via ensemble averaging of randomly initialized z-model runs; compare with the experimental studies of Read [28] & Youngs [34]. Video available at [23].

3.2. Multiscale decomposition for compressible RT and RM instabilities

For flows with bulk compression and vorticity, we propose a multiscale compressible-incompressible decomposition of the velocity field $u = v+w$, which is, in turn, based upon a two-phase elliptic system of Hodge type [5]. The incompressible component w is also irrotational, contains the small-scale structures of the flows, and has a discontinuity in its tangential component across the interface, but can be efficiently computed using the z-model; meanwhile, the compressible component v is forced by bulk compression and vorticity, but remains smooth near discontinuities and can thus be computed on relatively coarse grids while receiving sub-grid scale information from w. By analogy with turbulence models, such as LES and RANS, our multiscale decomposition involves a coarse-scale component v (the mean flow) and a fine-grid component w (the sub-grid scale fluctuations).

Our motivation for this type of multiscale decomposition comes from Mach number studies of single-mode RT problems [29] which showed that low-Mach flows exhibited more roll-up and small-scale structure. In addition, the low-Mach number limit is a well-known problem for traditional compressible solvers; the fast acoustic waves impose stringent time-step restrictions on explicit time-stepping schemes, and the piston-like motion at the unstable interface continually generates spurious

high-frequency noise. Our multiscale decomposition provides a novel mechanism for avoiding these issues with low-Mach flows.

Our numerical implementation of the multiscale decomposition is based on a simple hybrid Eulerian-Lagrangian scheme in which the small-scale flow is computed using the z-model, while the large-scale flow is computed using the C-method; appropriate source terms are added to the equations of motion to couple the two together. For RM problems, we formulate a weak baroclinic term for the z-model to account for vorticity production due to misalignment of pressure and density gradients; in addition, a version of Taylor's "frozen turbulence" hypothesis is enforced to ensure the correct balance between temporal and spatial derivatives of w.

We perform a number of numerical experiments involving RT and RM flows and demonstrate that our multiscale simulations agree both qualitatively and quantitatively with high-resolution reference simulations, while running two orders of magnitude (or more) faster.

4. Fast and smooth dynamic mesh adaptivity: the SAM algorithm

4.1. Smooth adaptive mesh redistribution

The central feature of our smooth adaptive meshing (SAM) algorithm is the explicit construction[2] of a space-time smooth diffeomorphism (or *ALE map*) $\psi(\cdot, t)$ that maps a fixed reference mesh $\mathcal{T}_{\text{ref}}$ to a moving adaptive mesh $\mathcal{T}(t)$ that "zooms in" on small-scale solution structures. The map ψ is constructed by prescribing the local cell volume of the adaptive mesh, i.e., the Jacobian determinant; this leads to the classical Monge-Ampère (MA) equation

$$\det \nabla \psi(x) = \mathsf{G} \circ \psi(x)\,, \tag{2}$$

where G is a user-constructed *target Jacobian function* that indicates regions of compression/expansion of the adaptive mesh.

Equation (2) is a nonlinear[3], non-local elliptic equation that is, in fact, ill-posed in multiple space dimensions; the single scalar equation (2) is insufficient to fully determine ψ and there are

[2] This is in contrast to Lagrangian-type or adaptive mesh refinement (AMR) methods.

[3] The nonlinearities appear not only as quadratic nonlinearities in the highest order derivatives on the left-hand side, but also in the composition $\mathsf{G} \circ \psi$, i.e. the right-hand side is itself an unknown of the problem.

infinitely many solutions. Previous studies have generally utilized variational methods to solve (2), wherein a unique solution is constructed by optimizing for certain geometric quantities (e.g. mesh displacement, skewness, etc.). However, variational techniques often lead to complicated formulations with numerical implementations requiring costly iterative techniques; as such, their applicability in $3D$ and in complex geometries has thus far been somewhat limited. Our approach is to instead optimize for efficiency.

Specifically, we propose a solver based on the deformation method of Dacorogna & Moser [7], which reduces (2) to the solution of a *linear* Poisson problem and a set of *linear* transport equations. Our numerical implementation is based on a compact 4th order accurate finite-difference scheme for the Poisson problem, and 5th order linear upwind spatial differencing and 4th order Runge-Kutta temporal integration for the transport equations. Our SAM algorithm is simple, accurate[4], flexible[5], and highly efficient, running approximately 200 times faster than a comparable state-of-the-art solver for (2).

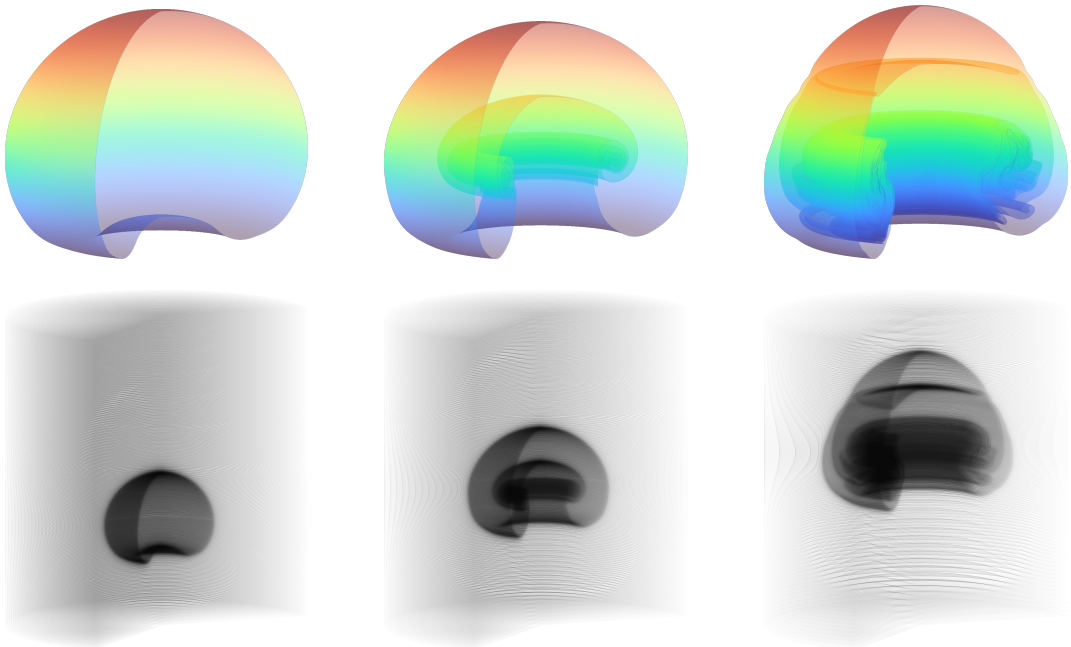


Figure 9: Three-dimensional unstable mushroom cloud and the associated adaptive mesh. Video available at [23].

[4]SAM is 4th order accurate whereas all other schemes are, to the best of our knowledge, 2nd order accurate at best.

[5]SAM is fully generalizable to unstructured meshes in complex geometries.

4.2. A perturbation reformulation for dynamic adaptivity

While our SAM algorithm as described above is highly efficient for a single mesh generation call, we encounter two problems with an implementation in practical applications: first, its use at every time-step of a dynamic fluids simulation is expensive; second, generating meshes with large zoom-in factors (necessary for capturing small-scale structures in RT problems, for instance) can create non-convex or tangled cells. While these may appear to be two different problems, our solution to both is the same; specifically, we introduce a novel *perturbation formulation* of (2) in which the map ψ is constructed via composition of a sequence of near identity deformations $\delta\psi_k \approx \mathrm{id}$ of the reference mesh $\mathcal{T}_{\mathrm{ref}}$.

A chain rule computation shows that each of the maps $\delta\psi_k$ solves an equation of exactly the same form as (2); consequently, we can use exactly the same solution algorithm as described above to construct each $\delta\psi_k$, then form the full map ψ via composition. As such, the basic SAM algorithm can be "bootstrapped" up with minimal extra effort to obtain the dynamic and large zoom-in version of the algorithm. Moreover, our numerical implementation is of optimal complexity when applied to hyperbolic systems of conservation laws.

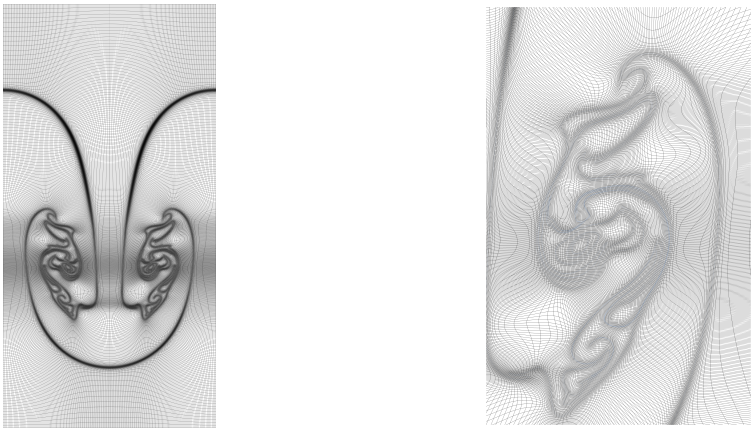
4.3. Implementation in an ALE framework

To demonstrate the efficacy of our SAM algorithm in practical applications, we formulate a simple coupled SAM-ALE for $2D$ gas dynamics. This is made possible by the explicit construction of the map ψ, which allows us to reformulate the Eulerian C-method and Lagrangian z-model[6] in a unified ALE framework via the chain rule of calculus. Thus, rather than computing Eulerian variables and Lagrangian interfaces, we instead evolve geometrically modified versions of these quantities, i.e. ALE variables and interfaces. On the other hand, the same Eulerian/Lagrangian numerical solution algorithms (with minor modifications via additional geometric terms) can be used to compute the ALE quantities; as such, the numerical implementation of our coupled SAM-ALE scheme is relatively straightforward.

We apply our SAM-ALE scheme to two challenging test problems: the infinite-strength Noh shock implosion, and the classical RT instability. For both problems, we demonstrate that coarse-

[6] For simplicity, we have implemented a simplified Lagrangian model in which the parametrized interface z is simply transported by the Eulerian velocity. Future work will formulate the full z-model dynamics in the ALE framework.

grid SAM-ALE simulations are qualitatively and quantitatively comparable to fine-grid uniform-mesh solutions, while running approximately 6-10 times faster. This type of speed-up result for the RT problem is, to the best of our knowledge, the first of its kind[7].



(a) Adaptive mesh **(b)** Zoom-in

Figure 10: SAM scheme applied to the RT instability. Video available at [23].

5. Future work

Our immediate task for the future is the complete implementation of the z-model in the ALE setting, together with a thorough study of our unified simulation framework applied to classical gas dynamics tests. Apart from this, there are several different avenues of research we are interested in pursuing in the future:

- incorporating combustion and other complex physics (such as magnetohydrodynamics, radiation transport, and plasma physics);
- implementation of level-set and semi-Lagrangian methods into our simulation framework;
- further refinement of the multiscale decomposition and development of new interface models for propagating waves in compressible gas dynamics;
- numerical study of shock formation and other types of blow-up in multiple space dimensions.

[7] In fact, most attempts at using moving-mesh adaptivity to simulate the RT instability focus on simply producing a simulation that does not crash due to mesh tangling.

References

[1] P. G. Baines and H. Mitsudera. On the mechanism of shear flow instabilities. *Journal of fluid mechanics*, 276:327–342, 1994.

[2] E. Baum, B. Peterson, B. Böhm, and A. Dreizler. On the validation of les applied to internal combustion engine flows: part 1: comprehensive experimental database. *Flow, turbulence and combustion*, 92:269–297, 2014.

[3] D. Boyer. An experimental study of the explosion generated by a pressurized sphere. *Journal of Fluid Mechanics*, 9(3):401–429, 1960.

[4] T. Buckmaster, S. Shkoller, and V. Vicol. Shock formation and vorticity creation for 3d euler. *Communications on Pure and Applied Mathematics*, 2020.

[5] C. A. Cheng and S. Shkoller. Solvability and regularity for an elliptic system prescribing the curl, divergence, and partial trace of a vector field on sobolev-class domains. *Journal of Mathematical Fluid Mechanics*, 19:375–422, 2017.

[6] D. Clark, C. Weber, J. Milovich, J. Salmonson, A. Kritcher, S. Haan, B. Hammel, D. Hinkel, O. Hurricane, O. Jones, et al. Three-dimensional simulations of low foot and high foot implosion experiments on the national ignition facility. *Physics of Plasmas*, 23(5), 2016.

[7] B. Dacorogna and J. Moser. On a partial differential equation involving the jacobian determinant. In *Annales de l'Institut Henri Poincaré C, Analyse non linéaire*, volume 7, pages 1–26. Elsevier, 1990.

[8] V. N. Gamezo, D. Desbordes, and E. S. Oran. Formation and evolution of two-dimensional cellular detonations. *Combustion and Flame*, 116(1-2):154–165, 1999.

[9] R. J. Grand, F. A. Gómez, F. Marinacci, R. Pakmor, V. Springel, D. J. Campbell, C. S. Frenk, A. Jenkins, and S. D. White. The auriga project: the properties and formation mechanisms of disc galaxies across cosmic time. *Monthly Notices of the Royal Astronomical Society*, 467(1): 179–207, 2017.

[10] W. Hogan, E. Moses, B. Warner, M. Sorem, and J. Soures. The national ignition facility. *Nuclear Fusion*, 41(5):567, 2001.

[11] S. C. Hsu and P. M. Bellan. A laboratory plasma experiment for studying magnetic dynamics of accretion discs and jets. *Monthly Notices of the Royal Astronomical Society*, 334(2):257–261, 2002.

[12] Y.-F. Jiang, J. M. Stone, and S. W. Davis. A global three-dimensional radiation magneto-hydrodynamic simulation of super-eddington accretion disks. *The Astrophysical Journal*, 796 (2):106, 2014.

[13] Y.-F. Jiang, J. M. Stone, and S. W. Davis. Super-eddington accretion disks around supermassive black holes. *The Astrophysical Journal*, 880(2):67, 2019.

[14] R. Krasny. Desingularization of periodic vortex sheet roll-up. *Journal of Computational Physics*, 65(2):292–313, 1986.

[15] C. Li, S. Zhu, Y.-l. Xu, and Y. Xiao. 2.5 d large eddy simulation of vertical axis wind turbine in consideration of high angle of attack flow. *Renewable energy*, 51:317–330, 2013.

[16] R. Linn, J. Reisner, J. J. Colman, and J. Winterkamp. Studying wildfire behavior using firetec. *International journal of wildland fire*, 11(4):233–246, 2002.

[17] M. M. Marinak, G. Kerbel, N. Gentile, O. Jones, D. Munro, S. Pollaine, T. Dittrich, and S. Haan. Three-dimensional hydra simulations of national ignition facility targets. *Physics of Plasmas*, 8(5):2275–2280, 2001.

[18] P. Moin and K. Mahesh. Direct numerical simulation: a tool in turbulence research. *Annual review of fluid mechanics*, 30(1):539–578, 1998.

[19] I. Neal, S. Shkoller, and V. Vicol. A characteristics approach to shock formation in 2d euler with azimuthal symmetry and entropy. *arXiv preprint arXiv:2302.01289*, 2023.

[20] J. Nuckolls, L. Wood, A. Thiessen, and G. Zimmerman. Laser compression of matter to super-high densities: Thermonuclear (ctr) applications. *Nature*, 239(5368):139–142, 1972.

[21] G. Pandya and S. Shkoller. Interface models for three-dimensional rayleigh–taylor instability. *Journal of Fluid Mechanics*, 959:A10, 2023.

[22] F. Porté-Agel, M. Bastankhah, and S. Shamsoddin. Wind-turbine and wind-farm flows: A review. *Boundary-layer meteorology*, 174(1):1–59, 2020.

[23] R. Ramani. Research videos. https://www.math.ucdavis.edu/~rramani/Research/research.html#Research.

[24] R. Ramani and S. Shkoller. A multiscale model for rayleigh-taylor and richtmyer-meshkov instabilities. *Journal of Computational Physics*, 405:109177, 2020.

[25] R. Ramani and S. Shkoller. A fast dynamic smooth adaptive meshing scheme with applications to compressible flow. *Journal of Computational Physics*, page 112280, 2023.

[26] R. Ramani, J. Reisner, and S. Shkoller. A space-time smooth artificial viscosity method with wavelet noise indicator and shock collision scheme, Part 1: The 1-D case. *Journal of Computational Physics*, 387:81 – 116, 2019. ISSN 0021-9991.

[27] R. Ramani, J. Reisner, and S. Shkoller. A space-time smooth artificial viscosity method with wavelet noise indicator and shock collision scheme, Part 2: The 2-D case. *Journal of Computational Physics*, 387:45 – 80, 2019. ISSN 0021-9991.

[28] K. Read. Experimental investigation of turbulent mixing by Rayleigh-Taylor instability. *Physica D Nonlinear Phenomena*, 12(1-3):45–58, 1984.

[29] S. J. Reckinger, D. Livescu, and O. V. Vasilyev. Comprehensive numerical methodology for direct numerical simulations of compressible rayleigh–taylor instability. *Journal of Computational Physics*, 313:181–208, 2016.

[30] C. Thomas, E. Campbell, K. Baker, D. Casey, M. Hohenberger, A. Kritcher, B. Spears, S. Khan, R. Nora, D. Woods, et al. Experiments to explore the influence of pulse shaping at the national ignition facility. *Physics of Plasmas*, 27(11), 2020.

[31] G. K. Vallis. Geophysical fluid dynamics: whence, whither and why? *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 472(2192):20160140, 2016.

[32] M. Vogelsberger, F. Marinacci, P. Torrey, and E. Puchwein. Cosmological simulations of galaxy formation. *Nature Reviews Physics*, 2(1):42–66, 2020.

[33] J. Von Neumann and R. D. Richtmyer. A Method for the Numerical Calculation of Hydrodynamic Shocks. *Journal of Applied Physics*, 21(3):232–237, Mar. 1950. doi: 10.1063/1.1699639.

[34] D. L. Youngs. Numerical simulation of turbulent mixing by Rayleigh-Taylor instability. *Physica D: Nonlinear Phenomena*, 12(1-3):32–44, 1984.

CHAPTER 1

The C-method

This chapter was published in:

- Journal of Computational Physics, Vol 387, R. Ramani, J. Reisner, & S. Shkoller, A space-time smooth artificial viscosity method with wavelet noise indicator and shock collision scheme, Part 1: The 1-D case, 81-116, Copyright Elsevier (2019); and
- Journal of Computational Physics, Vol 387, R. Ramani, J. Reisner, & S. Shkoller, A space-time smooth artificial viscosity method with wavelet noise indicator and shock collision scheme, Part 2: The 2-D case, 45-80, Copyright Elsevier (2019).

Journal of Computational Physics 387 (2019) 81–116

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

A space-time smooth artificial viscosity method with wavelet noise indicator and shock collision scheme, Part 1: The 1-*D* case

Raaghav Ramani [a], Jon Reisner [b], Steve Shkoller [a,*]

[a] *Department of Mathematics, University of California, Davis, CA 95616, USA*
[b] *Los Alamos National Lab XCP-4 MSF605, Los Alamos, NM 87544, USA*

ARTICLE INFO

Article history:
Received 22 June 2018
Received in revised form 26 February 2019
Accepted 28 February 2019
Available online 7 March 2019

Keywords:
Gas dynamics
Shocks and contacts
Euler equations
Artificial viscosity
Shock collision
Noise indicator

ABSTRACT

In this first part of two papers, we extend the *C*-method developed in [40] for adding localized, space-time smooth artificial viscosity to nonlinear systems of conservation laws that propagate shock waves, rarefaction waves, and contact discontinuities in one space dimension. For gas dynamics, the *C*-method couples the Euler equations to a scalar reaction-diffusion equation, whose solution *C* serves as a space-time smooth artificial viscosity indicator.

The purpose of this paper is the development of a high-order numerical algorithm for shock-wall collision and bounce-back. Specifically, we generalize the original *C*-method by adding a new collision indicator, which naturally activates during shock-wall collision. Additionally, we implement a new high-frequency wavelet-based noise detector together with an efficient and localized noise removal algorithm. To test the methodology, we use a highly simplified WENO-based discretization scheme. We show that our scheme improves the order of accuracy of our WENO algorithm, handles extremely strong discontinuities (ranging up to nine orders of magnitude), allows for shock collision and bounce back, and removes high frequency noise. The causes of the well-known "wall heating" phenomenon are discussed, and we demonstrate that this particular pathology can be effectively treated in the framework of the *C*-method. This method is generalized to two space dimensions in the second part of this work [41].

1. Introduction

This is the first in a two-part series of papers, in which we develop a high-order numerical algorithm to simulate compressible fluid flow with shock waves and contact discontinuities, as well as shock-wall collision and bounce-back. In the second part of this series [41], we treat problems in two space dimensions. In this first part, we begin the development for one-dimensional flows.

The initial-value problem for a nonlinear system of conservation laws in one space dimension is given as

$$\partial_t \boldsymbol{u}(x,t) + \partial_x F(\boldsymbol{u}(x,t)) = 0, \tag{1a}$$

* Corresponding author.
E-mail addresses: rramani@math.ucdavis.edu (R. Ramani), reisner@lanl.gov (J. Reisner), shkoller@math.ucdavis.edu (S. Shkoller).

https://doi.org/10.1016/j.jcp.2019.02.049
0021-9991/

$$\boldsymbol{u}(x, t = 0) = \boldsymbol{u}_0(x), \tag{1b}$$

where $\boldsymbol{u}(x, t)$ denotes a vector of conserved quantities, x denotes the space coordinate, and t denotes the time coordinate. Many different physical phenomena can be modeled by (1), including gas dynamics, described by the compressible Euler equations, which shall be the focus of this paper.

It is well known that solutions of (1) can develop finite-time discontinuities, even for smooth initial data $\boldsymbol{u}_0$. In this case, the discontinuities are propagated according to the *Rankine-Hugoniot* conditions (see §2.2). Consequently, it is important to develop robust numerical schemes that can approximate discontinuous solutions. This is a nontrivial task, since approximations to discontinuous solutions usually result in the occurrence of small-scale oscillations, or Gibbs-phenomenon; however, a variety of high-order discretization schemes and techniques have been developed to combat this issue and produce non-oscillatory solutions. In the case of 1-D gas dynamics, the construction of non-oscillatory, higher-order, numerical algorithms such as ENO by Harten, Engquist, Osher & Chakravarthy [16] and Shu & Osher [45], [46]; WENO by Liu, Osher, & Chan [29] and Jiang & Shu [19]; MUSCL by Van Leer [23], Colella [7], and Huynh [17]; or PPM by Colella & Woodward [9] requires carefully chosen *reconstruction* and *numerical flux*.

Such numerical methods evolve cell-averaged quantities; to calculate an accurate approximation of the flux at cell-interfaces, these schemes reconstruct kth-order ($k \geq 2$) polynomial approximations of the solution (and hence the flux) from the computed cell-averages, and thus provide kth-order accuracy away from discontinuities. See, for example, the convergence plots of Greenough & Rider [15] and Liska & Wendroff [25]. Given a polynomial representation of the solution, a strategy is chosen to compute the most accurate cell-interface flux, and this is achieved by a variety of algorithms. Centered numerical fluxes, such as Lax-Friedrichs, add dissipation as a mechanism to preserve stability and monotonicity. On the other hand, *characteristic-type* upwinding based upon exact (Godunov) or approximate (Roe, Osher, HLL, HLLC) Riemann solvers, which preserve monotonicity without adding too much dissipation, tend to be rather complex and PDE-specific; moreover, for strong shocks, other techniques may be required to dampen post-shock oscillations or to yield entropy-satisfying approximations (see Quirk [39]). Again, we refer the reader to the papers [15], [25] or Colella & Woodward [8] for a thorough overview, as well as a comparison of the effectiveness of a variety of competitive schemes.

Majda & Osher [22] have shown that *any* numerical scheme for a problem with discontinuities will suffer from a formal loss of accuracy near the discontinuity. Nonetheless, the use of high-order schemes is imperative for the resolution of finer structures in smooth regions of the flow. Formally high-order WENO schemes (as well as other high-order methods) maintain high-order accuracy in regions away from shocks, but are only first-order accurate at the discontinuity.

In order to ascertain the performance of a method, it is essential to conduct numerical tests for a range of problems with different features of varying complexity. These tests are made precise by calculating error norms of the computed solution relative to either an exact solution (if available), or a highly resolved solution which may be regarded as the exact solution. Proposed numerical algorithms should demonstrate small error norms and close to optimal convergence for a range of test problems. However, due to the fact that different tests can exhibit very different phenomena and features, it is not so surprising that there are a number of situations in which anomalous behavior of solutions is observed, which results in large errors and poor rates of convergence. Examples of such errors are wall-heating, the carbuncle phenomenon, long wavelength instabilities in slow-moving shocks, and non entropy-satisfying "expansion shocks" (see Quirk [39] for further details).

In this paper, we continue the development of the C-method [40], a nonlinear artificial viscosity modification of the Euler equations of gas dynamics, whose numerical discretization by a simple WENO-type (or even central differencing) scheme can stabilize the type of instabilities noted above. As proven in [40], weak solutions of the C-method modification of the Euler equations converge to the unique (entropy) solutions of the Euler equations as the artificial viscosity parameter tends to zero. Herein, we present numerical error analysis and order of accuracy studies for a number of classical shock tube experiments; we show that a highly simplified WENO discretization of the C-method yields highly accurate solutions displaying close to optimal rates of convergence.

For instance, we show that for the Sod problem, our simple WENO-type discretization of the C-method yields smaller errors and faster rates of convergence in the L^1, L^2, and L^∞ norms as compared to the same WENO discretization of the unmodified Euler equations. In particular, for the difficult problem of shock-wall collision (to be introduced in §1.2 and developed in §3) for the Sod problem on a grid with 801 cells, we show that the L^1 error with the C-method is 35% of the error without the C-method. Moreover, the order of convergence of solutions is approximately 0.95, which is close to optimal and more than twice the order of convergence when the C-method is not employed. Similar conclusions hold also for the extremely difficult LeBlanc problem, for which we show that the use of the C-method produces L^1 errors that are approximately four times smaller prior to shock-wall collision, and approximately three times smaller post shock-wall collision.

Our quantitative analysis, together with the qualitative observations we make via plot comparison, lead us to conclude that the use of a simple discretization of the C-method provides a flexible, highly accurate scheme that produces solutions with close to optimal rates of convergence for a variety of problems with different features.

1.1. Using artificial viscosity with conservation laws

Artificial viscosity is an effective method for the numerical stabilization of shock waves in gas dynamics; the simplest such regularization of (1) replaces the right-hand side with the linear second-order operator (see, for instance, [20,50,11])

$$\beta\,\Delta x\,\partial_{xx}\boldsymbol{u}(x,t)\,, \tag{2}$$

where $\beta = O(1)$ is a constant, and Δx denotes a small asymptotic parameter that, when the term (2) is numerically discretized, represents the grid spacing.

For each such $\beta > 0$, solutions to the regularized conservation law smooth the shock across a small region of width proportional to Δx, and simultaneously prevent small-scale oscillations from corrupting sound waves in numerical simulations; nevertheless, the uniform application of diffusion given by (2) ensures only first-order accuracy of numerical schemes and overly diffuses wave amplitudes and speeds.

In [49], Von Neumann and Richtmeyer replaced the uniform linear viscosity (2) with a nonlinear term given by

$$\beta\,(\Delta x)^2\partial_x\left(|\partial_x u|\,\partial_x\boldsymbol{u}\right)\,, \tag{3}$$

which we shall refer to as *classical artificial viscosity*. Here, $u(x,t)$ represents, in the case of the Euler equations of gas dynamics, the velocity of the fluid. The use of the localizing coefficient $|\partial_x u|$ in (3) concentrates the addition of viscosity to the narrow intervals containing shocks, while maintaining high-order accuracy in regions away from the shock, wherein the solution is smooth. See also Margolin [31] and Mattsson & Rider [32] for a description of the origin and the interpretation of artificial viscosity as a physical phenomenon.

It is now well-known [21,14] that classical artificial viscosity corrects for the over-dissipation of the linear viscosity (2), and allows for the implementation of numerical methods that are both non-oscillatory at shocks, as well as high-order accurate in smooth regions. On the other hand, the fact that the localizing coefficient $|\partial_x u|$ itself becomes highly irregular in regions containing shocks often results in the failure of such schemes to suppress spurious oscillations. This inadequacy of classical artificial viscosity may be observed with the highly singular phenomenon of shock-wave wall collision. In this case, large amplitude, high frequency, non-physical oscillations appear in the solution post-collision behind the shock curve, and the rough nature of $|\partial_x u|$ in both space and time means that the classical artificial viscosity is often unable to remove such oscillations.

This suggests that a space-time smoothed variant of the localizing coefficient $|\partial_x u|$ might allow for a less oscillatory, more accurate solution profile. We propose the use of the C-method as a means of producing such a localizing coefficient. A similar method is employed by Cabot & Cook [3,4], who use a high-wavenumber indicator together with a Gaussian filter to produce such a function, though we note that the produced function is only spatially regularized, and not temporally. See also the work of Barter & Darmofal [2], who utilize a PDE-based approach to smooth the localizing function. As we shall explain below, the function $C(x,t)$ will play the role of $|\partial_x u|$; not only will it be a space-time smooth approximation, but it will moreover be an envelope for $|\partial_x u|$, maintaining its highly localized properties while retaining a certain memory of the behavior of the shock wave.

1.2. Stabilizing shock collision

In the context of fixed-grid, explicit, finite-difference schemes, shock-wall collision and bounce-back leads to egregious oscillatory behavior. This is primarily due to the fact that the shock-wall collision causes an immediate change in the sign of the shock speed $\dot{\sigma}(t)$, leading to a discontinuity in $\dot{\sigma}(t)$. Consequently, shock-wall collision is a highly singular phenomenon that requires explicit stabilization. In §3, we introduce a simple modification of the C-method, which we call the wall C-method, that implements a space-time smooth stabilization for shock-wall collision. This method is then applied to various test cases in §5, with the computed solutions showing excellent agreement with the exact solution post shock-wall collision. Error analysis and convergence tests show that the wall C-method produces solutions with much smaller errors, even for the difficult LeBlanc shock tube problem.

1.3. High-frequency noise

The occurrence of high-frequency, often small amplitude, spurious oscillations (or *noise*) is a common issue in numerical schemes. One cause of this noise is related to the stability (CFL) condition for explicit time-integration methods. A simple method for suppressing such noise is the use of the linear viscosity (2), though, as explained above, this often results in the degradation of the solution in regions without noise. An alternative is to first decompose the solution using a basis of orthogonal *wavelets*, then truncate the decomposition so as to remove the high frequency components (which correspond with noise), though this may be very computationally expensive and, moreover, requires the use of a fully orthogonal basis of wavelets. In §4, we introduce a hybridized version of the two above methods, wherein wavelets are used to accurately locate high frequency noise, and then a linear viscosity is used, via a localized heat equation solver, to remove this noise. This noise detection and removal algorithm is very simple to implement, and is applied to a number of test problems in §5. Error analysis shows that the algorithm improves the accuracy of the solution while retaining the order of convergence; in particular, the algorithm is able to suppress high-frequency noise while preserving the amplitude of lower frequency (physical) sinusoidal waves for the Osher-Shu problem.

1.4. Outline of the paper

In §2.1, we introduce the compressible Euler equations, the corresponding flux, and the Rankine-Hugoniot jump conditions. In §2.2, we review the original C-method, as introduced in [40]. Then in §3, we discuss the problem of shock-wave wall collision, and introduce a novel generalization of the C-method, which relies on a new artificial *wall viscosity* mechanism that suppresses post shock-collision oscillations. We then introduce our WENO-C-W scheme as a discretized version of our new C-method for shock-wall collision. In §4, we present a wavelet based *noise indicator*, that locates regions of noise containing high frequency oscillations on the discretized domain. A noise removal algorithm, based on a localized solution of the heat equation, is then used to remove high frequency oscillations. We then describe our WENO-C-W-N algorithm which adds the noise indicator and noise removal scheme to our WENO-C-W method. Finally, in §5, we demonstrate the efficacy of our method for a number of classical shock tube problems, including the Sod, LeBlanc, Peak, and Osher-Shu tests. We show numerical results and order-of-accuracy studies, and in the process, we explain the cause and solution to the wall-heating problem. In Appendix A, we describe two WENO schemes that we use for comparison purposes: the first couples WENO with classical artificial viscosity, while the second couples WENO with Noh's artificial viscosity operator, designed specifically for the case of shock-wall collision and the wall-heating phenomenon.

2. The compressible Euler equations and the original C-method

2.1. The conservation laws of gas dynamics

The compressible Euler equations on a 1-D spatial interval $x_1 \le x \le x_M$, and a time interval $0 \le t \le T$ are written in vector-form as the following coupled system of nonlinear conservation laws:

$$\partial_t \boldsymbol{u}(x,t) + \partial_x \boldsymbol{F}(\boldsymbol{u}(x,t)) = \boldsymbol{0}, \quad x_1 < x < x_M\,, t > 0, \tag{4a}$$

$$\boldsymbol{u}(x,0) = \boldsymbol{u}_0(x), \quad x_1 \le x \le x_M\,, t = 0, \tag{4b}$$

where the 3-vector $\boldsymbol{u}(x,t)$ and *flux function* $\boldsymbol{F}(\boldsymbol{u}(x,t))$ are defined, respectively, as

$$\boldsymbol{u} = \begin{pmatrix} \rho \\ \rho u \\ E \end{pmatrix} \quad \text{and} \quad \boldsymbol{F}(\boldsymbol{u}) = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ u\,(E+p) \end{pmatrix},$$

while the given initial data for the problem is

$$\boldsymbol{u}_0(x) = \begin{pmatrix} \rho_0(x) \\ (\rho u)_0(x) \\ E_0(x) \end{pmatrix}.$$

The *conservative variables* ρ, ρu, and E denote the *density*, *momentum*, and *energy* of a compressible gas, while the variable u represents the *velocity field*. The variable p denotes the *pressure* function, and according to the ideal gas law is given by

$$p = (\gamma - 1)\left(E - \frac{1}{2}\rho u^2\right), \tag{5}$$

where γ is the adiabatic constant. Equations (4) represent the conservation of mass, linear momentum, and energy in the evolution of a compressible gas.

The total energy per unit volume E is the sum of kinetic energy and the potential energy,

$$E = \underbrace{\frac{1}{2}\rho u^2}_{\text{kinetic}} + \underbrace{\frac{p}{\gamma - 1}}_{\text{potential}}. \tag{6}$$

We also define the *specific internal energy per unit mass* of the system e, defined as

$$e = \frac{p}{(\gamma - 1)\rho}, \tag{7}$$

so that the total energy of the system may be written as the sum of the kinetic energy and the internal energy per unit volume ρe,

$$E = \frac{1}{2}\rho u^2 + \rho e\,.$$

The gradient of the flux $\boldsymbol{F}(\boldsymbol{u})$ is given by

$$\mathrm{D}\boldsymbol{F}(\boldsymbol{u}) = \begin{bmatrix} 0 & 1 & 0 \\ \frac{1}{2}(\gamma-3)u^2 & (3-\gamma)u & \gamma-1 \\ -\gamma\frac{uE}{\rho} + (\gamma-1)u^3 & \frac{\gamma E}{\rho} + \frac{3}{2}(1-\gamma)u^2 & \gamma u \end{bmatrix}$$

with eigenvalues

$$\lambda_1 = u + c\,,\ \lambda_2 = u\,,\ \lambda_3 = u - c\,,$$

where $c = \sqrt{\gamma p/\rho}$ denotes the sound speed (see, for example, Toro [48]). These eigenvalues determine the wave speeds. Since the behavior of the various wave patterns is greatly influenced by the speed of propagation, we define the *maximum wave speed* $S(\boldsymbol{u})$ as

$$S(\boldsymbol{u}) = [S(\boldsymbol{u})](t) = \max_{i=1,2,3}\ \max_{x}\{|\lambda_i(x,t)|\}\,. \tag{9}$$

We are interested in solutions $\boldsymbol{u}$ with discontinuous wave profiles, such as those with shock waves and contact discontinuities. The Rankine-Hugoniot (R-H) conditions determine the speed $\dot\sigma = \dot\sigma(t)$ of the moving shock or contact discontinuity, and represent conservation of mass, linear momentum and energy across the discontinuity (see, for example, [24]). For a shock wave, the R-H condition is given by the relation

$$F(\boldsymbol{u}_l) - F(\boldsymbol{u}_r) = \dot\sigma\,(\boldsymbol{u}_l - \boldsymbol{u}_r)$$

where the subscript l denotes the state to the left of the discontinuity, and the subscript r denotes the state to the right of the discontinuity. This means that the following three *jump conditions* must hold:

$$(\rho_l u_l) - (\rho_r u_r) = \dot\sigma\,(\rho_l - \rho_r) \tag{10a}$$

$$\left(\rho_l u_l^2 + p_l\right) - \left(\rho_r u_r^2 + p_r\right) = \dot\sigma\,\left((\rho u)_l - (\rho u)_r\right) \tag{10b}$$

$$(u_l(E_l + p_l)) - (u_r(E_r + p_r)) = \dot\sigma\,(E_l - E_r)\,. \tag{10c}$$

Uniqueness for weak solutions that have jump discontinuities in general does not hold, unless entropy conditions are satisfied (see the discussion in §2.9.4 in [40]). However, solutions obtained in the limit of zero viscosity are known to satisfy the entropy condition and are hence unique. We refer the reader to [40] for a discussion of the convergence of C-method solutions as $\Delta x \to 0$.

2.2. A review of the original C-method

We now briefly review the C-method from [40], which is a spacetime smooth version of classical artificial viscosity with a *compression switch*:

$$\beta(\Delta x)^2\,\partial_x\Big(\mathbb{1}_{(-\infty,0)}(\partial_x u)\,|\partial_x u|\,\partial_x u\Big)\,,$$

where the compression switch $\mathbb{1}_{(-\infty,0)}(\partial_x u)$ ensures that artificial viscosity is only activated during compression, and not in regions of expansion where there are no shocks.

The localizing function $C(x,t)$ is given as the solution to the scalar reaction-diffusion equation[1]

$$\partial_t C + \frac{S(\boldsymbol{u})}{\Delta x}C - S(\boldsymbol{u})\Delta x\,\partial_{xx}C = \frac{S(\boldsymbol{u})}{\Delta x}\,G\,,$$

where the forcing G is

$$G \equiv G(x,t) = \mathbb{1}_{(-\infty,0)}(\partial_x u)\,\frac{|\partial_x u|}{\max_x|\partial_x u|}\,, \tag{11}$$

and $S(\boldsymbol{u})$ is the maximum wave speed (9). The C-method artificial viscosity term is then given by

$$\tilde\beta(\Delta x)^2\,\partial_x\,(C\,\partial_x u)\ ,\ \text{where}\ \tilde\beta = \frac{\max_x|\partial_x u|}{\max_x C}\,\beta\,,$$

and the compressible Euler equations coupled with the C-method are written as the following Euler-C system:

[1] We note that this scalar reaction-diffusion equation is not Galilean invariant. In the current implementation, the C-method is viewed purely as a numerical tool, whereas the function C may very well be viewed as an important physical quantity, in which case the C-equation itself should be preserved under Galilean transformations. This can be accomplished by the addition of an advection term to the current C-equation. We have checked for some 1-D examples that the addition of such a term has little effect on the demonstrated success of the C-method.

$$\partial_t \rho + \partial_x(\rho u) = 0\,, \tag{12a}$$

$$\partial_t(\rho u) + \partial_x(\rho u^2 + p) = \tilde{\beta}(\Delta x)^2\, \partial_x\left(\rho C\, \partial_x u\right)\,, \tag{12b}$$

$$\partial_t E + \partial_x(u(E+p)) = \tilde{\beta}(\Delta x)^2\, \partial_x\left(\rho C\, \partial_x(E/\rho)\right)\,, \tag{12c}$$

$$\partial_t C + \frac{S(\boldsymbol{u})}{\Delta x} C - S(\boldsymbol{u})\Delta x\, \partial_{xx} C = \frac{S(\boldsymbol{u})}{\Delta x}\, G(\partial_x u)\,. \tag{12d}$$

Solutions of the Euler-C equations (12) converge to solutions of the Euler equations (4) as $\beta \to 0$ (see Section 2.9 in [40] for a proof). As was demonstrated in [40], a simple WENO-type numerical discretization of the Euler-C equations (12) (as will be described in §3) is an effective high-order scheme which compares favorably to the best state-of-the-art algorithms for the classical shock-tube experiments of Sod, Osher-Shu, Woodward-Colella, and LeBlanc. In particular, this simple WENO-type discretization of the C-method is able to remove the large overshoot in the LeBlanc contact discontinuity for the internal energy function (see [40]), whereas the other state-of-the-art schemes were not able to do so.

Herein, we generalize the C-method to allow for shock-wave wall collision and bounce-back, and introduce a wavelet-based *noise indicator* algorithm that locates high-frequency noise; a heat equation-based local solver will be used for noise removal. We shall also explain the well-known problem of wall-heating (see, for example, [43,37]).

3. A new C-method for shock-wall collision

We now consider the highly singular problem of shock-wall collision and bounce-back, and specifically, the removal of spurious post collision oscillations.

3.1. The Euler-C-W system

As a generalization to the Euler-C system (12), we consider the following coupled Euler-C-W system:

$$\partial_t \rho + \partial_x(\rho u) = 0, \tag{13a}$$

$$\partial_t(\rho u) + \partial_x(\rho u^2 + p) = \partial_x\left(\mathcal{B}^{(u)}(t)\, \rho\, C\, \partial_x u\right), \tag{13b}$$

$$\partial_t E + \partial_x(u(E+p)) = \partial_x\left(\mathcal{B}^{(E)}(t)\, \rho\, C\, \partial_x(E/\rho)\right)\,, \tag{13c}$$

$$\partial_t C + \frac{S(\boldsymbol{u})}{\varepsilon\Delta x} C - \kappa\,\Delta x \cdot S(\boldsymbol{u})\partial_{xx} C = \frac{S(\boldsymbol{u})}{\varepsilon\Delta x} G\,, \tag{13d}$$

$$\partial_t C_w + \frac{S(\boldsymbol{u})}{\varepsilon_w\Delta x} C_w - \kappa_w\,\Delta x \cdot S(\boldsymbol{u})\partial_{xx} C_w = \frac{S(\boldsymbol{u})}{\varepsilon_w\Delta x} G\,, \tag{13e}$$

where

$$\mathcal{B}^{(u)}(t) = (\Delta x)^2 \cdot \frac{\max_x |\partial_x u|}{\max_x C}\left(\beta^u + \beta^u_w \cdot \overline{C}_w(t)\right)\,, \tag{14a}$$

$$\mathcal{B}^{(E)}(t) = (\Delta x)^2 \cdot \frac{\max_x |\partial_x u|}{\max_x C}\left(\beta^E + \beta^E_w \cdot \overline{C}_w(t)\right)\,, \tag{14b}$$

and where the smooth and localized bump function $\overline{C}_w(t)$ is defined as

$$\overline{C}_w(t) = \frac{C_w(x_M, t)}{\max_x C_w(x,t)}, \tag{15}$$

and x_M denotes the right boundary, where the shock-wall collision and bounce-back is assumed (for simplicity) to occur. Furthermore, $S(\boldsymbol{u})$ is the maximum wave-speed (9), and $G = G(x,t)$ is the forcing to the C-equation, defined by (11). The indicator function $\mathbb{1}_{(-\infty,0)}(\partial_x u)$ is the *compression switch*, in which G is non-zero only if $\partial_x u < 0$. For convenience, we list all of the parameters and variables associated with the system (13) in Table 1. We note that due to the presence of the compression switch in the definition of G, we can instead define $G(x,t) = \mathbb{1}_{(-\infty,0)}(\partial_x \rho) \cdot \frac{|\partial_x \rho|}{\max_x |\partial_x \rho|}$ and obtain identical results.[2]

We shall explain the use of this new $C_w(x,t)$ function and the localized time-function $\overline{C}_w(t)$ below, when we present the results of numerical experiments of shock-wall collision.

We remark that the artificial viscosity terms on the right-hand side of the momentum equation (13b) and the energy equation (13c) ensure that the total energy is conserved; in particular, the solution $E(x,t)$ of (13c) continues to obey the identity (6). For simplicity, we consider the case of periodic boundary conditions. On the one hand, integration of the

[2] Indeed, this will be our strategy for the 2-D C-method that we introduce in [41].

Table 1
Relevant parameters and variables for the Euler-C-W system (13).

Parameter / Variable	Description
β^u, β^E	artificial viscosity coefficients for the momentum and energy, respectively.
β^u_w, β^E_w	wall viscosity coefficients for the momentum and energy, respectively.
$S(\boldsymbol{u})(t)$	maximum wave speed $\max_x(\max\{\lvert u(x,t)\rvert, \lvert u(x,t)\pm c\rvert\})$.
ε, ε_w	parameters controlling support of C and C_w, respectively.
κ, κ_w	parameters controlling smoothness of C and C_w, respectively.
$\overline{C}_w(t)$	smooth and localized bump function.

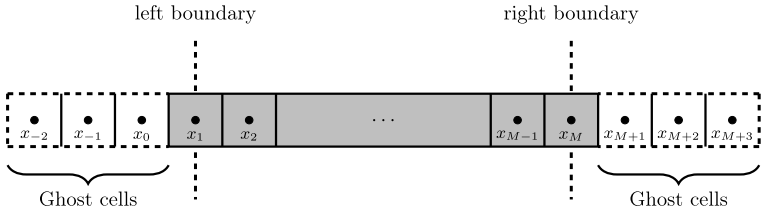


Fig. 1. The grid, together with ghost cells, for the WENO-C-W algorithm.

energy equation (13c) over the spatial domain $[x_1, x_M]$ shows that $\frac{\mathrm{d}}{\mathrm{d}t}\int_{x_1}^{x_M} E\,\mathrm{d}x = 0$. On the other hand, multiplying the momentum (13b) by u, integrating over the domain $[x_1, x_M]$, utilizing the conservation of mass (13a) together with the pressure identity (5), and the energy equation (13c), we find that

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{x_1}^{x_M}\left(\frac{1}{2}\rho u^2 + \frac{p}{\gamma-1}\right)\mathrm{d}x = 0\,.$$

This shows that the velocity u and pressure p adjust accordingly to maintain the relation (6), and that our modified Euler-C-W system conserves total energy.

3.2. Boundary conditions for the Euler-C-W system

We consider two types of boundary conditions on the interval $x_1 \le x \le x_M$. For many of the test problems, we employ the so-called *reflective* or *solid wall* boundary conditions at $x = x_1$ and $x = x_M$ and $t \ge 0$:

$$\partial_x\rho(x,t) = 0\,,\quad \rho u(x,t) = 0\,,\quad \partial_x E(x,t) = 0\,,\quad \partial_x C(x,t) = 0\,,\quad \partial_x C_w(x,t) = 0\,. \tag{16}$$

Alternatively, we shall sometimes use the *free flow* boundary conditions:

$$\partial_x\rho(x,t) = 0\,,\quad \partial_x(\rho u)(x,t) = 0\,,\quad \partial_x E(x,t) = 0\,,\quad \partial_x C(x,t) = 0\,,\quad \partial_x C_w(x,t) = 0\,. \tag{17}$$

3.3. The WENO-C-W algorithm

3.3.1. Discretization of the Euler-C-W system

We now describe the simple WENO-based space discretization scheme used for the Euler-C-W system (13). We use a formally fifth-order WENO reconstruction procedure together with upwinding, based on the sign of the velocity at the cell edges. We stress that the WENO-type discretization we use is highly simplified, and is not meant to be representative of the class of full WENO solvers. However, we note that, for certain problems, our simplified WENO-type discretization produces solutions with similar errors and convergence rates to those produced using a standard WENO scheme (see §5.2.5).

The spatial domain $x_1 \le x \le x_M$ is subdivided into M equally sized cells of width Δx, where the left-most and right-most cells are centered on the left and right boundaries, respectively (Fig. 1). We denote the cell centers by x_i for $i = 1, \ldots, M$, and the cell edges with the fractional index

$$x_{i+\frac{1}{2}} = \frac{x_i + x_{i+1}}{2},\ \text{for } i = 1, \ldots, M-1\,.$$

Any quantity evaluated at a cell center x_i shall be denoted by w_i, and a quantity evaluated at a cell edge $x_{i+\frac{1}{2}}$ is denoted by $w_{i+\frac{1}{2}}$. Given a vector w_i corresponding to cell-center values, and vectors $z_{i-\frac{1}{2}}$ and $z_{i+\frac{1}{2}}$ corresponding to cell edge values, we define the j^{th} component by

$$\left[\mathrm{WENO}(w_i, z_{i\pm\frac{1}{2}})\right]_j = \frac{1}{\Delta x}\left(\tilde{w}_{j+\frac{1}{2}} z_{j+\frac{1}{2}} - \tilde{w}_{j-\frac{1}{2}} z_{j-\frac{1}{2}}\right),$$

where the cell-edge values $\tilde{w}_{j+\frac{1}{2}}$ are calculated using a standard fifth-order WENO reconstruction procedure (see [19], [47]) with upwinding based on the sign of $z_{j+\frac{1}{2}}$.

Then, defining the vectors $\boldsymbol{u} = [\rho, \rho u, E]^T$ and $\boldsymbol{C} = [C, C_w]^T$, we now construct the operators $\mathcal{A}_{\text{WENO}}$ and $\mathcal{B}_{\text{WENO}}$ as

$$[\mathcal{A}_{\text{WENO}}(\boldsymbol{u}_i, \boldsymbol{C}_i)] = \begin{bmatrix} \left[\text{WENO}\left(\rho_i, \hat{u}_{i\pm\frac{1}{2}}\right)\right]_i \\ \left[\text{WENO}\left((\rho u)_i, \hat{u}_{i\pm\frac{1}{2}}\right)\right]_i + \tilde{\partial}_4 p_i - \mathcal{B}^{(u)}(t) \cdot \frac{\tilde{\partial_C}\left(u_{i+\frac{1}{2}}\right) - \tilde{\partial_C}\left(u_{i-\frac{1}{2}}\right)}{\Delta x} \\ \left[\text{WENO}\left((E+p)_i, \hat{u}_{i\pm\frac{1}{2}}\right)\right]_i - \mathcal{B}^{(E)}(t) \cdot \frac{\tilde{\partial_C}\left((E/\rho)_{i+\frac{1}{2}}\right) - \tilde{\partial_C}\left((E/\rho)_{i-\frac{1}{2}}\right)}{\Delta x} \end{bmatrix} \tag{18}$$

and

$$[\mathcal{B}_{\text{WENO}}(\boldsymbol{u}_i, \boldsymbol{C}_i)] = \begin{bmatrix} \frac{S(\boldsymbol{u}_i)}{\varepsilon \Delta x}\{C_i - G_i\} + \frac{\tilde{\partial_S} C_{i+\frac{1}{2}} - \tilde{\partial_S} C_{i-\frac{1}{2}}}{\Delta x} \\ \frac{S(\boldsymbol{u}_i)}{\varepsilon_w \Delta x}\left\{[C_w]_i - G_i\right\} + \frac{\tilde{\partial_S}[C_w]_{i+\frac{1}{2}} - \tilde{\partial_S}[C_w]_{i-\frac{1}{2}}}{\Delta x} \end{bmatrix}. \tag{19}$$

Here, we have used the notation $\tilde{\partial}_4 p_i$ to denote the fourth-order central difference approximation for the derivative of the pressure at the cell center x_i:

$$\tilde{\partial}_4 p_i = \frac{p_{i-2} - 8p_{i-1} + 8p_{i+1} - p_{i+2}}{12 \cdot \Delta x}.$$

The cell-edge velocities $\hat{u}_{i\pm\frac{1}{2}}$ used for upwinding are calculated using a fourth-order averaging:

$$\hat{u}_{i-\frac{1}{2}} = \frac{-u_{i-2} + 7u_{i-1} + 7u_i - u_{i+1}}{12}.$$

We have also used the notation $\tilde{\partial_C}\left(w_{i+\frac{1}{2}}\right)$ and $\tilde{\partial_S} C_{i+\frac{1}{2}}$ to denote

$$\tilde{\partial_C}\left(w_{i+\frac{1}{2}}\right) = \rho_{i+\frac{1}{2}} C_{i+\frac{1}{2}} \tilde{\partial} w_{i+\frac{1}{2}},$$
$$\tilde{\partial_S} C_{i+\frac{1}{2}} = \kappa \Delta x\, S(\boldsymbol{u}_i)\, \tilde{\partial} C_{i+\frac{1}{2}},$$

respectively. Here, the notation $z_{i+\frac{1}{2}}$ denotes a quantity calculated at the cell edge $x_{i+\frac{1}{2}}$ using the standard averaging

$$z_{i+\frac{1}{2}} = \frac{z_i + z_{i+1}}{2},$$

while the quantity $\tilde{\partial} w_{i+\frac{1}{2}}$ denotes the central difference approximation for $\partial_x w$ at the cell edge $x_{i+\frac{1}{2}}$,

$$\tilde{\partial} w_{i+\frac{1}{2}} = \frac{w_{i+1} - w_i}{\Delta x}.$$

Now, given $\boldsymbol{u}^n$ at a time $t = t_n = n\Delta t$, we evolve the solution as follows:

$$\boldsymbol{u}_i^{n+1} = \text{RK}\left(\boldsymbol{u}_i^n, \mathcal{A}_{\text{WENO}}(\boldsymbol{u}_i^n, \boldsymbol{C}_i^n)\right), \tag{20a}$$
$$\boldsymbol{C}_i^{n+1} = \text{RK}\left(\boldsymbol{C}_i^n, \mathcal{B}_{\text{WENO}}(\boldsymbol{u}_i^n, \boldsymbol{C}_i^n)\right), \tag{20b}$$

where RK denotes the explicit fourth-order Runge-Kutta time-integration method.

3.3.2. Discretization of boundary conditions and ghost node values

Boundary conditions for the functions C and C_w are imposed through the assigning of the so-called *ghost node* values. More precisely, the ghost node values for the functions C and C_w are prescribed via an even extension:

$$C_{1-k} = C_{1+k} \quad \text{and} \quad C_{M+k} = C_{M-k}, \tag{21}$$

for $k = 1, \ldots, M_g$, where M_g is the number of ghost nodes. For our (formally) fifth-order WENO scheme, $M_g = 3$.

The associated boundary conditions for the conservative variables are also imposed via the ghost node conditions. For the Dirichlet boundary condition, an odd extension is used, while for the Neumann boundary condition, an even extension

is used. More precisely, suppose that we wish to impose the free-flow boundary conditions (17). This is done by choosing the ghost node values as

$$\rho_{1-k} = \rho_{1+k} \quad \text{and} \quad \rho_{M+k} = \rho_{M-k}, \tag{22a}$$

$$\rho u_{1-k} = \rho u_{1+k} \quad \text{and} \quad \rho u_{M+k} = \rho u_{M-k}, \tag{22b}$$

$$E_{1-k} = E_{1+k} \quad \text{and} \quad E_{M+k} = E_{M-k}, \tag{22c}$$

for $k = 1, \ldots, M_g$.

The solid wall boundary conditions (16) are imposed by replacing the even extension of the momentum in (22) with the odd extension of the momentum:

$$\rho_{1-k} = \rho_{1+k} \quad \text{and} \quad \rho_{M+k} = \rho_{M-k}, \tag{23a}$$

$$\rho u_{1-k} = -\rho u_{1+k} \quad \text{and} \quad \rho u_{M+k} = -\rho u_{M-k}, \tag{23b}$$

$$E_{1-k} = E_{1+k} \quad \text{and} \quad E_{M+k} = E_{M-k}, \tag{23c}$$

for $k = 1, \ldots, M_g$. Again, it is easy to verify that the density ρ and the energy E satisfy the homogenous Neumann boundary condition in (16). To verify that the momentum satisfies the homogenous Dirichlet boundary condition, we need to use the momentum equation in the semi-discrete form (20). Suppose that at time-step n, the velocity at the boundaries vanishes: $u_M^n = u_1^n = 0$. For simplicity, we restrict to the right boundary in cell x_M. The even extensions of ρ and C and the odd extension of u mean that the diffusion term on the right-hand side of the momentum equation vanishes since

$$\begin{aligned}\tilde{\partial_C}\left(u_{M+\frac{1}{2}}\right) &= \rho_{M+\frac{1}{2}} \cdot C_{M+\frac{1}{2}} \cdot \frac{(u_{M+1} - u_M)}{\Delta x} \\ &= \rho_{M-\frac{1}{2}} \cdot C_{M-\frac{1}{2}} \cdot \frac{(-u_{M-1} + u_M)}{\Delta x} = \tilde{\partial_C}\left(u_{M-\frac{1}{2}}\right).\end{aligned}$$

Moreover, since the pressure p is evenly extended, the derivative at the boundaries $\tilde{\partial_4} p_M$ and $\tilde{\partial_4} p_1$, also vanishes. One can also check that the derivative of the flux term at the boundary vanishes: $\left[\text{WENO}\left((\rho u)_i, \hat{u}_{i\pm\frac{1}{2}}\right)\right]_M = 0$. This means that $\partial_t(\rho u) = 0$ at the boundaries, so that momentum satisfies $\rho u = 0$ at the boundaries for $t \geq 0$, provided that the initial momentum vanishes on the boundaries.

3.4. Using WENO-C-W for the Sod shock-wall collision problem

The reflection of a shock wave from a fixed wall was first considered in [10] from a theoretical viewpoint (see also [1, 36]). Further investigations in [37,12] were done primarily in the context of the wall-heating phenomenon (to be discussed below). The reflection of a shock-wave from a non-rigid boundary was considered in [33,18], wherein an artificial viscosity method was utilized to stabilize the solution.

As a motivating example, we first consider the classical Sod shock tube experiment. This is a Riemann problem on the domain $0 \leq x \leq 1$, with initial data given by

$$\begin{bmatrix} \rho_0 \\ (\rho u)_0 \\ E_0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 2.5 \end{bmatrix} \mathbb{1}_{[0,\frac{1}{2})}(x) + \begin{bmatrix} 0.125 \\ 0 \\ 0.25 \end{bmatrix} \mathbb{1}_{[\frac{1}{2},1]}(x) \text{ and } \gamma = 1.4\,, \tag{24}$$

where $\mathbb{1}_{[a,b)}(x)$ denotes the indicator function on the interval $a \leq x < b$. The solution consists of a rarefaction wave, a contact discontinuity, and a shock wave. The shock propagating to the right collides with the wall, modeled by the point $x = 1$, at time $t \approx 0.28$. In Fig. 2(a), we show the success of the WENO-C method for this problem prior to the collision of the shock wave with the wall at $x = 1$; however, as shown in Fig. 2(b), the WENO-C scheme (without the addition of the wall function $\overline{C}_w(t)$) is not sufficient to remove spurious oscillations post shock collision in the case of small $\beta = 0.5$. On the other hand, by setting $\beta = 4.0$, the velocity is mostly free of post shock-wall collision oscillations at $t = 0.36$, at the expense of an overly diffused shock profile prior to shock-wall collision at $t = 0.2$. Moreover, for more difficult problems, such as the LeBlanc problem considered in §5.4, very precise choices of the artificial viscosity parameters are required to maintain stability and correct wave speeds. Consequently, it is difficult to choose β such that the solutions both pre and post shock-wall collision are accurate and noise-free. The use of the wall viscosity will provide a nice solution strategy.

3.4.1. An explanation of the temporal bump function $\overline{C}_w(t)$

We now explain the use of the new $C_w(x,t)$ function together with the temporal bump function $\overline{C}_w(t)$. We shall assume, for simplicity, that the shock wave is traveling to the right, so that the shock wave collides with the wall $x = 1$. Thanks to the homogeneous Neumann boundary condition $\partial_x C_w = 0$ at the wall $x = 1$, there is a smooth growth (in time) of the amplitude of $C_w(1,t)$ just prior to shock-wall collision, followed by a smooth decrease of amplitude during shock bounce-back.

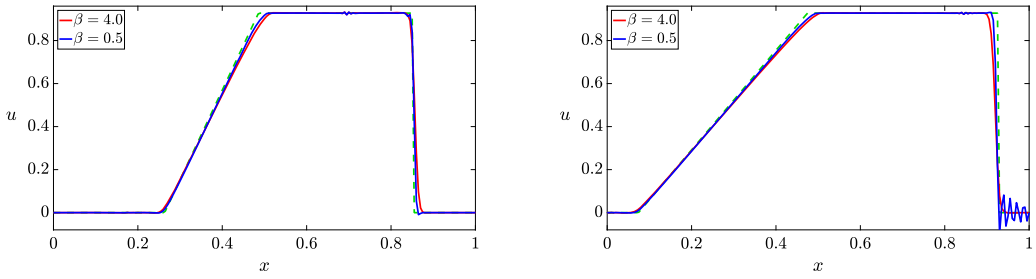


Fig. 2. The velocity profile for the Sod shock tube problem, calculated using our WENO-C scheme with 201 cells. The blue and red curves are the velocity profiles and the dashed green curve is the exact solution. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

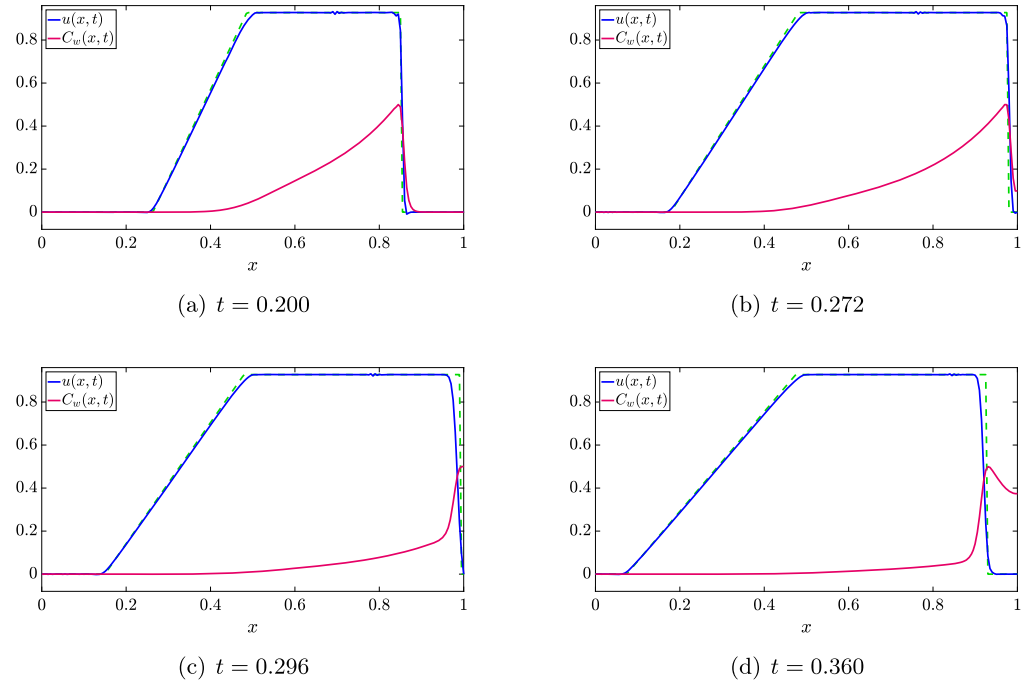


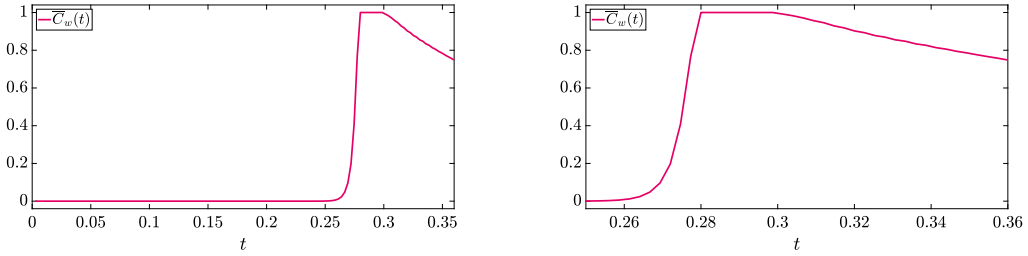
Fig. 3. The velocity profile for the Sod shock tube problem, calculated using our WENO-C-W scheme with 201 cells. The blue curve is the velocity profile and the dashed green curve is the exact solution. The red curve is the (normalized and resized) function $C_w(x,t)$.

In Fig. 3, we illustrate the WENO-C-W scheme as applied to Sod. While the shock is away from the wall, $C_w(1,t)$ is zero, and thus by formula (15) so is $\overline{C}_w(t)$; see the purple curve in Fig. 3(a). As the shock approaches the wall (as shown in Fig. 3(b)), the Neumann boundary condition for the C_w-equation ensures that $\overline{C}_w(t)$ increases smoothly, until it reaches a maximum when the shock collides with the wall (Fig. 3(c)), before smoothly decreasing back to zero as the shock moves away from the wall (Fig. 3(d)).

In Fig. 4, we plot the graph of $\overline{C}_w(t)$. The localized nature of the temporal bump function $\overline{C}_w(t)$ means that the extra viscosity, given by β_w in (14), is added only during shock-wall collision and bounce-back; prior to collision, no extra viscosity is added and the solution is consequently not overly diffused. In §5, we apply the WENO-C-W scheme to a number of different shock tube problems for shock collision and bounce-back.

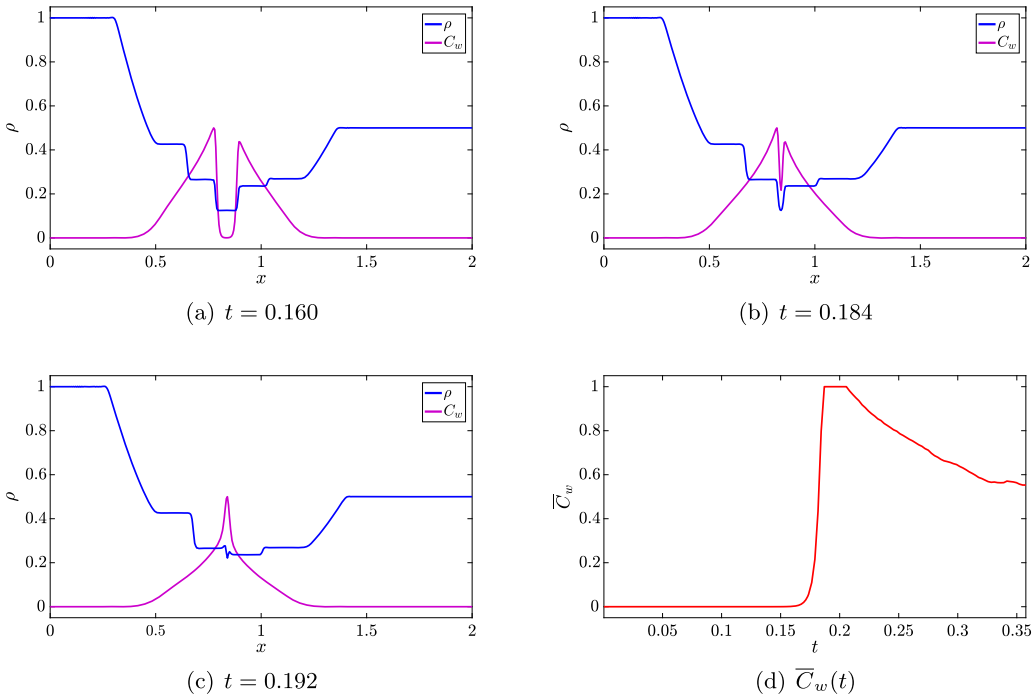
3.4.2. A generalization of our algorithm to shock-shock collision problems

We remark here that a shock hitting a wall is simply a special case of shock-shock collision; indeed, the shock-wall collision problem may be viewed as the collision between two identical shocks but with different signs for the shock speed. A simple generalization of the Euler-C-W algorithm which allows for arbitrary shock-shock collision is obtained by redefining the temporal bump function (15) with the new function



(a) $\overline{C}_w(t)$ (b) zooming in on $\overline{C}_w(t)$ during shock collision

Fig. 4. The wall indicator function $\overline{C}_w(t)$ for the Sod shock tube problem. The function is zero when the shock is away from the wall, increases smoothly as the shock approaches the wall, and reaches a maximum when the shock collides with the wall, before decreasing smoothly as the shock moves away from the wall.



(a) $t = 0.160$ (b) $t = 0.184$

(c) $t = 0.192$ (d) $\overline{C}_w(t)$

Fig. 5. The density profile for a non-identical shock-shock collision problem. The blue curve is the density profile and the purple curve in Fig. 5(a)-5(c) is the (normalized and resized) function $C_w(x,t)$. The red curve in Fig. 5(d) is the temporal bump function $\overline{C}_w(t)$.

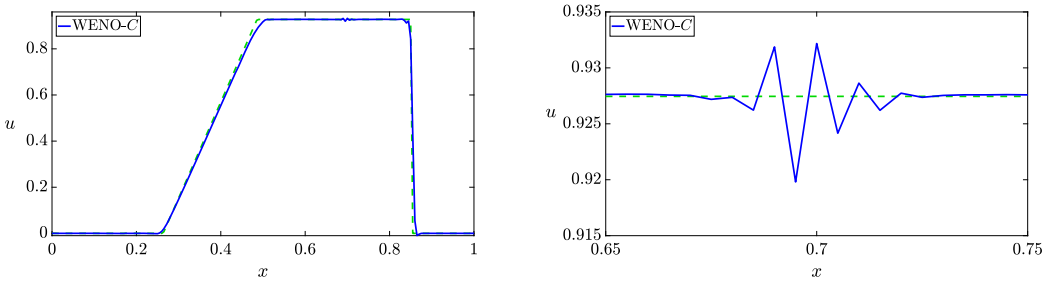
$$\overline{C}_w(t) = \sum_i \frac{C_w(x_i^*, t)}{\max_x C_w(x,t)}, \tag{25}$$

where $x_i^*(t)$ denotes the time-dependent local minima of the function $C_w(x,t)$ and approximates the location of the shock-shock collision (at the collision time). The functions $x_i^*(t)$ are analogous to the time-independent wall location x_M in the shock-wall collision problem (where the location of the collision is predetermined).

Fig. 5 shows the density function during shock-shock collision. Also shown, is the temporal bump function $\overline{C}_w$, which naturally increases as two shock waves approach one another, and provides a natural method for the addition of spacetime smooth additional artificial viscosity during the shock-shock collision process. As can be seen, the two shocks collide at $t = 0.192$, at which time the function $\overline{C}_w$ achieves its maximum value. We examine this problem of shock-shock collision in great detail in [42].

4. A wavelet-based noise indicator: the WENO-*C*-*W*-*N* method

Numerical solutions of gas dynamics often develop high-frequency noise. These (often small amplitude) spurious oscillations can occur if the time-step is too large or because of the smearing of contact discontinuities. Large time-step noise



(a) velocity profile at $t = 0.20$ (b) zooming in on the noise

Fig. 6. The velocity profile for the Sod shock tube problem, calculated using our WENO-*C* scheme with 201 cells. The blue curve is the velocity profile and the dashed green curve is the exact solution. There is noise in the region $x \in [0.65, 0.75]$. This is the location of the contact discontinuity in the density and momentum profiles.

can be seen with any explicit numerical scheme, while noise in the velocity field at the contact discontinuity is illustrated in Fig. 6 for the Sod problem. This noise is caused by the slightly different slopes that the momentum and density profiles have at the contact discontinuity.

To deal with the occurrence of spurious noise, we implement a localized *wavelet*-based noise indicator. Wavelets were first used in fluid dynamics in the analysis of turbulence by Farge [13] and Meneveau [34]. They have also been used in the numerical solution of PDE on adaptive grids (see the review paper [44]). With regard to noise detection and removal, wavelets have generally been used in the form of a nonlinear *filter*, in which a noisy function is first decomposed using wavelets, and the function is then *de-noised* by retaining only the low-frequency components. Such filtering techniques often over-smooth the noisy data, or introduce additional Gibbs-like oscillations [6].

The main novelty of our approach is the use of wavelets only for high-frequency noise detection, while noise removal is achieved by a highly localized heat equation approach.

4.1. Construction of wavelets

A wavelet is like a traditional wave (sine or cosine waves), but localized in space i.e. it has a finite support. We define a *mother wavelet* $\psi(x) = \psi_{0,0}(x)$ that represents the lowest frequency oscillation, and then use a dyadic scaling and integral translation to produce wavelets of higher frequencies:

$$\psi_{r,s}(x) = 2^{r/2}\psi(2^r x - s);\ r = 0, 1, 2, \ldots \text{ and } s = \pm 1, \pm 3, \ldots, \pm(2^r - 1).$$

Suppose that the spatial domain is given by $x_1 \le x \le x_M$. For our purposes, there are two key properties that the wavelet family $\{\psi_{r,s}\}$ needs to satisfy:

1. Zero mean:

$$\int_{x_1}^{x_M} \psi(x)\,\mathrm{d}x = 0.$$

   Note that due to the dyadic scaling and integral translation, this condition also ensures that wavelets of higher frequency have zero mean.
2. "Quasi-orthogonality" of the form:

$$\int_{x_1}^{x_M} \psi_{r,s}(x) \cdot \psi_{r,s'}(x)\,\mathrm{d}x = 0, \text{ for } r \ge 0 \text{ and } 0 \le s, s' \le 2^r - 1.$$

   That is, each wavelet is orthogonal to every other wavelet of the same frequency. This is to ensure that one can locate exactly where each frequency is active.

We define our wavelets to take the form shown in Fig. 7. Since we are only interested in the highest frequency noise, we provide the exact formula for the highest frequency wavelet as

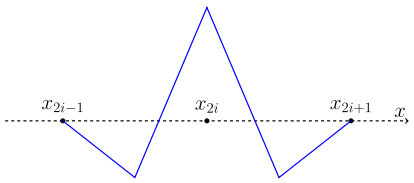


Fig. 7. The highest frequency wavelet ψ_i.

$$\psi_i(x) = \begin{cases} -\dfrac{a}{\Delta x}(x - x_{2i-1}), & \text{if} \quad x_{2i-1} \le x \le x_{2i-\frac{1}{2}} \\ +\dfrac{3a}{\Delta x}(x - x_{2i}) + a, & \text{if} \quad x_{2i-\frac{1}{2}} \le x \le x_{2i} \\ -\dfrac{3a}{\Delta x}(x - x_{2i}) + a, & \text{if} \quad x_{2i} \le x \le x_{2i+\frac{1}{2}} \\ +\dfrac{a}{\Delta x}(x - x_{2i+1}), & \text{if} \quad x_{2i+\frac{1}{2}} \le x \le x_{2i+1} \end{cases} \tag{26}$$

for each $i = 1, 2, \ldots, \frac{M-1}{2}$, where the notation $x_{k+\frac{1}{2}}$ denotes the midpoint of x_k and x_{k+1}. It is clear from formula (26) that each ψ_i is supported in the interval $\mathcal{I}_i := [x_{2i-1}, x_{2i+1}]$.

The constant $a := \sqrt{3/\Delta x}$ in (26) is a normalization factor to ensure that the wavelets have L^2 norm equal to 1. Since the highest frequency wavelets have disjoint supports, it is obvious that the quasi-orthogonality property is satisfied. One can also check that the zero mean property is satisfied.

4.2. High-frequency noise detection

Given a discretized spatial domain, the highest frequency wavelet is supported over two grid cells and is shown in Fig. 7. There are $\frac{M-1}{2}$ two-cell intervals in the computational domain. Each two-cell interval is denoted by $\mathcal{I}_j$, and there is a highest frequency wavelet $\psi_j(x)$ corresponding to each $\mathcal{I}_j$ for every $j = 1, \ldots, \frac{M-1}{2}$.

For a given function $f(x)$, we next compute the *wavelet coefficients* $\mathcal{C}_j(f)$ for this function. For each $j = 1, \ldots, \frac{M-1}{2}$,

$$\mathcal{C}_j(f) := \langle f,\, \psi_j \rangle_{L^2} = \int_{\mathcal{I}_j} f(x)\, \psi_j(x)\, \mathrm{d}x \text{ for } j = 1, \ldots, \frac{M-1}{2}\,.$$

Given the cell-center values $f(x_{2j-1})$, $f(x_{2j})$, $f(x_{2j+1})$, we can approximate the given function $f(x)$ on the interval $\mathcal{I}_j = [x_{2j-1}, x_{2j+1}]$ by a piecewise linear function $\tilde{f}(x)$; in particular, we define $\tilde{f}(x)$ by linear interpolation of the cell-center values of $f(x)$. We then approximate the wavelet coefficients by $\mathcal{C}_j(f) \approx \mathcal{C}_j(\tilde{f})$, and can compute

$$\mathcal{C}_j(\tilde{f}) = \langle \tilde{f}, \psi_j \rangle_{L^2} = -\sqrt{\frac{\Delta x}{48}} \cdot \Big[f(x_{2j+1}) - 2f(x_{2j}) + f(x_{2j-1}) \Big]. \tag{27}$$

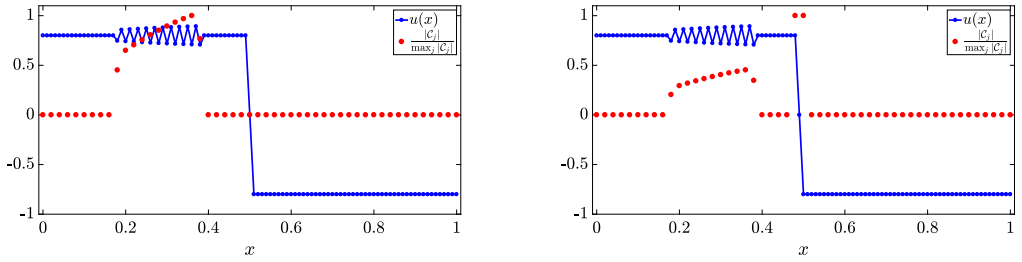
Notice that the right-hand side of (27) is proportional to the second-order central difference approximation to $f''(x_{2i})$. Also, note that if $f(x_{2i}) = \frac{1}{2}\left(f(x_{2i+1}) + f(x_{2i-1})\right)$, i.e., if the function $\tilde{f}$ is linear on $\mathcal{I}_i$, then the associated wavelet coefficient is zero. This is crucial in ensuring that only the *highest* frequency noise is detected.

The magnitude of the wavelet coefficients grows with the amplitude of the high-frequency oscillations. For example, consider the case that $f(x)$ is a hat function over the interval $\mathcal{I}_j = [x_{2j-1}, x_{2j+1}]$ and that $f(x_{2j-1j}) = f(x_{2j+1}) = 0$. Then the amplitude of the oscillation is given by the magnitude at the peak of the hat, $f(x_{2j})$, and $|\mathcal{C}_j(f)|$ is proportional to $f(x_{2j})$. Consequently, $|\mathcal{C}_j(f)|$ grows linearly with the amplitude of the oscillation.

On the other hand, suppose that we have a lower frequency oscillation, given by a hat function that spans 4 cells, say the intervals $\mathcal{I}_j$ and $\mathcal{I}_{j+1}$. In each of these intervals, the oscillating function is linear, so that the associated wavelet coefficients $\mathcal{C}_j(f)$ and $\mathcal{C}_{j+1}(f)$ are equal to zero. This illustrates the fact that the highest frequency wavelets detect *only* the highest frequency noise.

4.3. Noise detection in the presence of a shock wave

We next examine the noise detection algorithm, applied to a function $u(x)$ which has a shock discontinuity. For $j = 1, \ldots, (M-1)/2$, we again compute the wavelet coefficients $\mathcal{C}_j(u)$ for each two-cell interval $\mathcal{I}_j$ according to (27). Suppose the shock discontinuity occurs spans the two-cell interval $\mathcal{I}_j$; then, on $\mathcal{I}_j$ the shock curve is essentially linear and $C_j(u) = 0$, but if the shock is out of phase by one cell with $\mathcal{I}_j$, then the wavelet coefficient $C_j(u)$ can be large (see Fig. 8).



(a) shock is "in phase" with the wavelet (b) shock is "out of phase" with the wavelet

Fig. 8. Wavelet coefficients at the shock curve compared with wavelet coefficients in regions where there is noise. The blue curve is $u(x)$, and the red dots indicate the relative magnitude of the associated wavelet coefficient $\mathcal{C}_j(u)$. The wave profile in Fig. 8(b) is identical to that in Fig. 8(a), but shifted by one cell to the left.

In order to avoid over-diffusion at the shock, we prevent noise detection near shock discontinuities. This is achieved by noting that the function $C(x,t)$ attains a local maximum for points x along the shock curve. Consequently, we locate the local maximums of $C(x,t)$, by finding the cells for which $\partial_x C = 0$ and $\partial_{xx} C < 0$. We then deactivate the noise detection in the cells surrounding the shock curve.

Having deactivated the noise indicator near the discontinuity, the largest wavelet coefficients are now those where the high-frequency oscillations exist. We may then define the *noise detector function* $\mathbb{1}_{\text{noise}}(x)$ as follows: for each $j = 1, \ldots, \frac{M-1}{2}$ and $x \in \mathcal{I}_j$, we set $\mathbb{1}_{\text{noise}}(x) = 1$ if $|\mathcal{C}_j(u)| > \mathcal{C}_{\text{ref}} > 0$ and set $\mathbb{1}_{\text{noise}}(x) = 0$ otherwise.

The constant $\mathcal{C}_{\text{ref}}$ is obtained by computing the wavelet coefficient of a standard hat function on the interval $[-\Delta x, +\Delta x]$ with amplitude δh:

$$\mathcal{C}_{\text{ref}} = \delta h \cdot \sqrt{\frac{\Delta x}{12}}\,. \tag{28}$$

4.4. Noise removal algorithm

Having described the noise detection algorithm, we next propose an efficient scheme for removing noise from a given function $u(x)$ by solving a localized heat equation over the collection of intervals $\mathcal{I}_j$ where high-frequency noise has been detected.

The union of all noisy intervals $\mathcal{I}_j$ consists of K connected intervals $V_1, \ldots, V_K$. For each set V_k, $k = 1, \ldots, K$, we define the set W_k by affixing one cell on the left and one cell on the right. We then solve a localized heat equation for the "de-noised" solution $w(x,\tau)$ in each of the domains W_k:

$$\partial_\tau w(x,\tau) = \eta \cdot \partial_{xx} w(x,\tau), \text{ for } x \in W_k \text{ and } \tau \geq 0\,, \tag{29a}$$

$$w(x,0) = u(x), \qquad \text{for } x \in W_k\,, \tag{29b}$$

$$w(x,\tau) = u(x), \qquad \text{for } x \in \partial W_k\,, \tag{29c}$$

where $0 < \eta \ll 1$ is a small constant, which we refer to as the noise removal viscosity. The function $w(x,\tau) = u(x)$ for $x \in \left(\bigcup_{k=1}^{K} W_k\right)^{\mathcal{C}}$ and $\tau \geq 0$. We remark that the time τ is a "fictitious" time, introduced for the diffusion mechanism. Equation (29b) is the initial condition over the intervals where noise has been detected, and (29c) is a Dirichlet boundary condition ensuring that $w(x,\tau)$ continuously transitions to $u(x)$.

We use an explicit scheme to solve (29), and in practice, one time-step is sufficient to remove noise. If an explicit time-stepping scheme is used to solve (29), it is not necessary to construct the domains W_k. Instead, one can simply use the noise indicator function $\mathbb{1}_{\text{noise}}(x)$, and solve a modified heat equation:

$$\partial_\tau w(x,\tau) = \eta \cdot \mathbb{1}_{\text{noise}}(x) \cdot \partial_{xx} w(x,\tau), \quad \text{for } x_1 < x < x_M \text{ and } \tau \geq 0, \tag{30a}$$

$$w(x,0) = u(x), \qquad \text{for } x_1 < x < x_M, \tag{30b}$$

$$w(x,\tau) = u(x), \qquad \text{for } x = x_1 \text{ and } x = x_M. \tag{30c}$$

The utilization of an explicit scheme results in the stability constraint $\eta \Delta\tau/(\Delta x)^2 < 1/2$. However, in practice, we have found that much smaller values $\eta \Delta\tau/(\Delta x)^2 \ll 1/2$ are sufficient to dampen spurious noise. We also remark that the use of a single time-step means that the noise removal provided by the localized heat equation can be viewed as a *filtering* process, in which noise is removed through a local averaging. Consequently, the averaging provided by the Laplacian term on the

Table 2
Relevant parameters and variables used in the numerical tests.

Parameter / Variable	Description
β^u, β^E	artificial viscosity coefficients for the momentum and energy, respectively.
β_w^u, β_w^E	wall viscosity coefficients for the momentum and energy, respectively.
δh, η	amplitude of noise and noise removal viscosity, respectively.
ε, ε_w	parameters controlling support of C and C_w, respectively.
κ, κ_w	parameters controlling smoothness of C and C_w, respectively.

right-hand side of (30a), namely $(w_{i+1} - 2w_i + w_{i-1})/2$, can be replaced by other local averages, such as that provided by Gaussian filtering [3].

However, we wish to stress that there is a distinction between the operation of *smoothing* a noisy function and the noise removal process we have outlined. While, of course, removing high-frequency noise does indeed smooth the function, because we remove highly localized (in space and time) packets of oscillations, the procedure is quite different to more traditional smoothing algorithms, in which one uses truncation of frequencies in Fourier space or the analogous hyperviscosity operators in physical space. As such, it is difficult to obtain analytically the truncation error by means of a Taylor expansion, but it is possible to measure the error improvement by virtue of convergence studies comparing the algorithm with and without the noise removal algorithm activated. We provide results of such studies in §5.

4.5. The WENO-C-W-N algorithm

We now describe how we implement the above noise indicator algorithm for the Euler equations. The algorithm proceeds in two stages; in the first stage, we use the WENO-C-W scheme described in §3.3 to solve for an intermediary solution $\tilde{\boldsymbol{u}} = \left[\tilde{\rho}, \tilde{\rho u}, \tilde{E}\right]^T$; in the second stage, we feed this intermediary solution $\tilde{\boldsymbol{u}}$ into the noise indicator algorithm to de-noise the solution. The two-stage process is now described.

1. An intermediary solution $\tilde{\boldsymbol{u}}$ is obtained as

$$\tilde{\boldsymbol{u}}_i = \mathrm{RK}\left(\boldsymbol{u}_i^n, \mathcal{A}_{\mathrm{WENO}}(\boldsymbol{u}_i^n, \boldsymbol{C}_i^n)\right),$$
$$\boldsymbol{C}_i^{n+1} = \mathrm{RK}\left(\boldsymbol{C}_i^n, \mathcal{B}_{\mathrm{WENO}}(\boldsymbol{u}_i^n, \boldsymbol{C}_i^n)\right).$$

2. The intermediate velocity $\tilde{u}_i$ is then de-noised using the procedure described in §4.3 and §4.4, producing the noise-free velocity $u(x, t_{n+1})$. The updated solution $\boldsymbol{u}(x, t_{n+1})$ is then obtained as

$$\boldsymbol{u}(x, t_{n+1}) \equiv (\rho(x, t_{n+1}), \rho u(x, t_{n+1}), E(x, t_{n+1})) := \left(\tilde{\rho}(x), \tilde{\rho}(x) \cdot u(x, t_{n+1}), \tilde{E}(x)\right).$$

Remark 1. Implementation of our noise removal scheme has been motivated by the high-frequency oscillations of the velocity field that occur exactly at the contact discontinuity (see, for example, Fig. 6). We note that once the noise indicator function $\mathbb{1}_{\mathrm{noise}}(x)$ is computed, there are many possible choices for the noise removal portion of the algorithm. For example, while our implemented algorithm only removes high-frequency oscillations form the velocity, we could also remove such oscillations from ρ and E, and we could in place of the velocity field, instead remove oscillations from the momentum ρu. We have found that any of these choices produce the same relative errors in the one-dimensional test problems considered herein. Moreover, as we demonstrate in §5, the removal of high-frequency oscillations from u alone, is sufficient to remove noise from the density and energy as well. A more detailed examination of various noise removal algorithms (as well as those more ideally suited for parallelization) is made in [42].

5. Numerical simulations of classical shock tube experiments

In this section, we show results of the discretized C-method for a variety of classical shock tube experiments. For some of the problems, we will compare against WENO-based classical artificial viscosity schemes and Noh schemes. See Appendix A and Table 12 for a description of all of the numerical methods employed herein.

As with any artificial viscosity scheme, parameters must be chosen for the problem under consideration. Before presenting our numerical results, we consider this issue for the C-method, whose parameters are listed in Table 2. The artificial viscosity parameters β are chosen in the following manner: we set $\beta^E = 0$, choose β^u, β_w^u large enough so that post-shock oscillations are removed both pre and post shock-wall collision, then choose β_w^E large enough so that the wall-heating phenomenon (discussed later in §5.2) does not occur. A similar philosophy is applied to choice of parameters for the noise detection and removal algorithm: first, we determine the amplitude of highest-frequency oscillation δh and then choose the artificial viscosity parameter η large enough to diffuse the noise.

Table 3
L^1 error of the computed density minus the exact solution and convergence for the linear advection problem.

Scheme		Cells			
		51	101	201	401
WENO	Error	7.298×10^{-6}	2.318×10^{-7}	7.526×10^{-9}	2.654×10^{-10}
	Order	–	4.977	4.945	4.826

The parameters ε, ε_w, κ, and κ_w are $O(1)$ constants. Setting a larger value for ε or ε_w serves to increase the support of the corresponding C-function, while increasing the value of κ or κ_w produces smoother C-functions. For certain problems, smoothing the C-variables by using a larger κ further minimizes noise that occurs in the solution.

In Appendix C we demonstrate the accuracy of the C-method when the values of the parameters ε, ε_w, κ, and κ_w are fixed values for the different test problems. It is shown that the differences between the solutions computed using the optimized parameter sets we use for the problems in this section and the fixed-parameter sets we use for the runs in Appendix C are minimal, and that the fixed choice of parameters can be used for general problems. However, we wish to emphasize that one of the strengths of the C-method is its flexibility to optimize parameters for specific features associated with particular data.

The error analysis and convergence studies we perform for the numerical experiments considered in the following sections use the L^1, L^2, and L^∞ norms. Given two functions $f(x)$ and $g(x)$ defined on the computational grid with M cells, these error norms are defined by

$$||f-g||_{L^1} = \frac{1}{M}\sum_{i=1}^{M}|f(x_i)-g(x_i)|\,, \tag{31a}$$

$$||f-g||_{L^2} = \sqrt{\frac{1}{M}\sum_{i=1}^{M}|f(x_i)-g(x_i)|^2}\,, \tag{31b}$$

$$||f-g||_{L^\infty} = \max_{i=1,\ldots,M}|f(x_i)-g(x_i)|\,. \tag{31c}$$

As stated by Greenough & Rider [15], the L^1 and L^2 norms provide a global view of the errors in the computed solution, whereas the L^∞ norm highlights local errors, such as the undershoot or overshoot that occurs at a discontinuity. Thus, these three norms together provide a precise *quantitative* description of the errors of numerical solutions, and complement the *qualitative* evidence we provide through the visualization of numerical simulations.

5.1. Linear advection

We begin by considering a linear advection problem to demonstrate the high-order convergence of the base WENO scheme. The domain is $0 \le x \le 1$, the adiabatic constant is $\gamma = 1.4$, the initial data is

$$\begin{bmatrix} \rho_0 \\ (\rho u)_0 \\ E_0 \end{bmatrix} = \begin{bmatrix} 1+0.5\sin(2\pi x) \\ 1+0.5\sin(2\pi x) \\ 0.5+0.25\sin(2\pi x)+\frac{1}{\gamma-1} \end{bmatrix},$$

and we employ periodic boundary conditions. In the exact solution, the velocity and pressure remain a constant value of 1, while the initial density field is advected by the velocity, so that the density at time t satisfies $\rho(x,t) = \rho_0(x-t)$. We employ our simplified WENO scheme on grids with 51, 101, 201, and 401 cells. Each simulation is run with a CFL number of approximately 0.6, and the final time is $t = 1.0$.

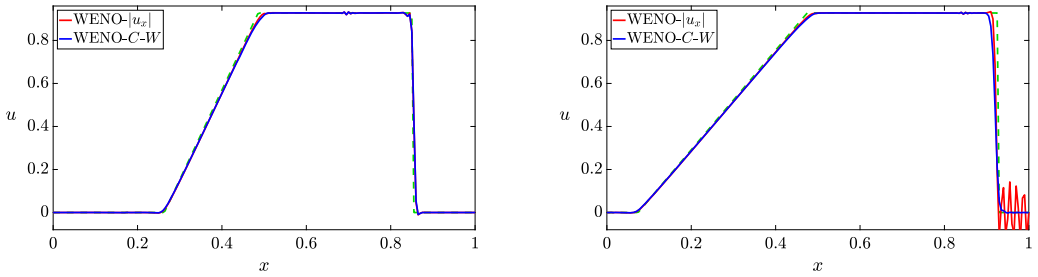
In Table 3, we list the L^1 error of the computed density minus the exact solution; as expected, the solutions converge with almost fifth-order accuracy.

5.2. The Sod shock tube problem

The data for the Sod shock tube is given in (24), with the exact solution given by a shock wave, a rarefaction wave, and a contact discontinuity. To simulate the shock-wave wall collision, we employ reflective boundary conditions (16). In the tests below, we employ our WENO-C-W scheme with 201 cells.

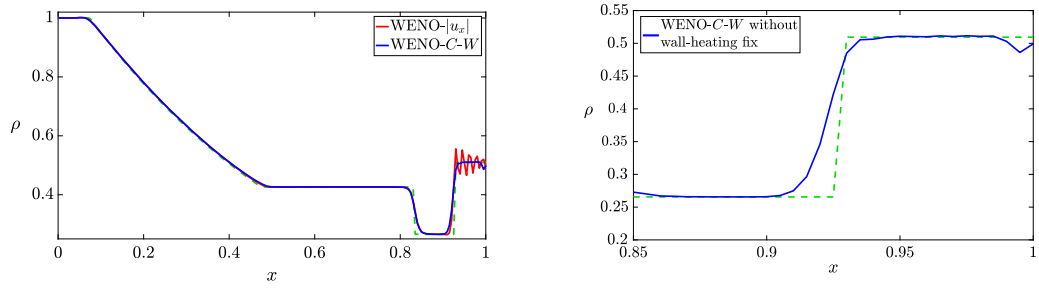
5.2.1. The wall-heating problem in the Sod shock tube

We first demonstrate the well-known wall-heating problem ([37,43]) in which an anomalous slope appears in the density and internal energy upon shock bounce-back. We shall then explain what the root cause of this problem is, and its solution.



(a) $t = 0.20$: pre shock-wall collision (b) $t = 0.36$: post shock-wall collision

Fig. 9. The velocity profile for the Sod shock tube problem, with the wall viscosity activated for the momentum equation. The dashed green curve is the exact solution.



(a) $t = 0.36$: post shock-wall collision (b) $t = 0.36$: zooming in on the undershoot

Fig. 10. The density profile for the Sod shock tube problem, calculated with the wall viscosity activated for the momentum equation. The dashed green curve is the exact solution.

We begin by choosing the parameters in equations (13) and (14) as

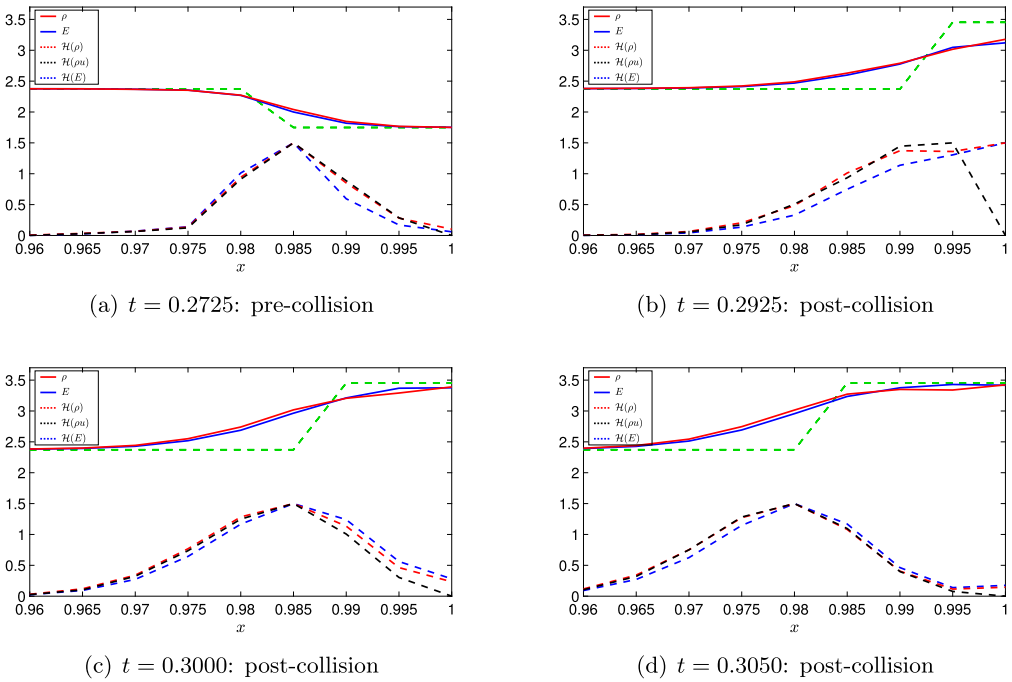
$$\beta^u = 0.5, \quad \beta^E = 0.0, \quad \beta^u_w = 3.0, \quad \beta^E_w = 0.0$$
$$\varepsilon = 1.0, \quad \kappa = 5.0, \quad \varepsilon_w = 50.0, \quad \kappa_w = 1.0\,.$$

The resulting solutions for the velocity before and after the shock-wall collision are shown in Fig. 9. Before the shock collision with the wall (Fig. 9(a)), the solution maintains a sharp shock front. After the shock collision (Fig. 9(b)), the high-frequency oscillations behind the shock wave are damped-out for sufficiently large $\beta^u_w > 0$, while maintaining a sharp front.

While post-collision oscillations in the density profile are suppressed, Fig. 10 shows the presence of the anomalous density slope $\partial_x \rho(1,t)$ at the wall (which should be zero). This incorrect slope is termed *wall heating* because the undershoot in the density results in an overshoot in the internal energy (7) (and hence temperature) at the wall. Noh [37] suggested that wall heating would occur in *any* artificial viscosity scheme, and is in fact built into the exact solutions of the difference equations of the artificial viscosity method. Menikoff [35] argues that wall-heating is caused by the *smearing* of the shock curve that occurs with any artificial viscosity scheme, and is thus unavoidable. Rider [43] argues that incorrect wave speeds result in too much or too little dissipation.

In fact, it appears that the wall-heating error is the result of the misalignment of the *gradient of fluxes* for the density, momentum and energy equations, which in turn is caused by a slight difference in the speed of the shock fronts for the density, momentum and energy. We define the *forcing terms*

$$\mathcal{H}(\rho) = -\partial_x(\rho u),$$
$$\mathcal{H}(\rho u) = -\partial_x(\rho u^2 + p) + \partial_x\left(\mathcal{B}^{(u)}(t)\,\rho\,C\,\partial_x u\right),$$
$$\mathcal{H}(E) = -\partial_x(u(E+p)) + \partial_x\left(\mathcal{B}^{(E)}(t)\,\rho\,C\,\partial_x(E/\rho)\right).$$



(a) $t = 0.2725$: pre-collision

(b) $t = 0.2925$: post-collision

(c) $t = 0.3000$: post-collision

(d) $t = 0.3050$: post-collision

Fig. 11. Comparison of the energy and energy forcing term $\mathcal{H}(E)$ (blue/blue dashed) with the density, suitably resized, and the density forcing term $\mathcal{H}(\rho)$ (red/red dashed) and the momentum forcing term $\mathcal{H}(\rho u)$ (black dashed) for the Sod shock tube problem with the wall viscosity activated for the momentum equation. The green dashed curve is the exact solution. The figures shown are zoomed in at the shock just before or just after the shock front has collided with the wall at $x = 1$.

In Fig. 11, we compare the energy and density profiles, along with the terms $\mathcal{H}(\rho)$, $\mathcal{H}(\rho u)$, and $\mathcal{H}(E)$, all suitably resized[3] for ease of comparison, at various times just before or after the shock fronts have collided with the wall, zoomed-in on the region next to the wall. In Fig. 11(a), the density and energy profiles are very similar, but the forcing terms are slightly misaligned; it is clear that $\mathcal{H}(E)$ is slightly behind both $\mathcal{H}(\rho)$ and $\mathcal{H}(\rho u)$. This misalignment causes the solution profiles for the energy and density to begin to diverge, as can be seen in Fig. 11(b). Again, there is a misalignment between the forcing terms $\mathcal{H}(E)$ and $\mathcal{H}(\rho)$. As the shock moves away from the wall in Fig. 11(c) and Fig. 11(d), the difference between the solution profiles is now clear. Even though the forcing terms are now better aligned, the earlier misalignment ensures that the difference between the energy and density profiles is permanent.

5.2.2. A solution to the wall-heating problem

The solution to the wall heating problem suggested by Noh [37] is the addition of a heat conduction term to the energy equation. For the WENO-Noh scheme we implement in this study, we shall use a heat conduction term of the form[4]

$$\partial_x \left(\beta^E_{\text{Noh}}\, \rho\, |\partial_x u|\, \partial_x e \right), \tag{32}$$

where $e = p/(\gamma - 1)\rho = c_v \Theta$ is the internal energy of the system, proportional to the temperature Θ, with c_v the specific heat capacity at a constant volume.

We use the following artificial (wall) viscosity for the energy equation (13c):

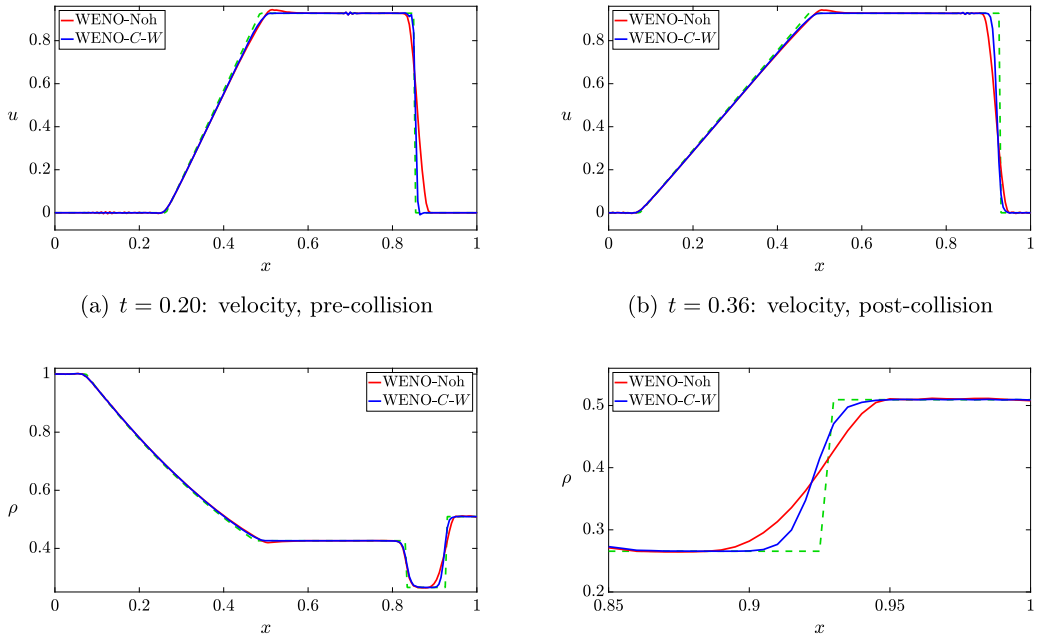
$$\partial_x \left(\mathcal{B}^{(E)}(t)\, \rho\, C\, \partial_x (E/\rho) \right). \tag{33}$$

There are two differences between the terms (33) and (32):

1. While (32) uses the oscillatory localizing coefficient $|\partial_x u(x,t)|$, we instead use the space-time smooth localizer $C(x,t)$.

[3] More precisely, we plot the following: first, $\frac{3}{2}\frac{\mathcal{H}}{\max_\Omega \mathcal{H}}$ for each of the forcing terms $\mathcal{H}(\rho)$, $\mathcal{H}(\rho u)$, and $\mathcal{H}(E)$; second, the function $1.1928 + 4.4403\rho$; and finally, the energy E.

[4] In equations (2.1)-(2.5) in the paper of Noh [37], there is, in fact, an additional term proportional to $-\rho|\partial_x u|^2 \partial_x u$ on the right-hand side of (32). We have found that this term is not necessary to remove the wall-heating error, and thus omit it from the Noh scheme we implement in this paper.



(a) $t = 0.20$: velocity, pre-collision

(b) $t = 0.36$: velocity, post-collision

(c) $t = 0.36$: density, post-collision

(d) $t = 0.36$: density, post-collision zoom-in

Fig. 12. The velocity and density profiles for the Sod shock tube problem before and after shock-wall collision.

2. We use $\partial_x(E/\rho)$ in our diffusion operator rather than the function $\partial_x e$. This difference can be explained as follows: equation (7) shows that

$$\partial_x\left(\mathcal{B}^{(E)}(t)\,\rho\,C\,\partial_x(E/\rho)\right) = \partial_x\left(\mathcal{B}^{(E)}(t)\,\rho\,C\,\partial_x e\right) + \partial_x\left(\mathcal{B}^{(E)}(t)\,\rho\,C\,u\partial_x u\right). \tag{34}$$

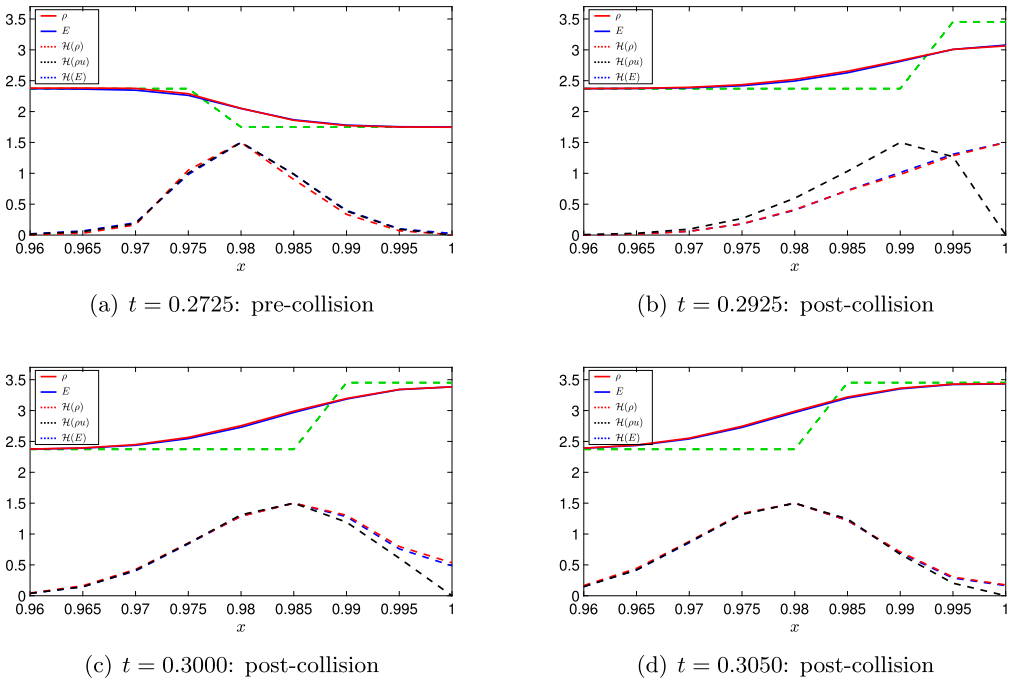
Hence, (34) has a similar form to (32) (with C replacing $|u_x|$), but with the additional term $\partial_x\left(\mathcal{B}^{(E)}(t)\,\rho\,C\,u\partial_x u\right)$. Indeed, the two terms in (34) are both proper diffusion operators near shock waves. This is easy to see: multiplying (34) by E and integrating by parts then shows that

$$\int_{x_1}^{x_M} \mathcal{B}^{(E)}(t)\,\rho\,C\,\partial_x e\partial_x E\,dx + \int_{x_1}^{x_M} \mathcal{B}^{(E)}(t)\,\rho\,C\,u\,\partial_x u\partial_x E\,dx.$$

At the shock, $\partial_x e$ has the same sign as $\partial_x E$ so that $\int_{x_1}^{x_M} \mathcal{B}^{(E)}(t)\,\rho\,C\,\partial_x e\partial_x E\,dx \geq 0$; moreover, in the case of a right-traveling shock front, $\partial_x u < 0$, $\partial_x E < 0$ and $u > 0$ at the shock, so that $\int_{x_1}^{x_M} \mathcal{B}^{(E)}(t)\,\rho\,C\,u\,\partial_x u\partial_x E\,dx \geq 0$, while for a left-moving shock, $\partial_x u < 0$, $\partial_x E > 0$ and $u < 0$, so that once again $\int_{x_1}^{x_M} \mathcal{B}^{(E)}(t)\,\rho\,C\,u\,\partial_x u\partial_x E\,dx \geq 0$. This then ensures that (33) is a *dissipative* operator and that the structure of the artificial viscosity term in (33) adjusts the Noh-type dissipation $\partial_x\left(\mathcal{B}^{(E)}(t)\,\rho\,C\,\partial_x e\right)$ by the velocity-dependent term $\partial_x\left(\mathcal{B}^{(E)}(t)\,\rho\,C\,u\partial_x u\right)$.

In our numerical experiments, presented below, we compare Noh's scheme, called WENO-Noh (see Appendix A), with our WENO-C-W scheme. For WENO-Noh, we set $\beta^u_{\text{Noh}} = 15.0$ and $\beta^E_{\text{Noh}} = 10.0$ in (40). These viscosity coefficients were chosen in the following manner: β^u_{Noh} was first chosen large enough to suppress the post-collision oscillations, and then β^E_{Noh} was chosen to correct the wall-heating error. In our WENO-C-W scheme, we set $\beta^u = 0.5$, $\beta^u_w = 3.0$, $\beta^E = 0.0$, and $\beta^E_w = 6.0$.

In Fig. 12, we compare the velocity and density profiles computed with the two schemes above. It is clear that the WENO-C-W scheme produces a superior solution both before and after the shock-wall collision. The large amount of viscosity needed in the WENO-Noh scheme post-collision means that the solution prior to shock-wall collision is affected, with a smeared shock curve and overshoot at the top of the expansion wave. Moreover, even the relatively large value of β^u_{Noh} as compared with $\beta^u + \beta^u_w$ is unable to fully suppress the oscillations behind the shock curve that occur post-collision. This is due to the smoothness of the localizing coefficient C as compared with the rough nature of $|\partial_x u|$.



(a) $t = 0.2725$: pre-collision

(b) $t = 0.2925$: post-collision

(c) $t = 0.3000$: post-collision

(d) $t = 0.3050$: post-collision

Fig. 13. Comparison of the energy and energy forcing term $\mathcal{H}(E)$ (blue/blue dashed) with the density, suitably resized, and the density forcing term $\mathcal{H}(\rho)$ (red/red dashed) and the momentum forcing term $\mathcal{H}(\rho u)$ (black dashed) for the Sod shock tube problem with the wall viscosity activated for the momentum and energy equations. The green dashed curve is the exact solution. The figures shown are zoomed in at the shock just before or just after the shock front has collided with the wall at $x = 1$.

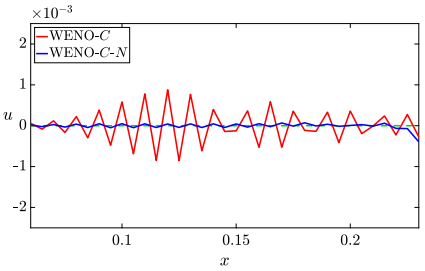
As is the case with the velocity profile, prior to shock collision the WENO-C-W scheme produces a superior solution for the density profile, with a much sharper shock front and more accurate expansion wave. Post-collision, the heat conduction terms ensure that neither of the methods exhibit the wall heating error. However, there are still small oscillations present in the solution computed with the WENO-Noh scheme, and the shock front is much more smeared than that of the solution computed with WENO-C-W.

Finally, comparing Fig. 11 and Fig. 13, we see that the wall viscosity for the energy equation has properly aligned the forcing terms $\mathcal{H}(\rho)$, $\mathcal{H}(\rho u)$ and $\mathcal{H}(E)$. Realignment of the gradient of fluxes removes the wall-heating problem, created by the smearing of the shock fronts. The artificial viscosity term (33) is responsible for this realignment.

5.2.3. *Noise removal with the noise indicator*

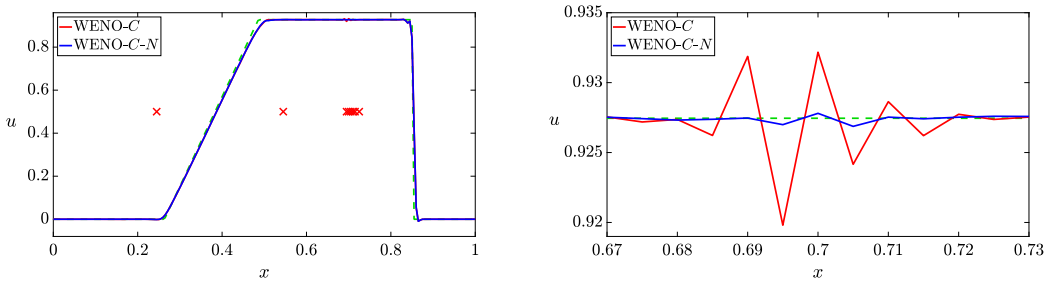
We now employ our noise indicator algorithm to the Sod shock tube problem with the aim of removing the noise present in the velocity profile at the contact discontinuity.

In the test below, we employ our WENO-C-N scheme with η chosen such that $\eta\Delta\tau/\Delta x^2 = 0.005$ in the heat equation used for noise removal; an explicit time-stepping scheme is used and only one time-step is taken. For the noise detection algorithm, C_{ref} in (28) is computed using $\delta h = 0.0001$. Fig. 14(b) shows that the noise indicator removes the spurious oscillations from the velocity profile. The localized diffusion mechanism ensures that the solution in other regions is unchanged, and this is demonstrated in Fig. 14(a). We note that the noise indicator algorithm affects neither the sharpness of the shock front nor the speed of the shock, nor the order of the numerical method (which we shall show results for below).



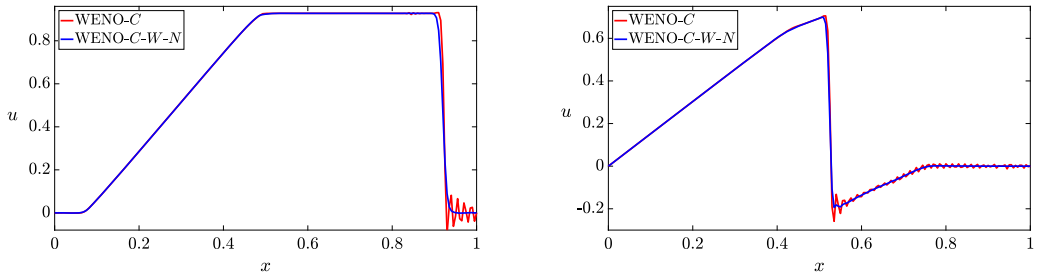
There is also high-frequency noise present to left of the expansion wave (shown in the figure to the left); again, the noise indicator detects and removes this noise.

In Fig. 15(a), we show the velocity profile, computed using the WENO-C-W-N scheme, after the shock-wall collision. Here, all of the post-collision noise is damped by the wall viscosity. The WENO-C-W-N scheme removes spurious oscillations in the solution, while ensuring that a sharp shock front and the correct wave speed are retained, even after multiple shock-wall collisions, as shown in Fig. 15(b).



(a) $t = 0.20$ (b) $t = 0.20$: zooming in on the noise at the contact discontinuity

Fig. 14. Comparison of the velocity profiles for the Sod shock tube problem computed with WENO-C and WENO-C-N with 201 cells. The red crosses in Fig. 14(a) indicate where the function $\mathbb{1}_{\text{noise}}(x)$ is active.



(a) $t = 0.36$ (b) $t = 0.90$

Fig. 15. Comparison of the velocity solution profile produced using WENO-C and WENO-C-W-N for the post-collision bounce-back in the Sod shock tube problem with 201 cells.

5.2.4. Error analysis and convergence tests

We now compare the errors of the various numerical schemes given in Table 12 in Appendix A applied to the Sod shock-wall collision and bounce-back problem. The solutions are computed with all parameters fixed across the different methods, giving an objective evaluation of each scheme.

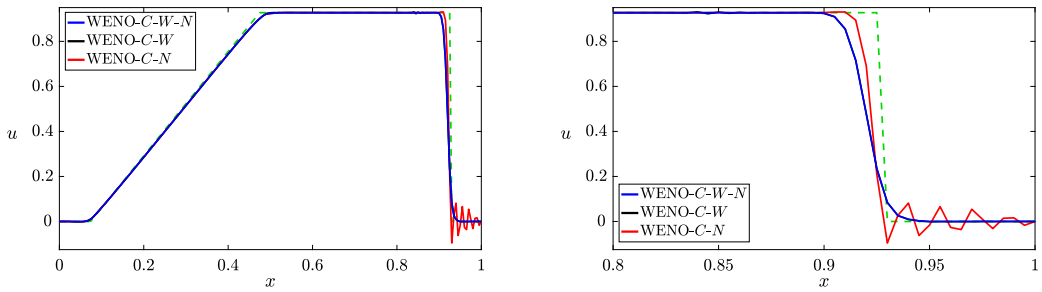
As advised by Greenough & Rider [15], and in order to fairly compare our results with those found in the numerics literature, we use the CFL number equal to 0.6. We have found that the use of a smaller CFL number of 0.3 does not (appreciably) change the conclusions of our numerical tests. For instance, the solutions produced using WENO and WENO-C-W-N show (roughly) the same relative error and order of convergence when CFL $= 0.3$ as they do when CFL $= 0.6$. However, the presence of the nonlinear artificial viscosity terms in the Euler-C equations, when combined with an explicit time-integration scheme, place restrictions on the CFL number that would otherwise not be present in the stand-alone WENO algorithm. In particular, for the Sod test problem, due to the additional artificial viscosity present during the shock-wall collision phase, we have found an upper bound on the CFL number to be ≈ 0.7. While our stand-alone WENO scheme is (formally) stable for much larger CFL numbers, the relative error and the order of convergence degrades as the CFL number is increased. Indeed, it is demonstrated by Greenough & Rider [15] that only 75% of the fifth-order convergence rate of WENO is achieved when CFL $= 1.0$, whereas the full fifth-order convergence is achieved for CFL $= 0.6$. Therefore, the use of the smaller CFL $= 0.6$ is also necessary for the stand-alone WENO scheme.

In Table 4, we list the L^1 error of the computed velocity minus the exact solution, and study the order of convergence for the Sod problem with shock-wall collision and bounce-back. WENO-C produces solutions that are significantly better than those produced with WENO-$|u_x|$, which are significantly better than those solutions produced with the stand-alone WENO algorithm. Note that the use of the noise removal algorithm consistently improves the error bounds, while maintaining the order of accuracy.

On the coarser grids containing 101 or 201 cells, the WENO-C-W and WENO-C-W-N schemes produce solutions with slightly larger errors than the solution produced with WENO-C-N; this is caused by the slight smearing of the shock, post wall collision. The solutions are, however, *qualitatively* significantly better, as evidenced by Fig. 12 and Fig. 15 above, as well as Fig. 16 below. Both WENO-C-W and WENO-C-W-N maintain a relatively high order of accuracy, whereas the presence of the post-collision noise ensures that both WENO and WENO-$|u_x|$ have convergence rates that are irregular and relatively poor.

Table 4
Post shock-wall collision ($t = 0.36$) L^1 error of the computed velocity minus the exact solution and convergence for the Sod problem with shock-wall collision and bounce-back.

Scheme		Cells			
		101	201	401	801
WENO	Error	1.662×10^{-2}	1.772×10^{-2}	1.086×10^{-2}	8.214×10^{-3}
	Order	–	−0.093	0.706	0.403
WENO-$\|u_x\|$	Error	1.534×10^{-2}	1.441×10^{-2}	8.444×10^{-3}	5.864×10^{-3}
	Order	–	0.090	0.771	0.526
WENO-Noh	Error	3.436×10^{-2}	1.799×10^{-2}	9.117×10^{-3}	4.795×10^{-3}
	Order	–	0.934	0.980	0.927
WENO-*N*	Error	1.667×10^{-2}	1.666×10^{-2}	1.064×10^{-2}	7.262×10^{-3}
	Order	–	0.001	0.648	0.550
WENO-*C*	Error	1.520×10^{-2}	1.160×10^{-2}	6.453×10^{-3}	3.927×10^{-3}
	Order	–	0.390	0.846	0.717
WENO-*C*-*N*	Error	1.504×10^{-2}	1.134×10^{-2}	6.412×10^{-3}	3.780×10^{-3}
	Order	–	0.407	0.823	0.763
WENO-*C*-*W*	Error	1.990×10^{-2}	1.151×10^{-2}	5.774×10^{-3}	3.019×10^{-3}
	Order	–	0.790	0.995	0.936
WENO-*C*-*W*-*N*	Error	1.983×10^{-2}	1.146×10^{-2}	5.770×10^{-3}	3.018×10^{-3}
	Order	–	0.791	0.990	0.935



(a) $t = 0.36$, velocity post-collision (b) $t = 0.36$, velocity post-collision zoom-in

Fig. 16. Comparison of the velocity solution profile produced using WENO-*C*-*W*-*N*, WENO-*C*-*W*, and WENO-*C*-*N* for the post-collision bounce-back in the Sod shock tube problem with 201 cells.

We remark that our conclusions described above do not change if we replace the L^1 norm with either the L^2 or L^∞ norms. We list in Table 5 the L^2 and L^∞ error analysis for the post shock-wall collision velocity for the Sod problem, where the velocity is computed using either WENO-$|u_x|$ or WENO-*C*-*W*-*N*.

For the L^2 error analysis, we first note the odd behavior for WENO-$|u_x|$ on the coarser grids with 101 and 201 cells. The increase in the L^2 error despite mesh refinement is caused by the base WENO scheme; referring to Table 4, we see that the L^1 error of solutions produced with WENO increases as the mesh is refined from 101 to 201 cells. This phenomenon is due to the large oscillations that occur post shock-wall collision. The WENO-*C*-*W*-*N* scheme removes these oscillations, at the cost of a slight smearing of the shock front; this smearing results in a larger L^2 error on coarse grids when compared with WENO-$|u_x|$, but smaller L^2 errors and better rates of convergence as the mesh is refined.

Table 5 shows that the L^∞ errors for both WENO-$|u_x|$ and WENO-*C*-*W*-*N* grow as the mesh is refined. As noted in [15], this is due to the localization of the error at the shock. However, we remark that the L^∞ errors for WENO-*C*-*W*-*N* are smaller than the L^∞ errors for WENO-$|u_x|$ on the grids with 201, 401, and 801 cells; moreover, these errors grow at a faster rate for WENO-$|u_x|$ than for WENO-*C*-*W*-*N*.

In addition to the figures and qualitative evidence provided, the L^1, L^2, and L^∞ error studies indicate that the *C*-method produces highly accurate solutions with close to optimal rates of convergence for the Sod shock-wall collision and bounce-back test.

5.2.5. Comparison with other schemes

For the purposes of benchmarking our WENO and WENO-*N* schemes prior to shock-wall collision, we present error analysis and convergence rates comparing our simplified WENO scheme with the scheme utilized by Greenough and Rider in [15]. The WENO scheme that is presented in [15] is formally fifth-order accurate in space, with time integration done

Table 5
Post shock-wall collision ($t = 0.36$) L^2 and L^∞ error of the computed velocity minus the exact solution and convergence for the Sod problem with shock-wall collision and bounce-back.

Norm	Scheme		Cells			
			101	201	401	801
L^2	WENO-$\lvert u_x\rvert$	Error	4.775×10^{-2}	6.068×10^{-2}	4.640×10^{-2}	3.765×10^{-2}
		Order	–	−0.346	0.387	0.302
	WENO-C-W-N	Error	6.953×10^{-2}	6.098×10^{-2}	4.423×10^{-2}	3.324×10^{-2}
		Order	–	0.189	0.463	0.412
L^∞	WENO-$\lvert u_x\rvert$	Error	4.262×10^{-1}	7.417×10^{-1}	8.024×10^{-1}	8.925×10^{-1}
		Order	–	−0.799	−0.113	−0.154
	WENO-C-W-N	Error	5.663×10^{-1}	6.926×10^{-1}	7.124×10^{-1}	7.456×10^{-1}
		Order	–	−0.290	−0.041	−0.066

Table 6
Pre shock-wall collision ($t = 0.20$) L^1_{GR} error analysis and convergence tests for the density for the Sod shock tube problem.

Scheme		Cells		
		101	201	401
WENO	Error	1.57×10^{-2}	7.93×10^{-3}	4.49×10^{-3}
	Order	–	0.99	0.82
WENO-N	Error	1.60×10^{-2}	7.90×10^{-3}	4.37×10^{-3}
	Order	–	1.02	0.85
RK3-WENO5	Error	1.58×10^{-2}	8.24×10^{-3}	4.47×10^{-3}
	Order	–	0.93	0.88

using a total variation diminishing (TVD) third-order Runge-Kutta method. Flux-splitting is accomplished using a method similar to the Lax-Friedrichs flux-splitting (see [15] for the details). We will refer to this method as RK3-WENO5.

The error norm utilized in [15] is of the form

$$||\rho(\cdot,t) - \rho^*(\cdot,t)||_{L^1_{GR}} := \frac{1}{M}\sum_{i=1}^{M} \frac{|\rho(x_i,t) - \rho^*(x_i,t)|}{|\rho^*(x_i,t)|},$$

where ρ is the computed density and ρ^* is the exact solution for the density. We will refer to this norm as the L^1_{GR} norm.

In Table 6, we calculate the L^1_{GR} errors for the density for the Sod shock tube problem computed with WENO and WENO-N, and compare them with the corresponding values in [15]. All simulations were run with a CFL number of 0.6. We see that our simplified WENO scheme and noise indicator algorithm compare well with the more complicated RK3-WENO5 scheme. Consequently, using our simplified WENO algorithm for the purposes of comparison in our error analysis for the artificial viscosity methods presented is justified; that is to say, comparing the performance of our artificial viscosity methods with our simplified WENO scheme is similar to comparing the performance of our artificial viscosity methods with the more complicated (and 'industry-standard') RK3-WENO5.

5.3. The Noh problem

As a further example of wall-heating, we next consider the classical 1-D planar Noh problem [37,25]. The domain of interest is $-0.5 \le x \le 0.5$, the adiabatic constant is $\gamma = 5/3$, and the initial data is given by

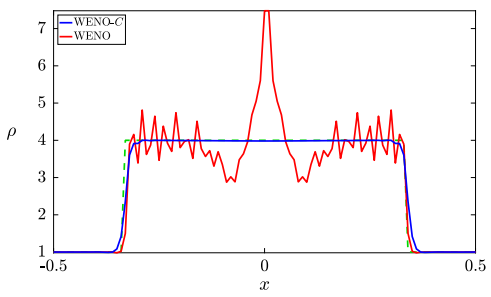
$$\begin{bmatrix} \rho_0 \\ (\rho u)_0 \\ E_0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 0.5 + \frac{10^{-6}}{\gamma - 1} \end{bmatrix} \mathbb{1}_{[-0.5,0)}(x) + \begin{bmatrix} 1 \\ -1 \\ 0.5 + \frac{10^{-6}}{\gamma - 1} \end{bmatrix} \mathbb{1}_{[0,0.5]}(x).$$

The solution for this problem consists of two infinite strength shocks propagating with speed 1/3 outwards from the origin, with a state of constant density and pressure left behind.

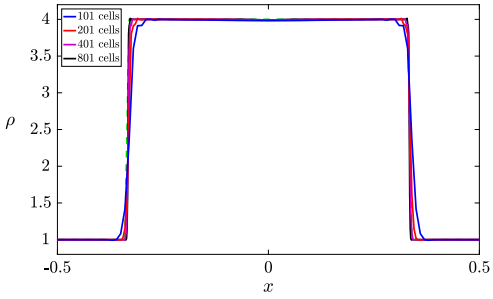
As demonstrated in [25], most schemes tend to produce the anomalous wall-heating at the center origin. We shall utilize our WENO-C method (i.e. no shock-wall collision algorithm) with 101 cells. We choose the relevant parameters as

$$\beta^u = 1.0, \qquad \beta^E = 10.0, \qquad \varepsilon = 50.0, \qquad \kappa = 1.0.$$

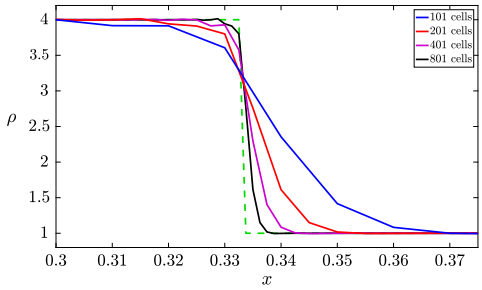
The value of β^u is chosen large enough to eliminate post-shock oscillations, while β^E is chosen to minimize the wall-heating. In Fig. 17, we compare the solutions computed using WENO and WENO-C; it is clear that WENO-C produces a much more accurate solution, with the post-shock oscillations and wall-heating eliminated.



(a) $t = 1.0$: density, comparison of WENO-C and WENO



(b) $t = 1.0$: density, comparison of WENO-C with different grid spacings



(c) $t = 1.0$: density zoom-in at the shock, comparison of WENO-C with different grid spacings

Fig. 17. The density profile at time $t = 1.0$ for the Noh problem, with the solution computed using (a) WENO or (b, c) WENO-C. The dashed green curve is the exact solution.

5.4. The LeBlanc shock tube problem

We now turn our attention to the LeBlanc shock tube problem. Here, the domain of interest is $0 \le x \le 9$, the adiabatic constant is $\gamma = \frac{5}{3}$, and the initial data is given by

$$\begin{bmatrix} \rho_0 \\ (\rho u)_0 \\ E_0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 10^{-1} \end{bmatrix} \mathbb{1}_{[0,3)}(x) + \begin{bmatrix} 10^{-3} \\ 0 \\ 10^{-9} \end{bmatrix} \mathbb{1}_{[3,9]}(x)\,.$$

The large jump in the initial energy E_0 produces a very strong shock wave, making the LeBlanc shock-tube problem a very difficult test case. Most numerical schemes tend to produce large overshoots in the internal energy at the contact discontinuity, which results in a loss of accuracy in the shock speed, as shown in Fig. 18. This overshoot in the internal energy is in fact an example of wall-heating; a small undershoot in the density and the continuity of the pressure at the contact produce this observed overshoot in the internal energy. We refer the reader to [40,26,30] for further details.

Our strategy is to add an additional diffusion term to the right-hand side of the energy equation (13c) that will serve to remove the large overshoot in the internal energy at the contact discontinuity. Specifically, we solve an additional C-equation for a variable C^e forced by $|\partial_x e|/\max_x |\partial_x e|$. Thus, equation (13c) is replaced by

$$\partial_t E + \partial_x(uE + up) = \partial_x\left(\mathcal{B}^{(E)}(t)\,\rho\, C\,\partial_x(E/\rho)\right) + \partial_x\left(\mathcal{B}^{(e)}(t)\,\rho\, C^e\,\partial_x\,(E/\rho)\right)\,, \tag{35}$$

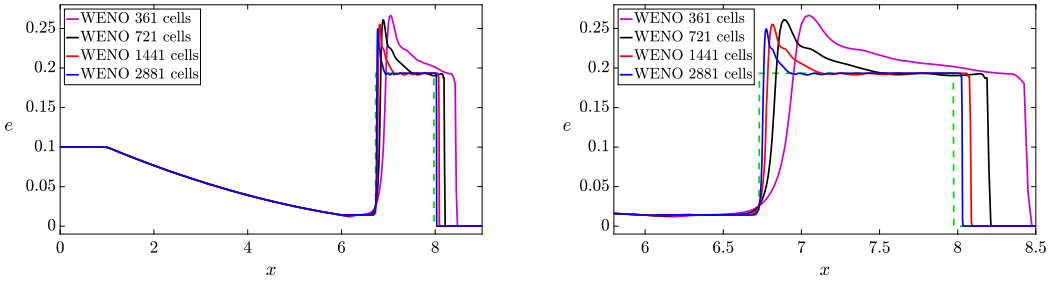
where the function C^e is computed using

$$\partial_t C^e + \frac{S(\boldsymbol{u})}{\varepsilon_e \Delta x} C^e - \kappa_e \Delta x \cdot S(\boldsymbol{u})\partial_{xx} C^e = \frac{S(\boldsymbol{u})}{\varepsilon_e \Delta x} G^e\,. \tag{36}$$

The artificial viscosity coefficients are given by (14) and

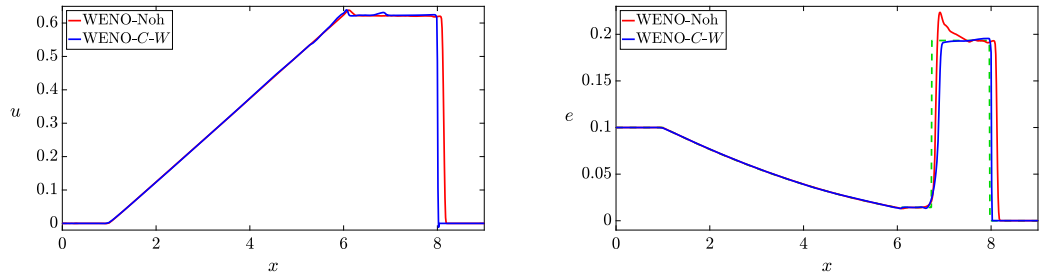
$$\mathcal{B}^{(e)}(t) = (\Delta x)^2 \cdot \frac{\max_x |\partial_x u|}{\max_x C^e}\left(\beta^e + \beta^e_w \cdot \overline{C}_w(t)\right)\,,$$

and C, $C_w(x,t)$, and $\overline{C}_w(t)$ are defined by (13d), (13e), and (15), respectively. The forcing to the C^e equation (36) is



(a) $t = 6.0$: internal energy (b) $t = 6.0$: internal energy, zoomed in

Fig. 18. The internal energy profile at time $t = 6.0$ for the LeBlanc shock tube problem, with the solution computed using WENO. The dashed green curve is the exact solution.



(a) $t = 6.0$: velocity (b) $t = 6.0$: internal energy

Fig. 19. The (a) velocity and (b) internal energy profiles for the LeBlanc shock tube problem before the collision with the wall. The solution is computed with viscosity activated for the momentum and energy equations.

$$G^e(x,t) = \mathbb{1}_{(0,\infty)}(\partial_x u)\frac{|\partial_x e|}{\max_x |\partial_x e|}\,.$$

Here, the indicator function $\mathbb{1}_{(0,\infty)}(\partial_x u)$ represents an *expansion switch*, in which G^e is non-zero only if $\partial_x u > 0$.

5.4.1. Stabilizing shock-wall collision

To simulate the collision of the shock-wave with the wall, we use solid wall boundary conditions (16). Motivated by the results for the Sod shock tube problem presented in 5.2.1, we add wall viscosity to the momentum and energy equations; we choose the parameters as

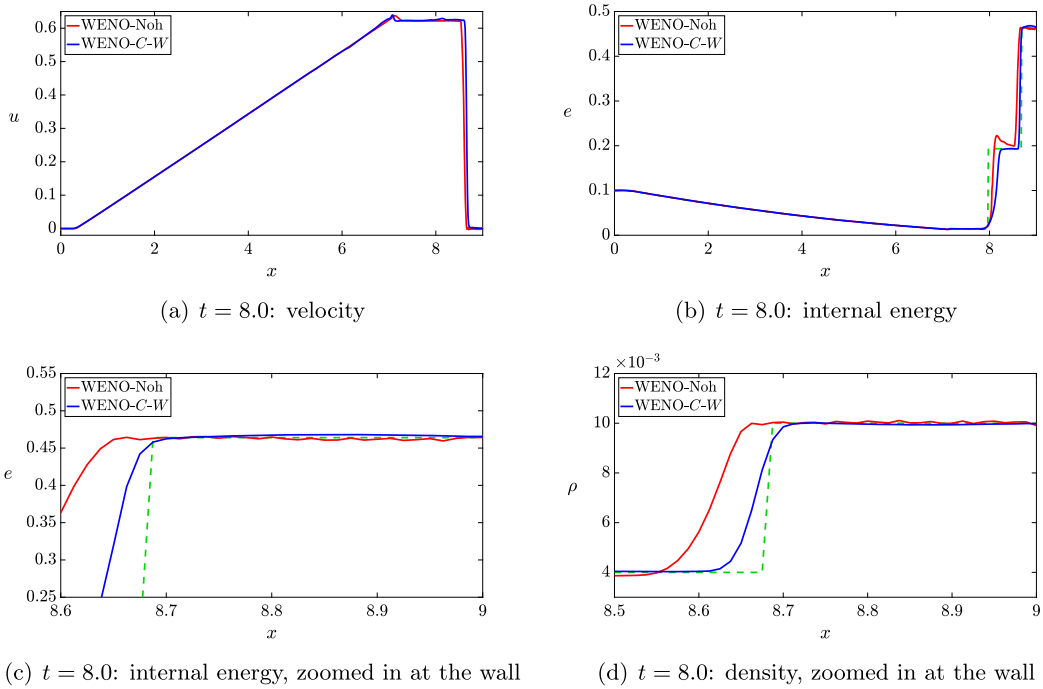
$$\beta^u = 0.001, \quad \beta^E = 0.0, \quad \beta^e = 0.4, \quad \beta^u_w = 4.0, \quad \beta^E_w = 0.0, \quad \beta^e_w = 0.0,$$
$$\varepsilon = 1.25, \quad \kappa = 10.0, \quad \varepsilon_e = 1.25, \quad \kappa_w = 14.0, \quad \varepsilon_w = 50.0, \quad \kappa_w = 4\,.$$

We employ our WENO-C-W scheme with 721 cells. Since this is a more challenging problem than the Sod shock tube problem, we use the smaller CFL number of 0.25.

For the purpose of comparison, we also implement the WENO-Noh scheme, with the parameters in (40) set as $\beta^u_{\text{Noh}} = 8.0$ and $\beta^E_{\text{Noh}} = 9.0$. These parameters were chosen with the aim of suppressing post-collision noise while preventing the occurrence of the wall heating error. We remark that WENO-Noh failed for CFL $= 0.25$, and required the much smaller CFL ≈ 0.045 to run.

The shock-wave moves to the right and collides with the right wall at time $t \approx 7.2$. Prior to shock collision, the WENO-Noh scheme produces a solution with an overshoot in the internal energy at the contact discontinuity. This results in an incorrect shock front and wave speed. The viscosity for the momentum at the shock and the energy at the contact discontinuity in our WENO-C-W scheme remove post-shock oscillations and the overshoot in the internal energy, respectively, as shown in Fig. 19.

Post shock-wall collision, the wall viscosity for the momentum and energy equations damp-out the oscillations behind the shock, while ensuring that the solution maintains a sharp shock front and the correct shock speed (see Fig. 20). Moreover, the wall viscosity for the energy equation prevents the wall heating error from occurring, as shown in Fig. 20(d). Due to the lack of smoothness of the localizing artificial viscosity coefficient $|\partial_x u|$, the WENO-Noh scheme is unable to fully suppress all the post-collision oscillations, though the heat conduction term in the energy equation prevents the wall heating error from occurring. In Fig. 20(c), we zoom in on the internal energy profile near the wall; it is evident that the



(a) $t = 8.0$: velocity

(b) $t = 8.0$: internal energy

(c) $t = 8.0$: internal energy, zoomed in at the wall

(d) $t = 8.0$: density, zoomed in at the wall

Fig. 20. The (a) velocity, (b) internal energy, (c) zoomed in internal energy and (d) zoomed in density profiles for the LeBlanc shock tube problem after the collision with the wall. The solution computed with WENO-C-W has the wall viscosity activated for the momentum and energy equations.

solution computed with WENO-C-W is better than that computed with WENO-Noh, but there is a small error between the computed solution and the exact solution. This error occurs because of a very small inaccuracy in the density profile, shown in Fig. 20(d). Since the density is so small here, and since the internal energy is given by (7), even tiny errors are greatly amplified, making it very difficult to get a completely accurate solution for the internal energy.

5.4.2. Error analysis and convergence tests

We now compare the errors of the various numerical schemes listed in Table 12 applied to the LeBlanc shock tube problem, with the various relevant parameters fixed across the different methods. The L^1 errors in the velocity are computed using formula (31a), and are listed in Table 7 (time $t = 6.0$) and Table 9 (time $t = 8.0$). All of the simulations were run with a CFL number of 0.25, except for WENO-Noh, which required CFL ≈ 0.045.

Prior to shock-wall collision, it is evident from Table 7 that the C-method produces a solution that is significantly better than those solutions produced without the C-method. The L^1 errors for the velocity computed using WENO-C are almost an order of magnitude smaller than the L^1 errors for the velocity computed using WENO and WENO-Noh. This is primarily due to the removal of the overshoot in the internal energy, which results in an accurate shock speed.

On the other hand, the removal of the overshoot in the internal energy through the use of C^e results in a more smeared contact discontinuity, as shown in Fig. 19(b). The smearing of the contact discontinuity results in a non-physical "bump" appearing in the velocity profile, as shown in Fig. 19(a). Note that this bump does not appear in the velocity profile computed using WENO-Noh, since in this case the contact discontinuity is sharper, at the expense of a large overshoot in the internal energy. We suggest, however, that this defect in the solution computed using C^e is relatively insignificant when compared against the magnitude of the error in the internal energy solutions computed without C^e. This is primarily due to the fact that the internal energy error results in a highly inaccurate shock speed which, in turn, leads to a highly inaccurate solution, as evidenced by Table 7. On the other hand, the velocity bump error arises from the correction of the overshoot in the internal energy, and subsequently the shock speed and location; the latter two corrections result in a much more accurate solution overall, again demonstrated in Table 7. Moreover, we note that the bump error decreases with mesh refinement approximately four times as fast as the overshoot error, as shown in Table 8. Here, the overshoot/bump error is computed by calculating the difference between the value at the peak of the overshoot/bump and the value of the exact solution there.[5]

We note here that Table 7 seems to suggest that the WENO and WENO-Noh schemes produce solutions that converge at first-order, even though the solutions computed using these schemes are very poor, as shown, for example, in Fig. 18.

[5] This error is thus a *local* L^∞ error.

Table 7
Pre shock-wall collision ($t = 6.0$) L^1 error analysis and convergence tests for the velocity for the LeBlanc shock tube problem.

Scheme		Cells			
		361	721	1441	2881
WENO	Error	3.469×10^{-2}	1.659×10^{-2}	8.010×10^{-3}	4.016×10^{-3}
	Order	–	1.065	1.050	0.996
WENO-Noh	Error	2.546×10^{-2}	1.239×10^{-2}	6.001×10^{-3}	3.010×10^{-3}
	Order	–	1.040	1.045	0.996
WENO-*N*	Error	3.468×10^{-2}	1.661×10^{-2}	8.015×10^{-3}	4.022×10^{-3}
	Order	–	1.062	1.051	0.995
WENO-*C*	Error	7.190×10^{-3}	3.959×10^{-3}	2.008×10^{-3}	1.096×10^{-3}
	Order	–	0.864	0.976	0.873
WENO-*C*-*N*	Error	7.169×10^{-3}	3.881×10^{-3}	2.007×10^{-3}	1.113×10^{-3}
	Order	–	0.885	0.951	0.851

Table 8
Comparison of the overshoot error in the internal energy and the bump error in the velocity for solutions to the LeBlanc shock tube problem at time $t = 6.0$.

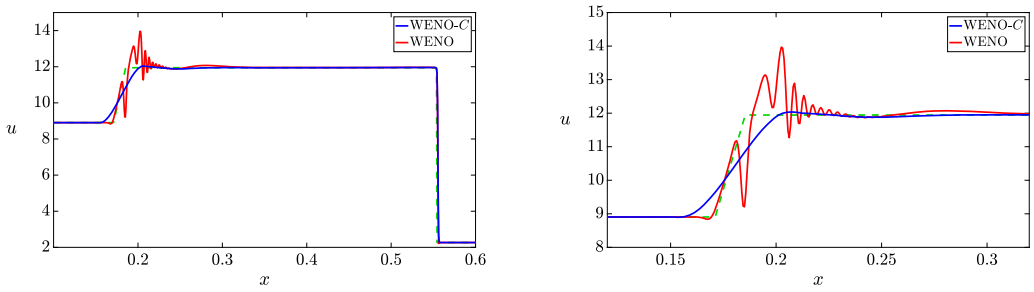
Scheme		Cells			
		361	721	1441	2881
WENO	Overshoot Error	7.330×10^{-2}	6.780×10^{-2}	6.200×10^{-2}	5.620×10^{-2}
	Order	–	0.113	0.129	0.142
WENO-*C*	Bump Error	1.500×10^{-2}	9.700×10^{-3}	6.600×10^{-3}	3.900×10^{-3}
	Order	–	0.629	0.556	0.759

Table 9
Post shock-wall collision ($t = 8.0$) L^1 error analysis and convergence tests for the velocity for the LeBlanc shock tube problem.

Scheme		Cells			
		361	721	1441	2881
WENO	Error	2.160×10^{-2}	9.832×10^{-3}	5.336×10^{-3}	2.896×10^{-3}
	Order	–	1.136	0.882	0.882
WENO-Noh	Error	1.528×10^{-2}	6.544×10^{-3}	3.407×10^{-3}	1.668×10^{-3}
	Order	–	1.224	0.942	1.030
WENO-*N*	Error	2.141×10^{-2}	9.684×10^{-3}	5.178×10^{-3}	2.793×10^{-3}
	Order	–	1.144	0.903	0.891
WENO-*C*	Error	5.703×10^{-3}	3.486×10^{-3}	2.024×10^{-3}	1.052×10^{-3}
	Order	–	0.710	0.785	0.944
WENO-*C*-*N*	Error	5.627×10^{-3}	3.384×10^{-3}	2.001×10^{-3}	1.045×10^{-3}
	Order	–	0.734	0.758	0.937
WENO-*C*-*W*	Error	6.077×10^{-3}	3.170×10^{-3}	1.694×10^{-3}	8.257×10^{-4}
	Order	–	0.939	0.904	1.037
WENO-*C*-*W*-*N*	Error	6.064×10^{-3}	3.143×10^{-3}	1.703×10^{-3}	8.363×10^{-4}
	Order	–	0.948	0.884	1.026

This "super-convergence" [19] is due to large errors on coarser meshes, rather than smaller errors on finer meshes, and is therefore superficial. On the other hand, the WENO-*C* and WENO-*C*-*N* schemes produce solutions with much smaller errors, and suggest close to first-order convergence.

Post shock-wall collision, the wall *C*-method produces a highly accurate non-oscillatory solution, while ensuring that the wall heating error does not occur. While the WENO-Noh scheme is able to suppress most of the oscillations, the large amount of viscosity needed due to the lack of smoothness of $|\partial_x u|$ results in a shock front that is too smeared, as well as the imposition of a smaller time-step. Again, we see that the noise indicator algorithm serves primarily as an error correction mechanism, removing small-scale high-frequency oscillations from the solution.



(a) $t = 0.0039$: velocity (b) $t = 0.0039$: velocity, zoomed in

Fig. 21. Comparison of WENO and WENO-C for the Peak shock tube problem with 801 cells. The green curve is the exact solution.

5.5. The Peak shock tube problem

We next consider the Peak shock tube problem, introduced in [25]. The domain of interest is $0.1 \le x \le 0.6$, the adiabatic gas constant is $\gamma = 1.4$, and the initial data is given by

$$\begin{bmatrix} \rho_0 \\ (\rho u)_0 \\ E_0 \end{bmatrix} = \begin{bmatrix} 0.1261192 \\ 11.1230540 \\ 1.962323 \times 10^3 \end{bmatrix} \mathbb{1}_{[0.1,0.5)}(x) + \begin{bmatrix} 6.591493 \\ 14.932505 \\ 24.800422 \end{bmatrix} \mathbb{1}_{[0.5,0.6]}(x).$$

The difficulty in simulating solutions to Peak is due to the fact that the shock wave moves significantly slower than the expansion wave; moreover, the distance between the contact discontinuity and the shock is very small, resulting in a sharp and narrow peak in the density. Most schemes produce inaccurate velocity profiles with large overshoots and low-frequency noise at the expansion wave [25,15].

The stand-alone WENO scheme produces a similarly poor velocity profile, but the C-method can be used to produce a good solution. Since the noise appears in the velocity profile in the region with the rarefaction wave, and since the usual C-method includes a compression switch so that artificial viscosity is active only in regions of compression, we employ an additional C-equation for the rarefaction wave, whose solution is $C^r(x,t)$. We consider the following modification to (13b):

$$\partial_t(\rho u) + \partial_x(\rho u^2 + p) = \partial_x\left(\rho\left(\tilde{\beta}^u C + \tilde{\beta}^r C^r\right)\partial_x u\right),$$
$$\partial_t C^r + \frac{S(\boldsymbol{u})}{\varepsilon\Delta x} C^r - \kappa\Delta x \cdot S(\boldsymbol{u})\partial_{xx} C^r = \frac{S(\boldsymbol{u})}{\varepsilon\Delta x} G^r,$$

where C is the solution to (13d), and where $\tilde{\beta}^u = \frac{\max_x|\partial_x u|}{\max_x C}\beta^u$ and $\tilde{\beta}^r = \frac{\max_x|\partial_x u|}{\max_x C^r}\beta^r$, with

$$G^r(x,t) = \mathbb{1}_{(0,+\infty)}(\partial_x u)\cdot\frac{|\partial_x u(x,t)|}{\max_x|\partial_x u(x,t)|}.$$

We remark that we have omitted the wall function $\overline{C}_w$ since we are not simulating the shock-wall collision for this problem.

WENO and WENO-C (with the above modification) are used on a grid with 801 cells and with a time-step $\Delta t \approx 3.55 \times 10^{-6}$, giving CFL $= 0.6$. The final time is $t = 0.0039$, and the results are shown in Fig. 21. The relevant parameters are chosen as

$$\beta^u = 1.0, \qquad \beta^r = 10.0, \qquad \varepsilon = 10.0, \qquad \kappa = 40.0, \qquad \varepsilon_r = 1.0, \qquad \kappa_r = 20.0.$$

As shown in Fig. 21, the extra viscosity provided by β^r at the rarefaction wave removes the large overshoot and low frequency non-physical oscillations that are present in the solution produced with WENO.

In [25], an error analysis of various schemes applied to the Peak shock tube problem with the above specifications is provided. We compute the L^1 and L^2 errors for the computed velocity minus the exact solution, using (31a) for the L^1 error and

$$||u(\cdot,t) - u^*(\cdot,t)||_{L^2} \approx \sqrt{\frac{1}{M}\sum_{i=1}^{M}|u(x_i,t) - u^*(x_i,t)|^2},$$

where M is the number of cells used in the simulation and u^* is the exact solution. Following [25], we list the errors in percentage form with the ratio in question given by $\frac{||u-u^*||}{||u^*||}$. We also list the smallest error computed from all the schemes considered in [25]; namely, the error computed from the scheme of Liu and Lax [27,28], which we will refer to as LL. We

Table 10
L^1 and L^2 error analysis for the velocity u for the Peak shock tube problem at time $t = 0.0039$.

Norm	Scheme		Cells 801
$\|u - u^*\|_{L^1}$	WENO	Error	1.057×10^{-1}
		%	1.0 %
	WENO-C	Error	7.260×10^{-2}
		%	0.7 %
	LL	Error	–
		%	0.8 %
$\|u - u^*\|_{L^2}$	WENO	Error	5.168×10^{-1}
		%	4.7 %
	WENO-C	Error	4.684×10^{-1}
		%	4.3 %
	LL	Error	–
		%	4.4 %

see in Table 10 that WENO-C compares very well with LL, with the solution producing smaller errors in both the L^1 and L^2 norms.

This test demonstrates the flexibility of the C-method; although a standard WENO scheme produces an inaccurate and oscillatory solution, a very simple modification of the C-method allows for the suppression of these oscillations, resulting in a more accurate solution.

5.6. The Osher-Shu shock tube problem

The Osher-Shu shock tube problem, introduced in [46], simulates a shock front, perturbed by sinusoidal fluctuations. The computational domain is $-1 \le x \le 1$, $\gamma = 1.4$, with initial data

$$\begin{bmatrix} \rho_0 \\ (\rho u)_0 \\ E_0 \end{bmatrix} = \begin{bmatrix} 3.857143 \\ 10.14185 \\ 39.1666 \end{bmatrix} \mathbb{1}_{[-1,-0.8)}(x) + \begin{bmatrix} 1 + 0.2\sin(5\pi x) \\ 0 \\ 2.5 \end{bmatrix} \mathbb{1}_{[-0.8,1]}(x)\,. \tag{37}$$

We employ free-flow boundary conditions (17) at the left wall $x = -1$, and solid wall boundary conditions (16) at the right wall $x = 1$.

5.6.1. Noise removal with the noise indicator

In order to test the efficacy of our noise detection and removal algorithm for the Osher-Shu test, we perform our numerical simulations using too large a time-step and hence a numerically unstable CFL number, which produces spurious high-frequency oscillation behind the shock.[6] Of course, high-frequency oscillations can be created by numerous numerical instabilities, but an unstable CFL number creates the prototypical oscillation pattern for testing a noise removal scheme.

Our goal is to remove the high-frequency noise from the solution without affecting the low-frequency sinusoidal oscillations that are the main feature of this test problem. To this end, we first compute a solution using WENO with 1025 cells with a time-step $\Delta t = 5.0 \times 10^{-4}$, giving a CFL number of 1.2.

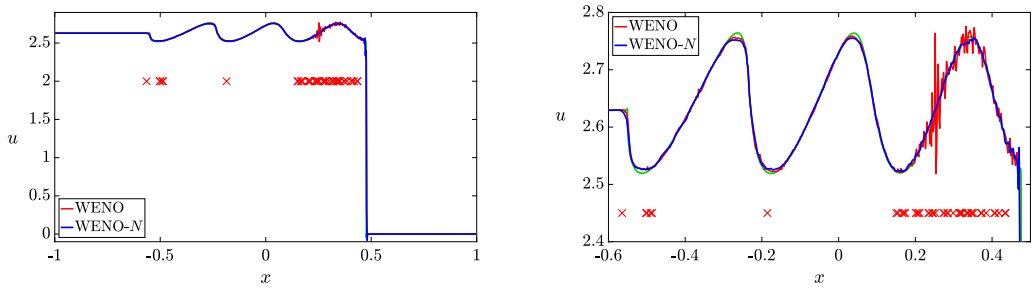
The relatively large number of cells and time-step produce noise with a frequency that is significantly higher than the lower frequency non-spurious oscillations present in the solution. The WENO-N scheme is used with the reference coefficient C_{ref} in (28) calculated using $\delta h = 10^{-3}$. The noise removal viscosity η is chosen such that $\eta \Delta\tau / \Delta x^2 = 0.25$ and only one time-step is taken in the heat equation. Since an exact solution is not available for this problem, our "exact" solution is computed with WENO using 8193 cells and a time-step of $\Delta t = 3.125 \times 10^{-5}$, so that CFL ≈ 0.6.

In Fig. 22, we compare the solutions computed with WENO and WENO-N. The noise indicator algorithm locates and removes the high-frequency noise present in the solution, without affecting the sinusoidal oscillations. The sharpness of the shock front remains unaffected with the use of the noise indicator, due to the deactivation of noise detection in a small region surrounding the shock.

For the purpose of benchmarking our noise detection and removal algorithm, we also conduct tests in which we use linear (hyperviscosity) operators (see [20,50,5,38,3,4]) of the form

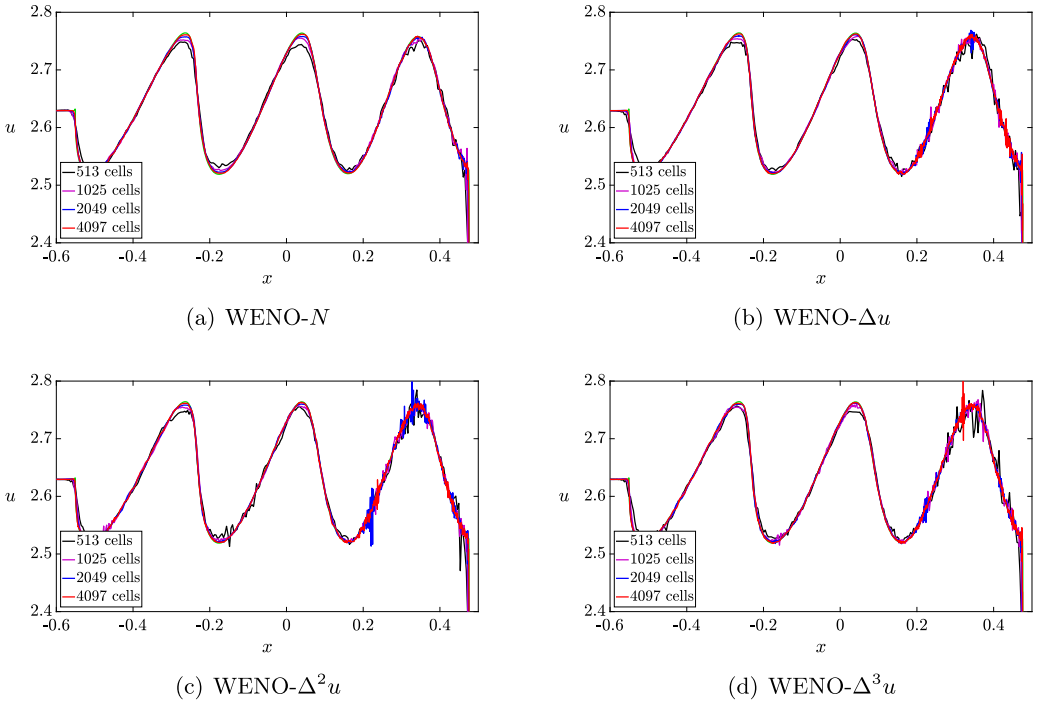
$$(-1)^{r-1} \beta_r (\Delta x)^{2r-1} \frac{\partial^{2r} u}{\partial x^{2r}} \tag{38}$$

[6] Artificially inflating the CFL number allows us to model a typical scenario in computational physics in which a DNS-type simulation requires a prohibitively small time-step, and forces simulations that require entering the unstable CFL regime. Our objective is to demonstrate that this high-frequency instability can be suppressed by use of our localized noise removal algorithm.



(a) $t = 0.36$: velocity (b) $t = 0.36$: velocity, zoomed in

Fig. 22. Comparison of WENO and WENO-N for the Osher-Shu problem with 1025 cells. The red crosses indicate where the noise indicator function $\mathbb{1}_{\text{noise}}(x)$ is active. The green curve is the "exact" solution.



(a) WENO-N (b) WENO-Δu

(c) WENO-$\Delta^2 u$ (d) WENO-$\Delta^3 u$

Fig. 23. Comparison of the velocity profiles at $t = 0.36$ for the Osher-Shu test. The green curve is the exact solution.

to remove noise, where $r \geq 1$. The equations of motion we consider are the Euler equations (4a) with the term (38) on the right-hand side of the momentum equation. When numerically approximated using our WENO-type discretization, the resulting scheme is referred to as the WENO-$\Delta^r u$ scheme. We perform numerical tests for the WENO-$\Delta^r u$ scheme with $r = 1, 2, 3$, and set $\beta_1 = 0.2$, $\beta_2 = 0.05$, and $\beta_3 = 0.01$, with these values determined *a posteriori* to optimize the resulting solutions.

We compare in Fig. 23 the WENO-N and WENO-$\Delta^r u$ simulations; each subfigure shows the computed velocity, obtained using one of the schemes on grids with 513, 1025, 2049, and 4097 cells, as well as the exact solution. The plots shown are zoomed in on the region behind the shock where there is high-frequency noise.

It is clear from these figures that, qualitatively, the WENO-N scheme produces solutions with minimal noise that appear to converge to the exact solution. The hyperviscosity schemes, on the other hand, produce solutions with erratic behavior; for instance, despite mesh refinement, the WENO-$\Delta^3 u$ solution on the 4097 cell grid appears much worse than the solutions on the 1025 and 2049 cell grids. Similarly inconsistent convergence behavior can be observed with WENO-Δu and WENO-$\Delta^2 u$. This is due to the CFL condition violation. It is interesting to observe, on the other hand, that the WENO-N solutions are not subject to this erratic convergence behavior. This is likely the result of the highly localized (in both space and time) nature of the noise detection. Overall, WENO-N appears to produce noise-free, accurate, and convergent solutions.

Table 11
L^1 and L^1_t error analysis and convergence tests for the velocity u for the Osher-Shu problem at time $t = 0.36$, with $\tilde{u} = u - u^*$ the difference between the computed solution u and the "exact solution" u^*.

Norm	Scheme		Cells			
			513	1025	2049	4097
$\|\tilde{u}\|_{L^1}$	WENO	Error	1.003×10^{-2}	5.478×10^{-3}	2.018×10^{-3}	1.258×10^{-3}
		Order	–	0.873	1.440	0.682
	WENO-Δu	Error	1.045×10^{-2}	4.717×10^{-3}	1.990×10^{-3}	9.770×10^{-4}
		Order	–	1.148	1.245	1.026
	WENO-$\Delta^2 u$	Error	1.050×10^{-2}	4.774×10^{-3}	2.459×10^{-3}	1.132×10^{-3}
		Order	–	1.137	0.957	1.119
	WENO-$\Delta^3 u$	Error	1.084×10^{-2}	4.806×10^{-3}	1.981×10^{-3}	1.109×10^{-3}
		Order	–	1.174	1.279	0.838
	WENO-N	Error	1.013×10^{-2}	4.432×10^{-3}	1.973×10^{-3}	1.005×10^{-3}
		Order	–	1.193	1.168	0.973
$\|\tilde{u}\|_{L^1_t}$	WENO	Error	7.328×10^{-3}	3.223×10^{-3}	1.139×10^{-3}	6.761×10^{-4}
		Order	–	1.185	1.501	0.752
	WENO-Δu	Error	7.484×10^{-3}	3.348×10^{-3}	1.192×10^{-3}	6.418×10^{-4}
		Order	–	1.161	1.490	0.893
	WENO-$\Delta^2 u$	Error	7.333×10^{-3}	3.316×10^{-3}	1.254×10^{-3}	7.255×10^{-4}
		Order	–	1.145	1.403	0.789
	WENO-$\Delta^3 u$	Error	7.419×10^{-3}	3.340×10^{-3}	1.196×10^{-3}	6.903×10^{-4}
		Order	–	1.151	1.482	0.793
	WENO-N	Error	7.066×10^{-3}	3.004×10^{-3}	1.050×10^{-3}	5.656×10^{-4}
		Order	–	1.234	1.517	0.893

Defining the L^1_t norm as

$$\|f\|_{L^1_t} := \int_0^t \int_{-1}^{+1} |f| \, \mathrm{d}x \, \mathrm{d}t \approx \frac{1}{KM} \sum_{j=1}^{K} \sum_{i=1}^{M} |f(x_i, t_j)| \,,$$

in Table 11 we compute the L^1 and L^1_t errors for the velocity at time $t = 0.36$ at various mesh refinements. Once again, we see that the noise indicator algorithm functions as an "error correcter", reducing the numerical error through the removal of high-frequency noise, while maintaining a relatively high order of accuracy. Among all the schemes considered, WENO-N produces solutions with the smallest errors, providing a quantitative validation of the observations made from Fig. 23.

We note that the numerical error and the order of convergence remains unchanged when using the density ρ instead of the velocity u; in particular, the WENO-N algorithm produces solutions with smaller errors and similar rates of convergence as WENO, when errors and accuracy are computed using ρ. And so, the removal of high-frequency noise in u, in turn, provides a density field that is also free of high-frequency oscillations (cf. Remark 1).

5.6.2. Stabilizing shock-wall collision for Osher-Shu

We now turn to the issue of stabilizing shock-wall collision for the Osher-Shu problem. The problem is set up as follows: the initial data is (37), the time-step is given by $\Delta t = 5 \times 10^{-4}$ with final time $t = 0.63$, and the number of cells is 512, so that the CFL number is 0.6. We impose the solid wall boundary conditions (23) at the right boundary $x = 1$, and free-flow boundary conditions (22) at the left boundary $x = -1$. The shock-wave moves to the right and collides with the wall at $x = 1$ at time $t \approx 0.5$. Post-collision, there is a large amount of noise present in the solution behind the shock-wave, and our aim is to remove the noise while preserving the sharpness of the shock front and minimizing the damping of the post-shock low frequency oscillations.

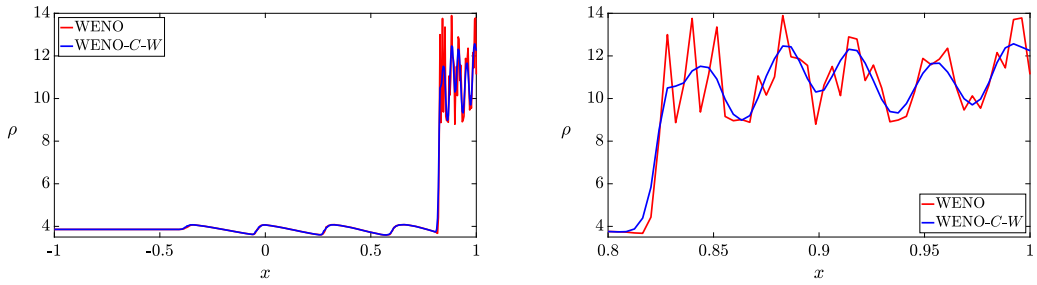
We employ our WENO-C-W scheme and choose the relevant parameters as

$$\beta^u = 1.0, \qquad \beta^E = 0.0, \qquad \beta^u_w = 2.5, \qquad \beta^E_w = 0.85$$
$$\varepsilon = 1.0, \qquad \kappa = 5.0, \qquad \varepsilon_w = 40.0, \qquad \kappa_w = 4.0 \,.$$

The results are shown in Fig. 24. Post shock-wall collision, WENO produces a noisy solution with high frequency noise interfering with the sinusoidal oscillations, while WENO-C-W produces a solution with a sharp front and without noise.

6. Concluding remarks

In this paper, we have presented three ideas: the first is a space-time smooth artificial viscosity method that is versatile and simple to implement; the second is a shock-wall collision scheme that can be used to suppress post-collision noise that occurs when a shock-wave collides with a fixed boundary and bounces-back; the third is a wavelet-based noise detection and removal scheme that is highly localized and can be used to remove noise present in solutions. We have demonstrated



(a) $t = 0.63$: density (b) $t = 0.63$: density, zoomed in

Fig. 24. Comparison of WENO vs. WENO-C-W for the density just after the shock-wall collision problem for the Osher-Shu problem with 512 cells.

the efficacy of the new method on a variety of 1-D test problems with different features, and demonstrated that the solutions produced retain sharp fronts, correct wave speeds, remain oscillation-free, are not subject to the wall-heating error, and maintain high-order accuracy.

Acknowledgements

Research reported in this publication was supported by the Office of Defense Nuclear Nonproliferation Research and Development and by the Defense Threat Reduction Agency under Interagency Agreement number HDTRA1825370 (DTRA10027 – 25370) as work for others. SS was partially supported by DTRA HDTRA11810022.

We would like to express our gratitude to the anonymous referees for their numerous suggestions that have greatly improved the manuscript.

Appendix A. The WENO-$|u_x|$ and WENO-Noh schemes

For our numerical simulations, we use a variety of combinations of the WENO scheme, the C-method, the wall C-method, and the noise indicator. For the purpose of comparison, we implement two additional methods. The first is a classical artificial viscosity scheme, WENO-$|u_x|$, and the second is WENO-Noh, an artificial viscosity method introduced by Noh [37]. We will employ WENO-Noh primarily as a comparison for the wall C-method for shock-wall collision, while WENO-$|u_x|$ will serve as a benchmark for the usual C-method.

A.1. WENO-$|u_x|$: classical artificial viscosity

This is the classical artificial viscosity scheme, where viscosity is only added to the momentum equation and the localizing coefficient is given by $|\partial_x u|$. More precisely, we implement the method in the following manner:

$$\partial_t \rho + \partial_x(\rho u) = 0, \tag{39a}$$

$$\partial_t(\rho u) + \partial_x(\rho u^2 + p) = \partial_x\left((\Delta x)^2 \beta^u \, \rho \, |\partial_x u| \, \partial_x u\right), \tag{39b}$$

$$\partial_t E + \partial_x(uE + up) = 0\,. \tag{39c}$$

The time and spatial discretizations are done in as in §3.3. This scheme will serve primarily as a benchmark for WENO-C.

A.2. WENO-Noh: an artificial viscosity method of Noh

This artificial viscosity scheme of Noh [37] introduces an additional heat conduction term to the energy equation, in addition to the usual viscosity term in the momentum equation. For more details, we refer the reader to [37]. We implement the method in the following fashion: the equations of motion are

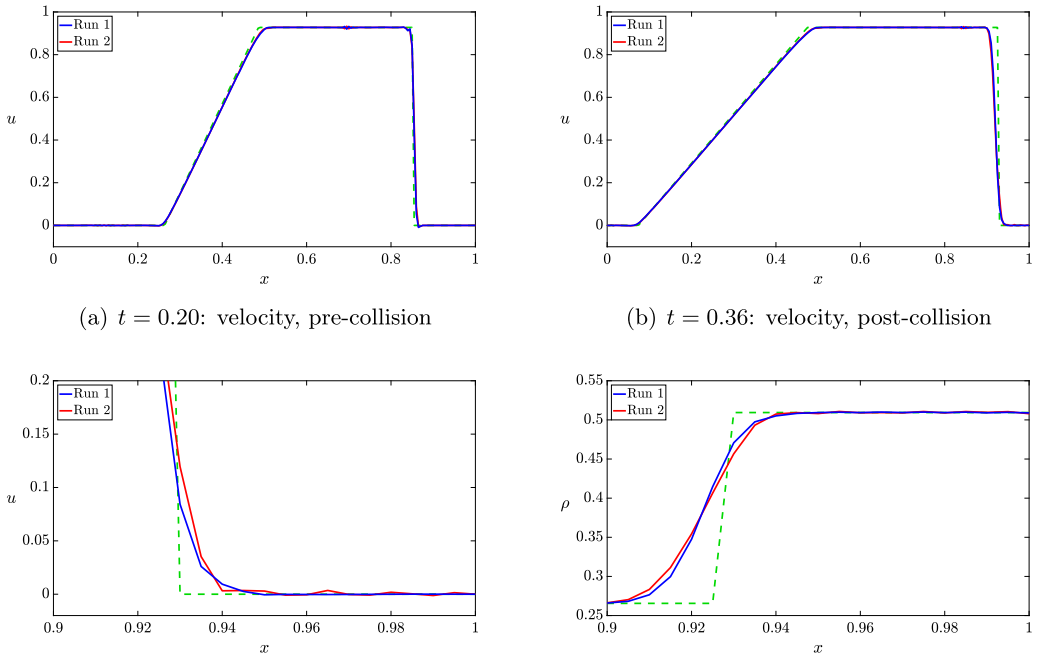
$$\partial_t \rho + \partial_x(\rho u) = 0, \tag{40a}$$

$$\partial_t(\rho u) + \partial_x(\rho u^2 + p) = \partial_x\left((\Delta x)^2 \beta^u_{\text{Noh}} \, \rho \, |\partial_x u| \, \partial_x u\right), \tag{40b}$$

$$\partial_t E + \partial_x(uE + up) = \partial_x\left((\Delta x)^2 \beta^E_{\text{Noh}} \, \rho \, |\partial_x u| \, \partial_x e\right). \tag{40c}$$

Table 12
Various numerical schemes used in the simulations.

Scheme	Description
WENO	standard fifth-order WENO procedure for the usual Euler equations.
WENO-$\|u_x\|$	WENO scheme with classical artificial viscosity.
WENO-Noh	WENO scheme with Noh's artificial viscosity method.
WENO-C	WENO scheme with the C-method.
WENO-C-W	WENO scheme with the C-method and the wall C-method outlined in §3.
WENO-N	WENO scheme with the noise indicator outlined in §4.
WENO-C-N	WENO scheme with the C-method and the noise indicator.
WENO-W-N	WENO scheme with the wall C-method and the noise indicator.
WENO-C-W-N	WENO scheme with the C-method, the wall C-method and the noise indicator.
WENO-$\Delta^r u$	WENO scheme with linear hyperviscosity (38).



(a) $t = 0.20$: velocity, pre-collision

(b) $t = 0.36$: velocity, post-collision

(c) $t = 0.36$: velocity, post-collision zoom-in

(d) $t = 0.36$: density, post-collision zoom-in

Fig. 25. Comparison of the optimized-parameter and fixed-parameter WENO-C-W runs for the Sod shock tube problem before and after shock-wall collision. The artificial viscosity parameters for the fixed-parameter Run 2 are chosen as $\beta^u = 1.0$, $\beta^E = 0.0$, $\beta^u_w = 5.0$, $\beta^E_w = 10.0$.

Here, $e = \frac{p}{(\gamma-1)\rho}$ is the specific internal energy of the system. The numerical discretization is then done in an identical fashion to that described in §3.3. We will employ this scheme with the aim of fully suppressing post-collision noise, even at the expense of a less accurate solution prior to shock-wall collision.

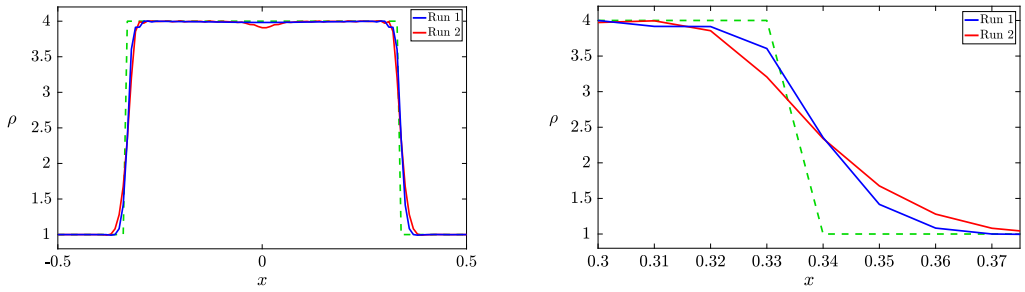
For readability, we will use Table 12 to refer to these various schemes.

Appendix B. Calculation of the exact solution post shock-wall collision

In this section, we provide details for the calculation of the exact solution to Sod-type problem post shock-wall collision. The solution, calculated based on the Rankine-Hugoniot conditions (10) and the assumption that the post-shock velocity is identically zero both pre and post shock-wall collision, is valid until the reflected shock front collides with the contact wave. We assume for simplicity that the shock front is traveling to the right (so that the shock speed $\dot{\sigma}$ satisfies $\dot{\sigma}(t) > 0$ pre shock-wall collision) and collides with, and reflects back off of, the right boundary. We also assume that the shock $\sigma(t)$ separates two constant states, $\boldsymbol{u}_l$ and $\boldsymbol{u}_r$, to the left and right of the shock, respectively.

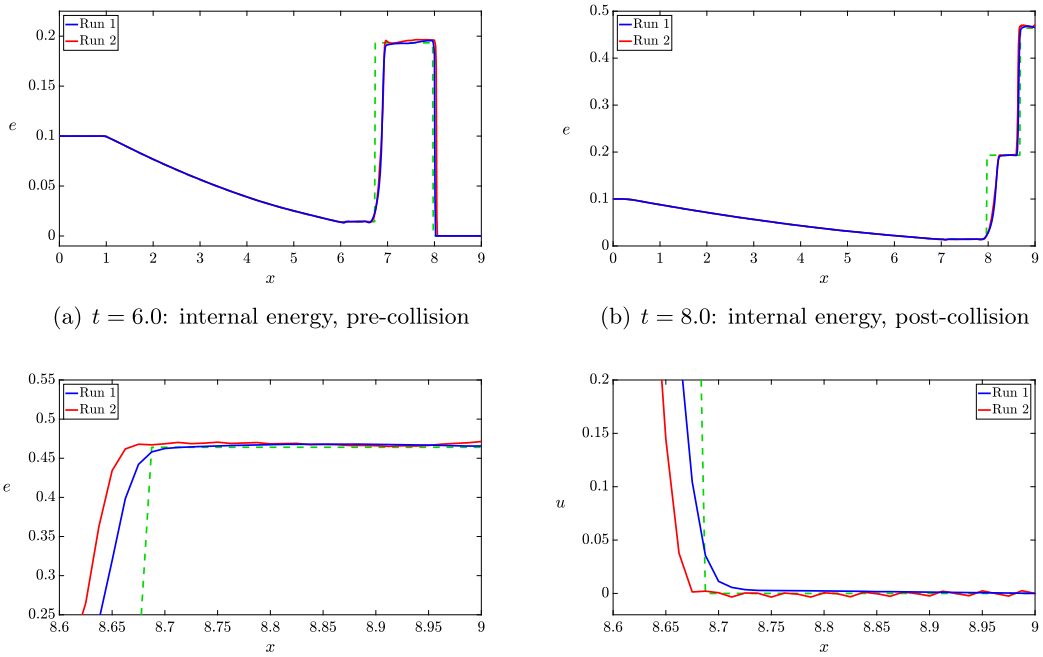
The left states $\boldsymbol{u}_l$ and the post-shock velocity $u_r = 0$ are all known; the unknowns are thus the post-shock density ρ_r, energy E_r (or, equivalently, pressure p_r), and shock speed $\dot{\sigma}(t)$. The R-H conditions (10b) and (10a) yield

$$p_r = \rho_l u_l^2 + p_l - \dot{\sigma}\,\rho_l u_l\,, \tag{41}$$



(a) $t = 1.0$: density (b) $t = 1.0$: density zoom-in at shock

Fig. 26. Comparison of the optimized-parameter and fixed-parameter WENO-*C* runs for the Noh problem. The artificial viscosity parameters for the fixed-parameter Run 2 are chosen as $\beta^u = 3.0$, $\beta^E = 30.0$.



(a) $t = 6.0$: internal energy, pre-collision (b) $t = 8.0$: internal energy, post-collision

(c) $t = 8.0$: internal energy, post-collision zoom-in (d) $t = 8.0$: velocity, post-collision

Fig. 27. Comparison of the optimized-parameter and fixed-parameter WENO-*C*-*W* runs for the LeBlanc shock tube problem before and after shock-wall collision. The artificial viscosity parameters for the fixed-parameter Run 2 are chosen as $\beta^u = 0.001$, $\beta^E = 0.0$, $\beta^e = 0.5$, $\beta^u_w = 4.0$, $\beta^E_w = 15.0$, $\beta^e_w = 0.0$.

$$\rho_r = \rho_l - \frac{\rho_l u_l}{\dot{\sigma}}\,, \tag{42}$$

respectively, so it only remains to calculate the shock speed $\dot{\sigma}$. Substituting (41) into (10c) and simplifying leads to a quadratic equation for $\dot{\sigma}$,

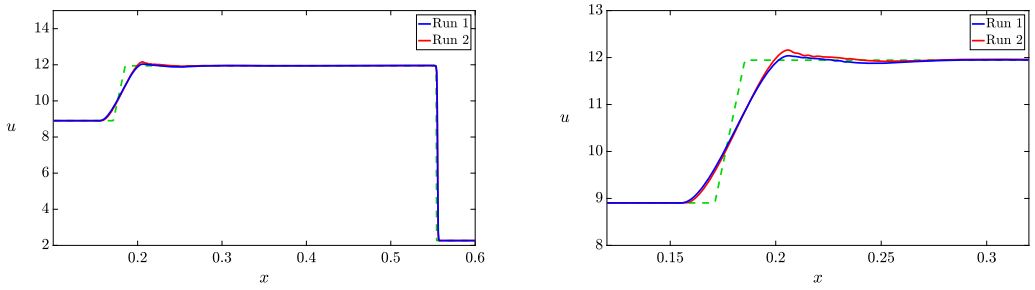
$$\dot{\sigma}^2 - \frac{1}{2}(3-\gamma)u_l\dot{\sigma} - \frac{(\gamma-1)(E_l+p_l)}{\rho_l} = 0\,,$$

which has solutions

$$\dot{\sigma} = \frac{1}{4}(3-\gamma)u_l \pm \sqrt{\left(\frac{(3-\gamma)u_l}{4}\right)^2 + \frac{(\gamma-1)(E_l+p_l)}{\rho_l}}\,.$$

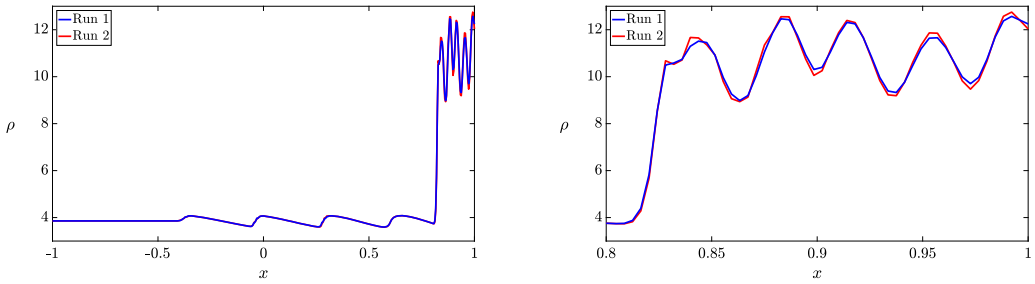
We take the negative root $\dot{\sigma} < 0$, since the shock moves to the left post shock-wall collision:

$$\dot{\sigma} = \frac{1}{4}(3-\gamma)u_l - \sqrt{\left(\frac{(3-\gamma)u_l}{4}\right)^2 + \frac{(\gamma-1)(E_l+p_l)}{\rho_l}}\,. \tag{43}$$



(a) $t = 0.0039$: velocity (b) $t = 0.0039$: velocity zoom-in

Fig. 28. Comparison of the optimized-parameter and fixed-parameter WENO-C runs for the Peak shock tube problem. The artificial viscosity parameters for the fixed-parameter Run 2 are chosen as $\beta^u = 1.0$, $\beta^E = 0.0$, $\beta^r = 10.0$.



(a) $t = 0.63$: density, post-collision (b) $t = 0.63$: density, post-collision zoom-in

Fig. 29. Comparison of the optimized-parameter and fixed-parameter WENO-C-W runs for the Osher-Shu shock tube problem after shock-wall collision. The artificial viscosity parameters for the fixed-parameter Run 2 are chosen as $\beta^u = 1.0$, $\beta^E = 0.0$, $\beta^u_w = 2.5$, $\beta^E_w = 0.85$.

The relations (41), (42), and (43) then provide the complete exact solution post shock-wall collision, up until the time that the reflected shock front collides with the contact discontinuity.

Appendix C. Comparison of optimized-parameter runs with fixed-parameter runs

In this section, we compare the optimized-parameter runs presented in §5, with a set of runs using fixed parameters. The fixed-parameter runs have the C-equation parameters ε and κ take the fixed value 1, with the exception of ε_w, to which we assign the fixed value $\varepsilon_w = 50.0$. The artificial viscosity parameters β are still free to choose, and so vary from problem to problem. The particular choices of β for each test problem shown in Figs. 25–29 are listed in the corresponding caption. For the initial data, we refer the reader to the relevant section in the main body of the paper.

We present results for the Sod shock-wall collision, Noh, LeBlanc shock-wall collision, Peak, and Osher-Shu shock-wall collision problems. In Figs. 25–29, Run 1 indicates the solution computed using the optimized set of parameters, while Run 2 indicates the solution computed using the fixed set of parameters.

References

[1] R.A. Alpher, R.J. Rubin, Normal reflection of shock waves from moving boundaries, J. Appl. Phys. 25 (3) (1954) 395–399.
[2] G.E. Barter, D.L. Darmofal, Shock capturing with PDE-based artificial viscosity for DGFEM: Part I, formulation, J. Comput. Phys. 229 (5) (2010) 1810–1827.
[3] A.W. Cook, W.H. Cabot, A high-wavenumber viscosity for high-resolution numerical methods, J. Comput. Phys. 195 (2) (2004) 594–601.
[4] A.W. Cook, W.H. Cabot, Hyperviscosity for shock-turbulence interactions, J. Comput. Phys. 203 (2004) 379.
[5] E.J. Caramana, M.J. Shashkov, P.P. Whalen, Formulations of artificial viscosity for multi-dimensional shock wave computations, J. Comput. Phys. 144 (1) (1998) 70–97.
[6] R.R. Coifman, D.L. Donoho, Translation-Invariant De-Noising, Springer New York, New York, NY, 1995, pp. 125–150.
[7] P. Colella, A direct Eulerian MUSCL scheme for gas dynamics, SIAM J. Sci. Stat. Comput. 6 (1985) 104–117.
[8] P. Colella, P.R. Woodward, The numerical simulation of two-dimensional fluid flow with strong shocks, J. Comput. Phys. 54 (1984) 115–173.
[9] P. Colella, P.R. Woodward, The piecewise parabolic method (PPM) for gas dynamical simulations, J. Comput. Phys. 54 (1984) 174–201.
[10] R. Courant, K.O. Friedrichs, Supersonic Flow and Shock Waves, Applied Mathematical Sciences, Springer, New York, 1999.
[11] R.J. DiPerna, Convergence of the viscosity method for isentropic gas dynamics, Commun. Math. Phys. 91 (1) (Mar 1983) 1–30.
[12] R. Donat, A. Marquina, Capturing shock reflections: an improved flux formula, J. Comput. Phys. 125 (1) (1996) 42–58.
[13] M. Farge, Wavelet Transforms and Their Applications to Turbulence, Annu. Rev. Fluid Mech., vol. 24, Annual Reviews, Palo Alto, CA, 1992, pp. 395–457.
[14] R.A. Gentry, R.E. Martin, B.J. Daly, An Eulerian differencing method for unsteady compressible flow problems, J. Comput. Phys. 1 (1966) 87–118.

[15] J.A. Greenough, W.J. Rider, A quantitative comparison of numerical methods for the compressible Euler equations: fifth-order weno and piecewise-linear Godunov, J. Comput. Phys. 196 (1) (May 2004) 259–281.
[16] A. Harten, B. Engquist, S. Osher, S.R. Chakravarthy, Uniformly high-order accurate essentially non-oscillatory schemes, III, J. Comput. Phys. 71 (1987) 231–303.
[17] H.T. Huynh, Accurate upwind methods for the Euler equations, SIAM J. Numer. Anal. 32 (1995) 1565–1619.
[18] O. Igra, G. Ben-Dor, G. Mazor, M. Mond, Head-on collision between normal shock waves and a rubber-supported plate, a parametric study, Shock Waves 2 (3) (Sep 1992) 189–200.
[19] G.-S. Jiang, C.-W. Shu, Efficient implementation of weighted ENO schemes, J. Comput. Phys. 126 (1) (June 1996) 202–228.
[20] R. Landshoff, A Numerical Method for Treating Fluid Flow in the Presence of Shocks, Los Alamos National Laboratory Report, LA-1930, 1955, p. 1.
[21] A. Lapidus, A detached shock calculation by second-order finite differences, J. Comput. Phys. 2 (1967) 154–177.
[22] Andrew Majda, Stanley Osher, Propagation of error into regions of smoothness for accurate difference approximations to hyperbolic equations, Commun. Pure Appl. Math. 30 (6) (1977) 671–705.
[23] B. Van Leer, Towards the ultimate conservative difference scheme. V. A second-order sequel to Godunov's method, J. Comput. Phys. 32 (1979) 101–136.
[24] R.J. LeVeque, Finite Volume Methods for Hyperbolic Problems, Cambridge Texts in Applied Mathematics, Cambridge University Press, 2002.
[25] R. Liska, B. Wendroff, Comparison of several difference schemes on 1D and 2D test problems for the Euler equations, SIAM J. Sci. Comput. 25 (2003) 995–1017.
[26] W. Liu, J. Cheng, C.-W. Shu, High-order conservative Lagrangian schemes with Lax-Wendroff type time discretization for the compressible Euler equations, J. Comput. Phys. 228 (2009) 8872–8891.
[27] X.-D. Liu, P.D. Lax, Solution of two-dimensional Riemann problems of gas dynamics by positive schemes, SIAM J. Sci. Comput. 19 (2) (1998) 319–340.
[28] X.-D. Liu, P.D. Lax, Positive schemes for solving multi-dimensional hyperbolic systems of conservation laws ii, J. Comput. Phys. 187 (2) (2003) 428–440.
[29] X.-D. Liu, S. Osher, T. Chan, Weighted essentially non-oscillatory schemes, J. Comput. Phys. 115 (1994) 200–212.
[30] R. Loubère, M. Shashkov, A subcell remapping method on staggered polygonal grids for arbitrary-Lagrangian-Eulerian methods, J. Comput. Phys. 209 (2005) 105–138.
[31] L.G. Margolin, The reality of artificial viscosity, Shock Waves 29 (1) (2019) 27–35.
[32] A.E. Mattsson, W.J. Rider, Artificial viscosity: back to the basics, Int. J. Numer. Methods Fluids 77 (7) (2015) 400–417.
[33] G. Mazor, O. Igra, G. Ben-Dor, M. Mond, H. Reichenbach, F.T. Smith, Head-on collision of normal shock waves with a rubber-supported wall, Philos. Trans. R. Soc. Lond. A 338 (1650) (1992) 237–269.
[34] C. Meneveau, Analysis of turbulence in the orthonormal wavelet representation, J. Fluid Mech. 232 (1991) 469–520.
[35] R. Menikoff, Errors when shock waves interact due to numerical shock width, SIAM J. Sci. Comput. 15 (5) (1994) 1227–1242.
[36] R.F. Meyer, The impact of a shock wave on a movable wall, J. Fluid Mech. 3 (1957) 309–323.
[37] W.F. Noh, Errors for calculations of strong shocks using an artificial viscosity and an artificial heat flux, J. Comput. Phys. 72 (1) (1987) 78–120.
[38] T. Passot, A. Pouquet, Hyperviscosity for compressible flows using spectral methods, J. Comput. Phys. 75 (2) (1988) 300–313.
[39] J.J. Quirk, A contribution to the great Riemann solver debate, Int. J. Numer. Methods Fluids 18 (1994) 555–574.
[40] J. Reisner, J. Serencsa, S. Shkoller, A space-time smooth artificial viscosity method for nonlinear conservation laws, J. Comput. Phys. 235 (2013) 912–933.
[41] R. Ramani, J. Reisner, S. Shkoller, A space-time smooth artificial viscosity method with wavelet noise indicator and shock collision scheme, Part 2: the 2-D case, J. Comput. Phys. (2019), https://doi.org/10.1016/j.jcp.2019.02.048.
[42] R. Ramani, J. Reisner, S. Shkoller, A space-time smooth artificial viscosity method for shock-shock and shock-contact collision, In preparation.
[43] W.J. Rider, Revisiting wall heating, J. Comput. Phys. 162 (2) (2000) 395–410.
[44] K. Schneider, O.V. Vasilyev, Wavelet methods in computational fluid dynamics, in: Annual Review of Fluid Mechanics, in: Annu. Rev. Fluid Mech., vol. 42, Annual Reviews, Palo Alto, CA, 2010, pp. 473–503.
[45] C.-W. Shu, S. Osher, Efficient implementation of essentially non-oscillatory shock-capturing schemes, J. Comput. Phys. 77 (1988) 439–471.
[46] C.-W. Shu, S. Osher, Efficient implementation of essentially nonoscillatory shock-capturing schemes. II, J. Comput. Phys. 83 (1) (1989) 32–78.
[47] C.-W. Shu, High-order finite difference and finite volume weno schemes and discontinuous Galerkin methods for cfd, Int. J. Comput. Fluid Dyn. 17 (2) (2003) 107–118.
[48] E.F. Toro, Riemann Solvers and Numerical Methods for Fluid Dynamics, Springer-Verlag, Berlin, Heidelberg, 2009.
[49] J. Von Neumann, R.D. Richtmyer, A method for the numerical calculation of hydrodynamic shocks, J. Appl. Phys. 21 (1950) 232–237.
[50] M.L. Wilkins, Use of artificial viscosity in multidimensional fluid dynamic calculations, J. Comput. Phys. 36 (3) (1980) 281–303.

Journal of Computational Physics 387 (2019) 45–80

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

A space-time smooth artificial viscosity method with wavelet noise indicator and shock collision scheme, Part 2: The 2-*D* case

Raaghav Ramani [a], Jon Reisner [b], Steve Shkoller [a,*]

[a] Department of Mathematics, University of California, Davis, CA 95616, USA
[b] Los Alamos National Lab, XCP-4 MSF605 Los Alamos, NM 87544, USA

ARTICLE INFO

Article history:
Received 22 June 2018
Received in revised form 26 February 2019
Accepted 28 February 2019
Available online 7 March 2019

Keywords:
Gas dynamics
Shocks and contacts
Euler equations
Artificial viscosity
Shock collision
Noise indicator

ABSTRACT

This is the second part to our companion paper [18]. Herein, we generalize to two space dimensions the *C*-method developed in [20,18] for adding localized, space-time smooth artificial viscosity to nonlinear systems of conservation laws that propagate shock waves, rarefaction waves, and contact discontinuities. For gas dynamics, the *C*-method couples the Euler equations to scalar reaction-diffusion equations, which we call *C*-equations, whose solutions serve as space-time smooth artificial viscosity indicators for shocks and contacts. We develop a high-order numerical algorithm for gas dynamics in 2-*D* which can accurately simulate the Rayleigh-Taylor (RT) instability with Kelvin-Helmholtz (KH) roll-up of the contact discontinuity, as well as shock collision and bounce-back. Solutions to our *C*-equations not only indicate the location of the shocks and contacts, but also track the geometry of the evolving fronts. This allows us to implement both directionally isotropic and anisotropic artificial viscosity schemes, the latter adding diffusion only in directions tangential to the evolving front. We additionally produce a novel shock collision indicator function, which naturally activates during shock collision, and then smoothly deactivates. Moreover, we implement a high-frequency 2-*D* wavelet-based noise detector together with an efficient and localized noise removal algorithm.
To test the methodology, we use a highly simplified WENO-based discretization scheme. We provide numerical results for some classical 2-*D* test problems, including the RT problem, the Noh problem, a circular explosion problem from the Liska & Wendroff [13] review paper, the Sedov blast wave problem, the double Mach 10 reflection test, and a shock-wall collision problem. In particular, we show that our artificial viscosity method can eliminate the wall-heating phenomenon for the Noh problem, and thereby produce an accurate, non-oscillatory solution, even though our simplified WENO-type scheme fails to run for this problem.

1. Introduction

This is the second in a two-part series of papers, in which we develop a high-order numerical algorithm, the *C*-method, to simulate compressible fluid flow with shock waves and contact discontinuities, as well as shock-wall collision and bounce-

* Corresponding author.
E-mail addresses: rramani@math.ucdavis.edu (R. Ramani), reisner@lanl.gov (J. Reisner), shkoller@math.ucdavis.edu (S. Shkoller).

https://doi.org/10.1016/j.jcp.2019.02.048
0021-9991/

back. In the first part [18], we developed our scheme in one space dimension. This second part is devoted to the more geometric problem of simulating shocks, contacts, and collisions in two space dimensions.

Compressible fluid flow is an example of a system of nonlinear conservation laws, which in two space dimensions, can be written as

$$\partial_t \boldsymbol{U}(\boldsymbol{x}, t) + \partial_x \boldsymbol{F}(\boldsymbol{U}(\boldsymbol{x}, t)) + \partial_y \boldsymbol{G}(\boldsymbol{U}(\boldsymbol{x}, t)) = \boldsymbol{0}\,,$$
$$\boldsymbol{U}(\boldsymbol{x}, t = 0) = \boldsymbol{U}_0(\boldsymbol{x})\,,$$

where $\boldsymbol{x} = (x, y)$ denotes coordinates on the plane, t denotes time, $\boldsymbol{U}(\boldsymbol{x}, t)$ is a vector of conserved quantities, and $\boldsymbol{F}(\boldsymbol{U})$ and $\boldsymbol{G}(\boldsymbol{U})$ denote the horizontal and vertical flux functions, respectively.

Even with smooth initial conditions, multi-dimensional conservation laws such as the compressible Euler equations develop singularities in finite-time [27,4] and, in general, solutions consist of propagating shock waves and contact discontinuities. In two space dimensions, shocks and contacts produce *curves of discontinuities*, also known as *fronts*, which propagate according to the *Rankine-Hugoniot* conditions (see §2). The objective of a high-order numerical scheme is to produce a simulation which keeps the fronts sharp, while simultaneously providing high-order accuracy in smooth regions away from the front.

1.1. Numerical discretization

In §1.1 of [18], we described the tools necessary for designing high-order accurate numerical schemes in 1-D. In multi-D, similar tools are required to obtain non-oscillatory numerical schemes, but the multi-dimensional analogues are generally limited by mesh considerations. For structured grids (such as products of uniform 1-D grids), dimensional splitting is commonly used, decomposing the problem into a sequence of 1-D problems. This technique is quite successful, but stringent mesh requirements prohibits its use on complex domains. Moreover, applications to PDE outside of variants of the Euler equations may be somewhat limited. For further discussion of the limitations of dimensional splitting, we refer the reader to Crandall & Majda [6], and Jiang & Tadmor [9]. For unstructured grids, dimensional splitting is not available and alternative approaches must be employed, necessitated by the lack of multi-D Riemann solvers. WENO schemes on unstructured triangular grids have been developed in Hu & Shu [7], but using simplified methods, which employ reduced characteristic decompositions, can lead to a loss of monotonicity and stability.

Algorithms that explicitly introduce diffusion provide a simple way to stabilize higher-order numerical schemes and subsequently remove non-physical oscillations near shocks. We refer the reader to the introductory sections in [20,18] for a review of the classical artificial viscosity method [36].

In this paper, we develop a stable high-order 2-D numerical scheme that does not use approximate Riemann solvers or characteristic decompositions, but instead relies upon a 2-D generalization of the 1-D C-method [20,18]. The extensive error analysis and convergence tests performed for the 1-D C-method in [18] indicate that the scheme is high order accurate. We expect that the 2-D implementation of the method presented in this paper is similarly high-order accurate, and we perform a number of numerical experiments to qualitatively support this claim. We also present error analysis and convergence studies for the Sedov problem to provide quantitative evidence for the claim of high order accuracy; further extensive convergence tests will be presented in [19].

1.2. Localization to shocks and contacts and capturing the geometry of the front

As we noted above, part one [18] provides a self-contained development of the 1-D C-method; for problems in one space dimension, the function $C(x, t)$ is the solution to a forced scalar reaction-diffusion equation, and serves as a highly localized space-time smooth indicator for the shock location. In 2-D, we again solve a scalar reaction-diffusion equation for $C(\boldsymbol{x}, t)$ on the plane; in this instance, the function $C(\boldsymbol{x}, t)$ not only serves as a localized indicator for both shocks and contacts, but is also able to accurately represent the geometry of the evolving fronts. Having this geometry allows us to define time-dependent normal and tangent vectors to the fronts, which, in turn, enables the construction of artificial viscosity methods that add diffusion only in certain directions rather than uniformly across the mesh. As we shall explain below, this type of anisotropic artificial viscosity scheme is essential in accurately capturing Kelvin-Helmholtz roll-up without overly diffusing the mixing regions in such flows.

1.3. Shock collision and wavelet-based noise removal

In part one [18], we consider the difficult problem of shock-wall collision and bounce-back in 1-D. In particular, when a shock wave collides with and reflects off of a fixed boundary, spurious oscillations develop behind the reflected shock. In addition to these post-collision oscillations, most schemes produce solutions that exhibit the phenomenon of anomalous *wall heating* [21]. A novel modification to the C-method in 1-D [18], wherein the time-dependent artificial viscosity parameter $\mathcal{B}(t)$ naturally increases during shock-wall collision and bounce-back, allows for the addition of extra "wall viscosity" to the shock front. This suppresses post-collision noise and the wall heating error, and ensures that the solution retains high-order

accuracy away from the shock and prior to collision. In §5 of this paper, we present a generalization of the 1-*D* shock collision scheme to the 2-*D* setting, and apply the method to a circular explosion problem in §10.2.

The occurrence of high frequency noise in numerical solutions to PDE is a well-known issue. The first part of this work [18] provides a description of an efficient wavelet-based noise detection and heat equation-based noise removal algorithm for 1-*D* gas dynamics simulations. Error analysis and convergence tests in [18] show that the noise detection and removal algorithm decreases the errors of numerical solutions and improves the rate of convergence. In §6 of this paper, we present the natural extension of this 1-*D* algorithm to the 2-*D* case, with applications to the RT problem in §12 and the Noh problem in §8.

1.4. Outline of the paper

In §2, we introduce the 2-*D* compressible Euler equations and the Rankine-Hugoniot jump conditions relating the speed of propagation of curves of discontinuity with the jump discontinuities in the conservative variables. In §3, we present the *isotropic C*-method, and in §4, the *anisotropic C*-method, the latter designed for the long-time evolution of contact discontinuities. A new 2-*D* shock-wall collision scheme is introduced in §5, and a wavelet-based noise detection and *localized* heat equation-based noise removal procedure is developed in §6. Details about the numerical methods implemented are provided in §7 and Appendix A. We then apply the methods to a number of test problems. In particular, the *C*-method produces an accurate, non-oscillatory solution for the difficult Noh problem, whereas our simplified WENO-type algorithm (as well as more sophisticated advection schemes) fails to run for this problem.

2. The compressible Euler equations in 2-*D*

The compressible Euler equations in a 2-*D* rectangular domain $\Omega = (x_1, x_M) \times (y_1, y_N) \subset \mathbb{R}^2$ and a time interval $[0, T]$ are given in conservation law form as

$$\partial_t \boldsymbol{U}(\boldsymbol{x}, t) + \partial_x \boldsymbol{F}(\boldsymbol{U}(\boldsymbol{x}, t)) + \partial_y \boldsymbol{G}(\boldsymbol{U}(\boldsymbol{x}, t)) = \boldsymbol{0}, \quad \boldsymbol{x} \in \Omega\,, t > 0, \tag{1a}$$

$$\boldsymbol{U}(\boldsymbol{x}, 0) = \boldsymbol{U}_0(\boldsymbol{x}), \quad \boldsymbol{x} \in \Omega\,, t = 0, \tag{1b}$$

where the 4-vector $\boldsymbol{U}(\boldsymbol{x}, t)$ and the flux functions $\boldsymbol{F}(\boldsymbol{U})$ and $\boldsymbol{G}(\boldsymbol{U})$ are defined as

$$\boldsymbol{U} = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ E \end{pmatrix} \quad \text{and} \quad \boldsymbol{F}(\boldsymbol{U}) = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ u(E+p) \end{pmatrix} \quad \text{and} \quad \boldsymbol{G}(\boldsymbol{U}) = \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ v(E+p) \end{pmatrix}.$$

Here, the vector $\boldsymbol{x} = (x, y)$ denotes the two Cartesian coordinates, ρ and E are the density and energy fields, respectively, u and v are the velocities in the x-direction and y-direction, respectively, and we use the notation $\boldsymbol{u}$ to denote the velocity vector field $\boldsymbol{u} = (u, v)$. The pressure function p is defined by the equation of state

$$p = (\gamma - 1)\left(E - \frac{1}{2}\rho|\boldsymbol{u}|^2\right), \tag{2}$$

where $|\boldsymbol{u}| = \sqrt{u^2 + v^2}$ and $\gamma > 1$ is the adiabatic constant. The system (1) represents the conservation of mass, momentum, and energy:

$$\partial_t \rho + \operatorname{div}(\rho \boldsymbol{u}) = 0\,, \quad \partial_t(\rho \boldsymbol{u}) + \operatorname{div}(\rho \boldsymbol{u} \otimes \boldsymbol{u}) + \nabla p = 0\,, \quad \partial_t E + \operatorname{div}((E+p)\boldsymbol{u}) = 0\,,$$

and defines a hyperbolic system, in the sense that both $\nabla \boldsymbol{F}$ and $\nabla \boldsymbol{G}$ have real eigenvalues; in particular, the four eigenvalues of $\nabla \boldsymbol{F}$ are

$$\lambda_1 = u - c\,, \quad \lambda_2 = \lambda_3 = u\,, \quad \lambda_4 = u + c\,,$$

and the four eigenvalues of $\nabla \boldsymbol{G}$ are

$$\lambda_5 = v - c\,, \quad \lambda_6 = \lambda_7 = v\,, \quad \lambda_8 = v + c\,.$$

The eigenvalues $\lambda_1, \lambda_5, \lambda_4$ and λ_8 correspond to sound waves, while the repeated eigenvalues $\lambda_2, \lambda_3, \lambda_6$, and λ_7 correspond to vorticity and entropy waves. We define the *maximum wave speed* $S(\boldsymbol{u})$ as

$$S(\boldsymbol{u}) = [S(\boldsymbol{u})](t) = \max_{i=1,\ldots,8} \max_{\Omega} \{|\lambda_i(x, y, t)|\}\,. \tag{3}$$

We focus on solutions $\boldsymbol{U}$ to the compressible Euler equations (1) that have a jump discontinuity across a time-dependent curve

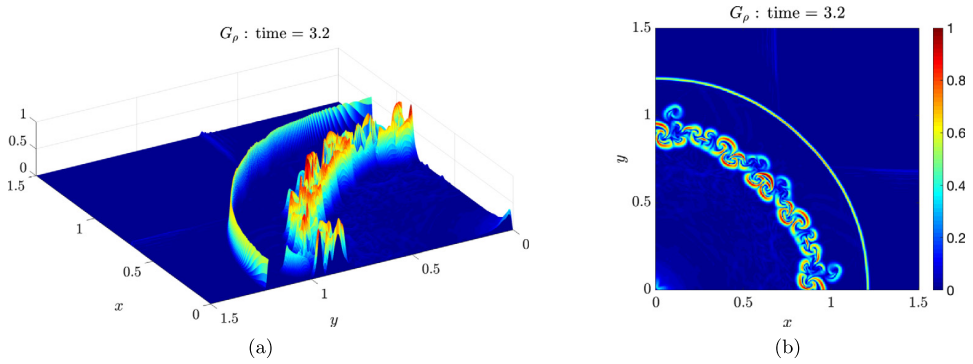


Fig. 1. The function G_ρ for the Sod explosion problem at time $t = 3.2$. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

$$\Gamma(t) = \cup_{i=1}^{p} \Gamma_i(t)\,, \tag{4}$$

where for each index $i = 1, ..., p$, $\Gamma_i(t)$ represents either a shock front or a contact discontinuity. In the case that $\Gamma_i(t)$ is a closed curve, we define $\boldsymbol{U}^+(x,t)$ to be the value of $\boldsymbol{U}(x,t)$ inside of $\Gamma_i(t)$ and $\boldsymbol{U}^-(x,t)$ to be the value of outside of $\Gamma_i(t)$. We then set

$$[\boldsymbol{U}]_-^+ = \boldsymbol{U}^+ - \boldsymbol{U}^- \text{ at } \Gamma(t).$$

Let $\vec{n} = \vec{n}(x,y,t)$ and $\vec{\tau} = \vec{\tau}(x,y,t)$ be the unit normal and tangent vectors to $\Gamma(t)$, respectively. The Rankine-Hugoniot conditions relate the speed of propagation $\dot{\sigma}$ of the curve of discontinuity $\Gamma(t)$ with the jump discontinuity in the variables $\boldsymbol{U}$ via the relation

$$\dot{\sigma}[\boldsymbol{U}]_-^+ = [\boldsymbol{F}\cdot\vec{n}\,]_-^+\,.$$

If $\Gamma_i(t)$ is a shock,

$$\left[\boldsymbol{u}\cdot\vec{n}\right]_-^+ \neq 0 \text{ and } [\boldsymbol{u}\cdot\vec{\tau}\,]_-^+ = 0\,,$$

while if $\Gamma_i(t)$ is a contact discontinuity, then

$$\left[\boldsymbol{u}\cdot\vec{n}\right]_-^+ = 0 \text{ and } [\boldsymbol{u}\cdot\vec{\tau}\,]_-^+ \neq 0.$$

For the problems we consider in this paper, $[\rho]_-^+ \neq 0$ for both shocks and contacts.

3. The *C*-method in the 2-*D* setting

3.1. Smoothly localizing the curves of discontinuity and tracking the geometry

As noted above, the jump discontinuities of the density function describe the location of both the shock and contact fronts; as such, it is natural to use $|\nabla\rho(\boldsymbol{x},t)|$ as an indicator for these time-dependent curves. Consequently, we track discontinuities by considering the quantity

$$G_\rho := \frac{|\nabla\rho|}{\max_\Omega |\nabla\rho|}\,. \tag{5}$$

We note that in the 1-*D* *C*-method formulation in [18], we use $|\partial_x u|$ instead of $|\partial_x\rho|$ as the forcing function for the *C*-equation. In 1-*D*, using either $|\partial_x u|$ or $|\partial_x\rho|$ as the forcing, together with the compression switch, produces identical results. In 2-*D*, the velocity is a vector quantity, whereas the density is a scalar function, so that the latter provides a simpler approach to tracking discontinuities.

In 2-*D*, we are interested in tracking both shock fronts and contact curves, and the density gradient provides a natural method for tracking both of these discontinuities. In Fig. 1, we see that G_ρ captures both of the discontinuities, namely the shock front and the contact discontinuity, present in the solution.

Although the function G_ρ is able to localize artificial viscosity to a curve of discontinuity, its use in artificial viscosity operators often serves to produce an oscillatory solution [20,18]. This is due to the rough nature of the localizing function G_ρ. Consequently, we first produce a space-time smooth variant of G_ρ through the use of the *C*-method. We describe the natural 2-*D* generalization of the 1-*D* *C*-equation as follows: we first define the operator $\mathscr{L}$ as

$$\mathscr{L}[C\,;\varepsilon,\kappa] \coloneqq -\frac{S(\boldsymbol{u})}{\varepsilon|\delta\boldsymbol{x}|}C + \kappa\, S(\boldsymbol{u})|\delta\boldsymbol{x}|\,\Delta C\,, \tag{6}$$

where $\delta\boldsymbol{x} = (\delta x\,, \delta y)$ with δx, δy the grid spacings in the x and y directions, respectively, $S(\boldsymbol{u})$ is the maximum wave-speed (3), and $\Delta = \partial_{xx} + \partial_{yy}$ denotes the 2-D Laplace operator. The space-time smooth version of G_ρ, which we denote by $C = C(\boldsymbol{x}, t)$, is the solution to the scalar linear parabolic PDE

$$\partial_t C(\boldsymbol{x}, t) - \mathscr{L}[C(\boldsymbol{x}, t); \varepsilon, \kappa] = \mathbb{1}_{\mathrm{div}\boldsymbol{u}<0}\frac{S(\boldsymbol{u})}{\varepsilon|\delta\boldsymbol{x}|}G_\rho(\boldsymbol{x}, t)\,, \tag{7}$$

where the function $\mathbb{1}_{\mathrm{div}\boldsymbol{u}<0}$ is a *compression switch*[1] that ensures that C vanishes in regions of expansion, where there are no discontinuities in the solution. The parameters ε and κ in (6) control the support and smoothness of the solution C, respectively [20,18].

The function $C(\boldsymbol{x}, t)$ provides not only the location of the shock and contact fronts, but also a good approximation to the geometry of the front. Specifically, the C function is sufficiently localized so as to provide the shape of the evolving front. In Fig. 2, we show results of the evolution of a contact discontinuity associated to the Rayleigh-Taylor instability (which is discussed in §12). As can be seen, $|\nabla\rho|$ and hence the function G_ρ track the material interface accurately but the function G_ρ is very rough, particularly in directions tangential to the contact front. The function C, by contrast, is smooth and exhibits no oscillatory behavior, but is still highly localized to the contact discontinuity.

3.2. The isotropic C-method and the Euler-C system

We begin with a very natural generalization of the 1-D C-method to the 2-D setting. In particular, we first consider the 2-D Euler-C system with isotropic space-time smooth artificial viscosity:

$$\partial_t\rho + \mathrm{div}(\rho\boldsymbol{u}) = 0, \tag{8a}$$

$$\partial_t(\rho u) + \partial_x(\rho u^2 + p) + \partial_y(\rho uv) = \mathrm{div}\left(\tilde{\beta}^u\,\rho\,C\,\nabla u\right), \tag{8b}$$

$$\partial_t(\rho v) + \partial_x(\rho uv) + \partial_y(\rho u^2 + p) = \mathrm{div}\left(\tilde{\beta}^u\,\rho\,C\,\nabla v\right), \tag{8c}$$

$$\partial_t E + \mathrm{div}(\boldsymbol{u}(E + p)) = \mathrm{div}\left(\tilde{\beta}^E\,\rho\,C\,\nabla(E/\rho)\right), \tag{8d}$$

$$\partial_t C - \mathscr{L}[C\,;\varepsilon,\kappa] = \mathbb{1}_{\mathrm{div}\boldsymbol{u}<0}\frac{S(\boldsymbol{u})}{\varepsilon|\delta\boldsymbol{x}|}G_\rho\,, \tag{8e}$$

where the pressure p is given by the equation of state (2), $\mathscr{L} = \mathscr{L}[C\,;\varepsilon,\kappa]$ is the operator defined in (6), G_ρ is the forcing function defined in (5), and the artificial viscosity parameters are given by

$$\tilde{\beta}^{(\cdot)} \coloneqq \frac{|\delta\boldsymbol{x}|^2}{\max_\Omega C}\,\beta^{(\cdot)}\,, \tag{9}$$

with $\beta^{(\cdot)}$ a constant.

We refer to the system (8) as the *isotropic* C-method, because artificial viscosity is added uniformly in all directions (although clearly still localized to the fronts via the use of C). We remark that the particular form of artificial viscosity used in the momentum equations (8b) and (8c) as well as the energy equation (8d) ensure that total energy remains conserved, and that E continues to evolve as the total energy function. For simplicity, suppose that periodic boundary conditions are enforced. On the one hand, integrating the energy equation (8d) yields $\frac{\mathrm{d}}{\mathrm{d}t}\int_\Omega E\,\mathrm{d}\boldsymbol{x} = 0$. On the other hand, multiplying (8b) and (8c) by u and v, respectively, integrating over the domain Ω, summing the resulting quantities and utilizing the conservation of mass equation (8a), the energy equation (8d), and the equation of state (2) yields

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_\Omega \frac{1}{2}\rho|\boldsymbol{u}|^2 + \frac{p}{\gamma - 1}\,\mathrm{d}\boldsymbol{x} = 0\,.$$

This shows that the velocity $\boldsymbol{u}$ and the pressure p adjust accordingly to maintain the relation (2), and that the Euler-C system conserves the total energy.

The space-time smooth localizing function C ensures that viscosity is added only at discontinuities, thereby ensuring that the solution retains high-order accuracy away from discontinuities, and, moreover, given that $C(\boldsymbol{x}, t)$ is a solution to a reaction-diffusion equation, it is smooth in both space and time. See [20] for the analysis of the solutions C.

[1] $\mathbb{1}_{\mathrm{div}\boldsymbol{u}<0}$ is equal to 1 on the set $\mathrm{div}\boldsymbol{u} < 0$ and is equal to zero on $\mathrm{div}\boldsymbol{u} \geq 0$.

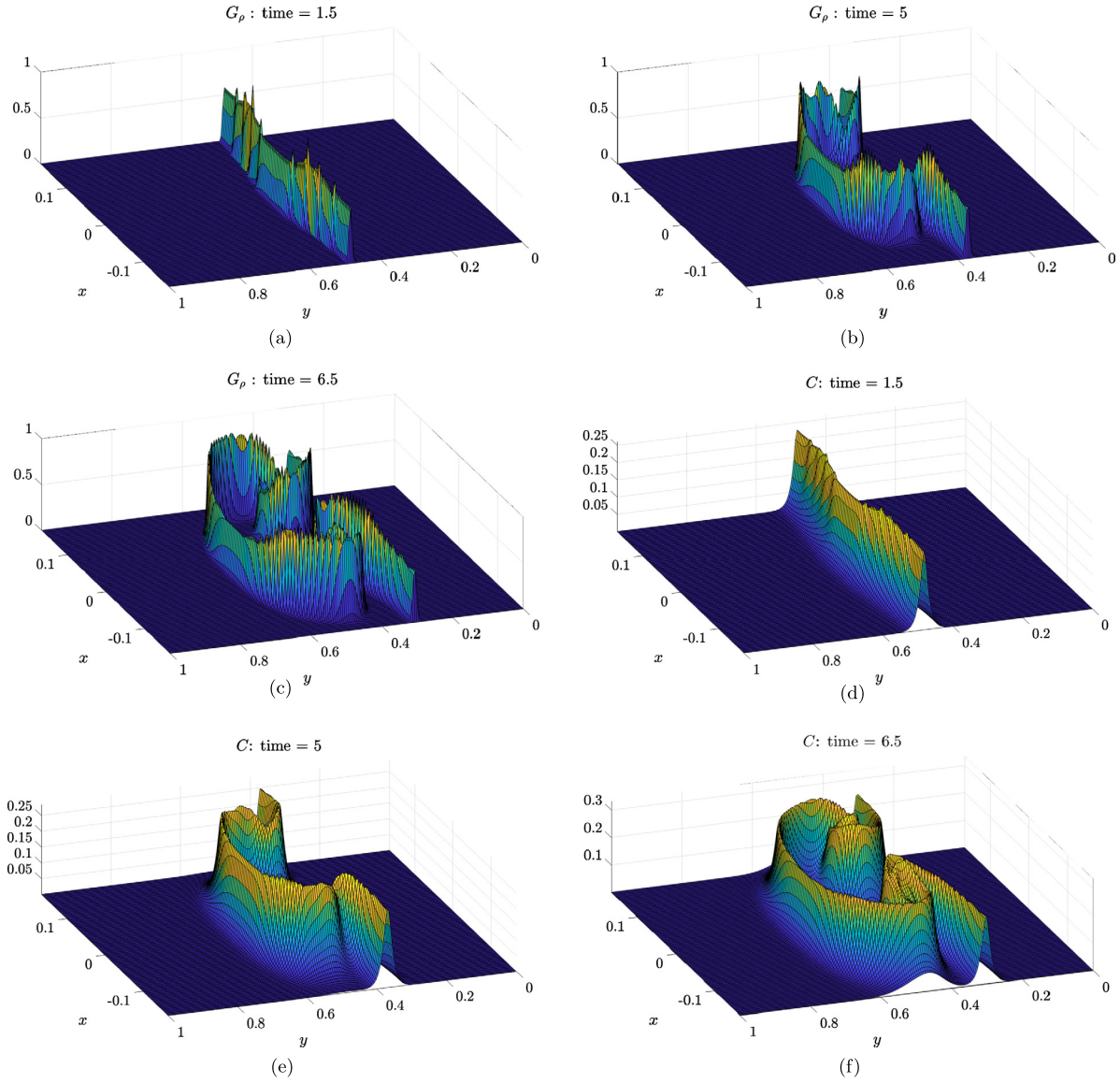


Fig. 2. Comparison of G_ρ and C for the Rayleigh-Taylor instability at various times. Plots (a,b,c) are surface plots of G_ρ, and plots (d,e,f) are surface plots of the corresponding C.

4. The anisotropic C-method for contact discontinuities

We next consider the *anisotropic* C-method, specifically designed for the long-time evolution of contact discontinuities in the presence of Rayleigh-Taylor (RT) instabilities which lead to Kelvin-Helmholtz (KH) roll-up. We are particularly interested in the case that $[\rho]^+_- \neq 0$ across the contact discontinuity,[2] for which we also have that $\left[\boldsymbol{u}\cdot\vec{n}\right]^+_- = 0$ and $[\boldsymbol{u}\cdot\vec{\tau}]^+_- \neq 0$.

Conservation of mass can be written as

$$\partial_t \rho + \boldsymbol{u}\cdot\nabla\rho = -\rho\,\mathrm{div}\,\boldsymbol{u}\,,$$

[2] KH instabilities can occur with a constant density profile, but we consider problems for which $[\rho]^+_- \neq 0$. The method described here requires the condition $[\rho]^+_- \neq 0$ to calculate the normal and tangent vectors to the evolving front. However, we remark that our algorithm can be adapted for problems for which $[\rho]^+_- = 0$, and details are provided in the paper [19] under preparation.

and near an evolving front with tangent and normal vectors given by $\vec{\tau}$ and $\vec{n}$, respectively, the divergence of the velocity vector field $\boldsymbol{u}$ is given by

$$\operatorname{div} \boldsymbol{u} = \partial_{\vec{\tau}} \boldsymbol{u} \cdot \vec{\tau} + \partial_{\vec{n}} \boldsymbol{u} \cdot \vec{n} \,.$$

Across a contact curve, $\partial_{\vec{n}} \boldsymbol{u} \cdot \vec{n}$ remains smooth, while $\partial_{\vec{\tau}} \boldsymbol{u} \cdot \vec{\tau}$ can become extremely oscillatory due to a combination of the discontinuity of $\boldsymbol{u} \cdot \vec{\tau}$ across the contact curve together with interpolation error of the contact curve onto a fixed grid (particularly in specifying the initial data). For simulations that require a great deal of time steps, it is important to add artificial viscosity in the tangential directions, but not in the normal directions.

In classical RT problems (and particularly for low Mach-number flows), instabilities that are generated by a small perturbation of the equilibrium interface position require a large number of time-steps to fully develop the KH roll-up of the contact curve, and it is most often the case that fluid mixing (due to numerical dissipation) is present in this roll-up region, so that the amplitude of the density and the gradient of the density can be significantly smaller than their maximum values.

The objective of our anisotropic C-method is to add diffusion only in directions that are tangent to the contact discontinuity, while adding no artificial diffusion in directions that are normal to the contact curve. In doing so, we can maintain a very sharp interface, and prevent over-diffusion of the slight "hills-and-valleys" which arise in the KH mixing zones. Moreover, when generating the RT instability of a small interface perturbation on a uniform rectangular grid (rather than using a velocity perturbation as done by ATHENA [29]), spurious *tangential spikes* can form in the velocity fields u and v along the contact curve; these spikes, in turn, generate small-scale numerical KH structures which contaminate the solution (this is discussed further in §12.1). Consequently, it is necessary to remove these spikes while maintaining a sharp interface; this may be accomplished through the use of anisotropic diffusion.

4.1. Calculating the normal and tangential directions to $\Gamma(t)$

The first task is to accurately compute a good approximation to the tangent vectors $\vec{\tau}$ to any curve of discontinuity $\Gamma(t)$, defined in (4). For the problems we consider here, this may be accomplished by setting

$$\tau_1(x, y, t) \coloneqq \vec{\tau} \cdot \boldsymbol{e}_1 = -\partial_y \rho \,, \tag{10a}$$

$$\tau_2(x, y, t) \coloneqq \vec{\tau} \cdot \boldsymbol{e}_2 = \partial_x \rho \,, \tag{10b}$$

where $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$ denote the unit vectors in the x and y directions, respectively. In Fig. 3, we provide vector plots of $\vec{\tau}$ calculated using (10), as well as surface plots of each component of $\vec{\tau}$, for the specific case of the RT instability.[3]

As shown in Fig. 3, the functions τ_1 and τ_2 suffer from the same issue affecting the localizing function G_ρ; namely, a lack of smoothness in both space and time. Consequently, we utilize the space-time smoothing mechanism provided by the C-method to produce regularized versions of τ_1 and τ_2, which we denote C^{τ_1} and C^{τ_2}, respectively:

$$\partial_t C - \mathscr{L}\left[C^{\tau_1}; \varepsilon, \kappa\right] = \frac{S(\boldsymbol{u})}{\varepsilon |\delta \boldsymbol{x}|} \tau_1 \,, \tag{11a}$$

$$\partial_t C - \mathscr{L}\left[C^{\tau_2}; \varepsilon, \kappa\right] = \frac{S(\boldsymbol{u})}{\varepsilon |\delta \boldsymbol{x}|} \tau_2 \,, \tag{11b}$$

where $\mathscr{L}$ is the operator defined in (6). We also define the function

$$\mu = \mu(t) \coloneqq \max_{\Omega} \left\{ \max \left\{ |C^{\tau_1}| \,, |C^{\tau_2}| \right\} \right\} \,, \tag{12}$$

which can be used to produce a "normalized" tangent vector $\frac{1}{\mu} \vec{C^\tau} = \frac{1}{\mu} C^{\tau_1} \boldsymbol{e}_1 + \frac{1}{\mu} C^{\tau_2} \boldsymbol{e}_2$.

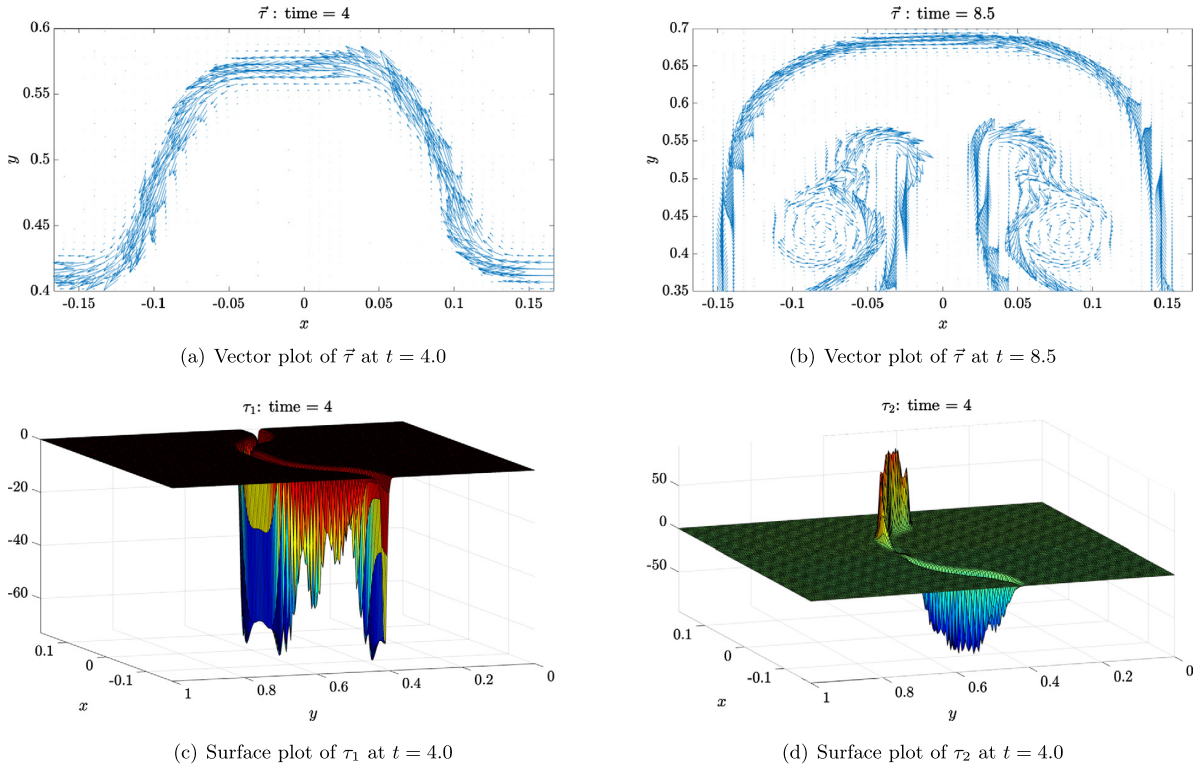
In Fig. 4, we provide vector plots of the vector $\vec{C^\tau}$, as well as surface plots of the components $\frac{1}{\mu} C^{\tau_1}$ and $\frac{1}{\mu} C^{\tau_2}$. These should be contrasted with the corresponding figures in Fig. 3; we see that $\vec{C^\tau}$ is much smoother than $\vec{\tau}$, while remaining localized to the discontinuity $\Gamma(t)$.

4.2. Directional artificial viscosity and the Euler-C^τ system

We now consider the following Euler-C^τ system for contact discontinuity evolution:

$$\partial_t \rho + \operatorname{div}(\rho \boldsymbol{u}) = 0 \,, \tag{13a}$$

[3] We note that the vector plots in Figs. 3(a) and 3(b) appear asymmetric only because of the Matlab plotting routine we use. In fact, for the RT problem we consider in this paper, the interface $\Gamma(t)$ is symmetric across the line $x = 0$. Since we fix an orientation for $\Gamma(t)$, the horizontal component of the tangent vector τ_1 is symmetric across $x = 0$, while the vertical component τ_2 is anti-symmetric, as can be seen in Figs. 3(c) and 3(d). These facts, combined with the particular Matlab plotting routine we employ, cause the perceived asymmetry in the vector plots.



(a) Vector plot of $\vec{\tau}$ at $t = 4.0$

(b) Vector plot of $\vec{\tau}$ at $t = 8.5$

(c) Surface plot of τ_1 at $t = 4.0$

(d) Surface plot of τ_2 at $t = 4.0$

Fig. 3. Calculation of the tangent vector $\vec{\tau}$ for the RT instability.

$$\partial_t(\rho u) + \partial_x(\rho u^2 + p) + \partial_y(\rho uv) = \partial_i\left(\tilde{\beta}\,\rho\, C\, C^{\tau_i}\, C^{\tau_j}\,\partial_j u\right), \tag{13b}$$

$$\partial_t(\rho v) + \partial_x(\rho uv) + \partial_y(\rho v^2 + p) = \partial_i\left(\tilde{\beta}\,\rho\, C\, C^{\tau_i}\, C^{\tau_j}\,\partial_j v\right), \tag{13c}$$

$$\partial_t E + \operatorname{div}(\boldsymbol{u}(E + p)) = 0, \tag{13d}$$

$$\partial_t C - \mathscr{L}[C\,;\varepsilon,\kappa] = \frac{S(\boldsymbol{u})}{\varepsilon|\delta\boldsymbol{x}|}G_\rho\,, \quad \partial_t C^{\tau_i} - \mathscr{L}[C^{\tau_i}\,;\varepsilon,\kappa] = \frac{S(\boldsymbol{u})}{\varepsilon|\delta\boldsymbol{x}|}\tau_i \text{ for } i = 1,2\,, \tag{13e}$$

where G_ρ is given by (5), τ_1 and τ_2 by (10), $\mathscr{L}$ by (6), and we utilize the Einstein summation convention, where a repeated free index in the same term implies summation over all values of that index. We note that $\partial_1 \equiv \partial_x$ and $\partial_2 \equiv \partial_y$, and we use the two notations interchangeably. The artificial viscosity parameter is defined by

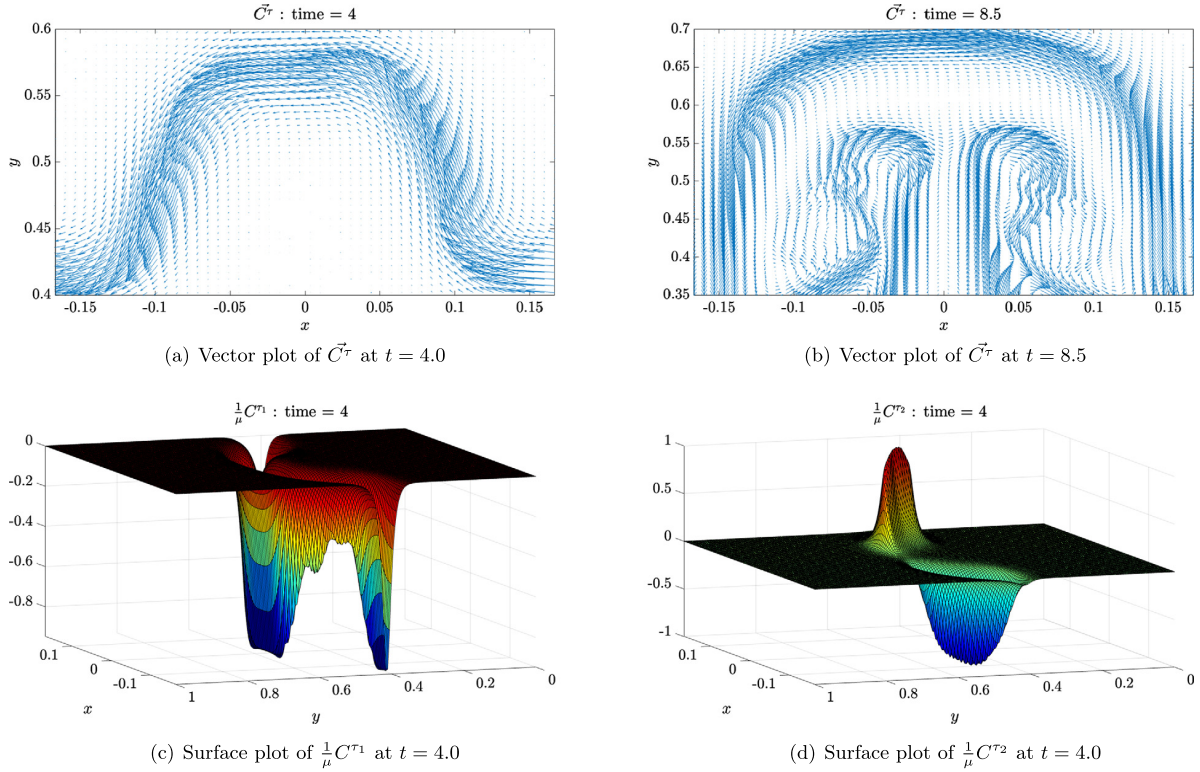
$$\tilde{\beta} = \frac{|\delta\boldsymbol{x}|^2}{\mu^2\,\max_\Omega C}\,\beta\,, \tag{14}$$

with $\mu = \max_\Omega\left\{\max\left\{|C^{\tau_1}|\,,|C^{\tau_2}|\right\}\right\}$.

We note that the artificial viscosity operator $-\partial_i\left(\tilde{\beta}\,\rho\, C\, C^{\tau_i}\, C^{\tau_j}\,\partial_j\right)$ is a positive semi-definite operator. The proof is as follows: taking the scalar product of $-\partial_i\left(\tilde{\beta}\,\rho\, C\, C^{\tau_i}\, C^{\tau_j}\,\partial_j \boldsymbol{u}\right)$ with $\boldsymbol{u}$ and integrating over Ω yields

$$\int_\Omega \tilde{\beta}\,\rho\, C\, C^{\tau_j}\,\partial_j\boldsymbol{u}\cdot C^{\tau_i}\partial_i\boldsymbol{u}\,\mathrm{d}\boldsymbol{x} = \int_\Omega \tilde{\beta}\,\rho\, C\,|\partial_{\vec{C}^\tau}\boldsymbol{u}|^2\,\mathrm{d}\boldsymbol{x} \geq \tilde{\lambda}\int_\Omega |\partial_{\vec{C}^\tau}\boldsymbol{u}|^2\,\mathrm{d}\boldsymbol{x}\,,$$

for some $\tilde{\lambda} \geq 0$. Here, $\partial_{\vec{C}^\tau} = \vec{C}^\tau\cdot\nabla$ denotes the smoothed tangential derivative operator. Thus, the anisotropic artificial viscosity operator is obtained as the Euler-Lagrange extremum associated to the function $\int \tilde{\beta}\,\rho\, C\,|\partial_{\vec{C}^\tau}\boldsymbol{u}|^2\,\mathrm{d}\boldsymbol{x}$. Just as in the case of isotropic artificial viscosity, our solutions to the Euler-C^τ system preserve total energy $E(t)$ (see §3.2).



(a) Vector plot of $\vec{C}^{\tau}$ at $t = 4.0$

(b) Vector plot of $\vec{C}^{\tau}$ at $t = 8.5$

(c) Surface plot of $\frac{1}{\mu}C^{\tau_1}$ at $t = 4.0$

(d) Surface plot of $\frac{1}{\mu}C^{\tau_2}$ at $t = 4.0$

Fig. 4. Calculation of the smoothed tangent vector $\vec{C}^{\tau}$ for the RT instability.

5. The *C*-method for shock-wall collision

We now present a simple extension of the 1-*D* shock-wall collision scheme (see §3 [18]). Recall that the main novelty of the 1-*D* shock wall collision scheme is the use of a *wall function* $\overline{C}(t)$ which naturally activates during shock-wall collision and bounce-back. This allows for the addition of extra "wall viscosity" during shock-collision, which results in the suppression of post-collision noise while maintaining high-order accuracy prior to collision.

The natural generalization to the two-dimensional setting results in the 2-*D* Euler-*C*-*W* scheme:

$$\partial_t \rho + \mathrm{div}(\rho \boldsymbol{u}) = 0, \tag{15a}$$

$$\partial_t(\rho u) + \partial_x(\rho u^2 + p) + \partial_y(\rho uv) = \mathrm{div}\left(\mathcal{B}^u \, \rho \, C \, \nabla u\right), \tag{15b}$$

$$\partial_t(\rho v) + \partial_x(\rho uv) + \partial_y(\rho u^2 + p) = \mathrm{div}\left(\mathcal{B}^u \, \rho \, C \, \nabla v\right), \tag{15c}$$

$$\partial_t E + \mathrm{div}(\boldsymbol{u}(E + p)) = \mathrm{div}\left(\mathcal{B}^E \, \rho \, C \, \nabla(E/\rho)\right), \tag{15d}$$

$$\partial_t C - \mathscr{L}[C\,;\varepsilon,\kappa] = \mathbb{1}_{\mathrm{div}\boldsymbol{u}<0} \, \frac{S(\boldsymbol{u})}{\varepsilon|\delta \boldsymbol{x}|} G_\rho\,, \tag{15e}$$

$$\partial_t C_w - \mathscr{L}[C_w\,;\varepsilon_w,\kappa_w] = \mathbb{1}_{\mathrm{div}\boldsymbol{u}<0} \, \frac{S(\boldsymbol{u})}{\varepsilon_w|\delta \boldsymbol{x}|} G_\rho\,, \tag{15f}$$

where $\mathscr{L}$ is the operator defined in (6), G_ρ is the forcing function defined in (5), and $\mathbb{1}_{\mathrm{div}\boldsymbol{u}<0}$ is a compression switch. The artificial viscosity parameters $\mathcal{B}$ are given by

$$\mathcal{B}^{(\cdot)} = \frac{|\delta \boldsymbol{x}|^2}{\max_\Omega C} \left(\beta^{(\cdot)} + \beta_w^{(\cdot)} \overline{C}(x,t)\right), \tag{16}$$

with the wall function $\overline{C}(x,t)$ defined by

$$\overline{C}(x,t) = \frac{C_w(x, y_l, t)}{\max_\Omega C_w(x,y,t)}\,.$$

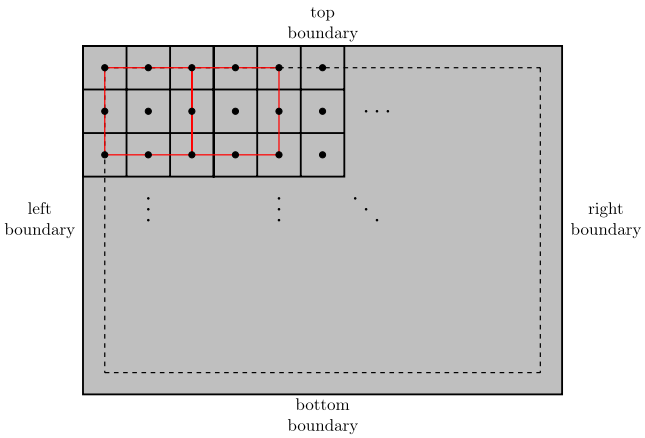


Fig. 5. Grid setup for the construction of the wavelets in 2-*D*. The dashed black curve denotes the boundary $\partial\Omega$, while the solid black lines indicate cell edges. The black dots indicate cell centers, and the domains bounded by red lines indicate the support of each of the highest frequency wavelets.

Here, we assume that the shock-wall collision occurs at the bottom boundary $y = y_l$. A Neumann boundary condition for C_w is enforced at the bottom boundary $y = y_l$. As in the 1-*D* case, this results in the smooth growth in time of the amplitude of $\overline{C}(x,t)$ as the shock approaches the wall, and allows for the addition of "wall viscosity" during shock-wall collision and bounce-back.

6. A wavelet-based 2-*D* noise detection and removal algorithm

In this section, we extend the noise indicator algorithm presented in [18] to the two-dimensional setting, using the same ideas as in the 1-*D* case. Again, we construct a family of wavelets $\{\psi_{i,j}\}$ and obtain a set of wavelet coefficients $\{C_{i,j}\}$, found by calculating the inner product of the highest frequency wavelets with the noisy function. These wavelet coefficients will indicate the location of the noise, and we employ a localized heat equation-based solver to remove this noise.

6.1. Construction of the highest frequency wavelets

We first discretize our grid by assuming we have M cells in the x-direction, and N cells in the y-direction. Label the cell centers by (x_i, y_j), where x_i and y_j are given by

$$x_i = x_l + (i-1)\cdot\delta x \quad \text{for } i = 1,\dots,M,$$
$$y_j = y_l + (j-1)\cdot\delta y \quad \text{for } j = 1,\dots,N,$$

with $\delta x = (x_r - x_l)/(M-1)$ and $\delta y = (y_r - y_l)/(N-1)$. We group the cells into 3×3 blocks of 9 cells each, and then define the highest frequency wavelet with support over the domain spanned by the cell centers in each of these blocks, as shown in Fig. 5.

This yields a set of $\frac{(M-1)}{2}\times\frac{(N-1)}{2}$ highest frequency wavelets. We denote these wavelets by $\psi_{i,j} = \psi_{i,j}(x,y)$, for $i = 1,\dots,(M-1)/2$ and $j = 1,\dots,(N-1)/2$, with each $\psi_{i,j}$ is supported in the rectangular domain $\mathcal{I}_{i,j} = [x_{2i-1}, x_{2i+1}]\times[y_{2j-1}, y_{2j+1}]$.

We now have to fix a form for the 2-*D* wavelet. The two key properties that are required of the wavelet family are:

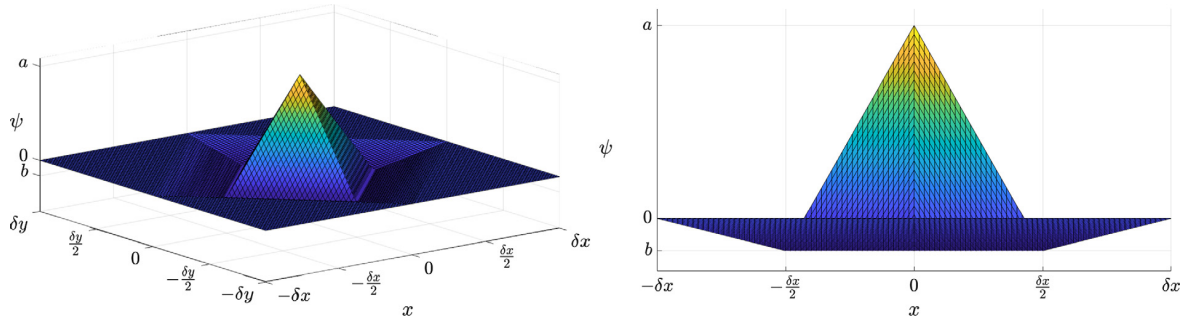
1. Zero mean:

$$\int_\Omega \psi_{i,j}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = 0.$$

2. "Quasi-orthogonality" of the form:

$$\int_\Omega \psi_{i,j}(\boldsymbol{x})\cdot\psi_{r,s}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = \delta_{ir}\delta_{js}, \text{ for } i,r = 1,\frac{M-1}{2} \text{ and } j,s = 1,\dots,\frac{N-1}{2},$$

so that each of the highest frequency wavelets is orthogonal to every other wavelet of the same frequency.



(a) the highest frequency wavelet ψ (b) cross sectional view of ψ along $y = 0$

Fig. 6. The highest frequency wavelet $\psi = \psi(x, y)$ in 2-D. We assume that the support of ψ is $\mathcal{I} = [-\delta x, \delta x] \times [-\delta y, \delta y]$.

We recall from [18] that each member of the 1-D wavelet family is oscillatory and orthogonal to all linear functions. Consequently, we design the 2-D wavelet family to also have these two properties. For simplicity, assume that a highest frequency wavelet ψ is supported in the domain $[-\delta x, \delta x] \times [-\delta y, \delta y]$. The highest frequency wavelets $\psi_{i,j}$ are then obtained by translation of ψ to the domain $\mathcal{I}_{i,j}$. Our wavelets take the form shown in Fig. 6.

The exact formula for the highest frequency wavelet ψ that is supported in the domain $\mathcal{I} = [-\delta x, \delta x] \times [-\delta y, \delta y]$ is

$$\psi(x, y) = \begin{cases} \psi^{(1)}(x, y), & \text{if } 0 \le |\frac{x}{\delta x}| + |\frac{y}{\delta y}| \le \frac{1}{2}, \\ \psi^{(2)}(x, y), & \text{if } \frac{1}{2} \le |\frac{x}{\delta x}| + |\frac{y}{\delta y}| \le 1, \end{cases}$$

where $\psi^{(1)}$ and $\psi^{(2)}$ are given by

$$\psi^{(1)}(x, y) = \begin{cases} a + 2(b-a)\left(\frac{x}{\delta x} + \frac{y}{\delta y}\right), & \text{if } x > 0, y > 0, \\ a + 2(b-a)\left(-\frac{x}{\delta x} + \frac{y}{\delta y}\right), & \text{if } x < 0, y > 0, \\ a + 2(b-a)\left(-\frac{x}{\delta x} - \frac{y}{\delta y}\right), & \text{if } x < 0, y < 0, \\ a + 2(b-a)\left(\frac{x}{\delta x} - \frac{y}{\delta y}\right), & \text{if } x > 0, y < 0, \end{cases}$$

$$\psi^{(2)}(x, y) = \begin{cases} 2b - 2b\left(\frac{x}{\delta x} + \frac{y}{\delta y}\right), & \text{if } x > 0, y > 0, \\ 2b - 2b\left(-\frac{x}{\delta x} + \frac{y}{\delta y}\right), & \text{if } x < 0, y > 0, \\ 2b - 2b\left(-\frac{x}{\delta x} - \frac{y}{\delta y}\right), & \text{if } x < 0, y < 0, \\ 2b - 2b\left(\frac{x}{\delta x} - \frac{y}{\delta y}\right), & \text{if } x > 0, y < 0. \end{cases}$$

Here, the values of a and b are chosen so that ψ satisfies $||\psi||_{L^2(\Omega)} = 1$ and the zero mean condition $\langle \psi, 1 \rangle_{L^2(\Omega)} = 0$, and can be calculated as

$$a = -6b \quad \text{and} \quad b = -\sqrt{\frac{3}{8} \cdot \frac{1}{\delta x \, \delta y}}.$$

The highest frequency wavelets $\psi_{i,j}$ are then obtained by translation of ψ to the domain $\mathcal{I}_{i,j} = [x_{2i-1}, x_{2i+1}] \times [y_{2j-1}, y_{2j+1}]$.

Now, given a function $f(x, y)$ defined at the cell centers (x_i, y_j), we wish to calculate the inner product of f with each of the highest frequency wavelets. The first step is to approximate the function $f(x, y)$ over $\mathcal{I}_{i,j}$, the support of $\psi_{i,j}$. In 1-D, a function $f(x)$ is approximated as a piecewise linear function over the interval $\mathcal{I}_i$, the support of a 1-D highest frequency wavelet, by linearly interpolating between the cell center values (see §4 in [18]). The analogue of a line in the two-dimensional setting is a plane, so a first attempt is to approximate $f(x, y)$ by a plane in each of the sub-cells $[x_r, x_{r+1}] \times [y_s, y_{s+1}]$. However, this is not possible, since 3 points define a plane, whereas each of the sub-cells contains 4 cell center values.

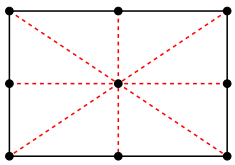


Fig. 7. Dividing $\mathcal{I}_{i,j}$ into 8 regions.

Our solution is to divide $\mathcal{I}_{i,j}$ into 8 regions as shown in Fig. 7. Each of these regions contains precisely 3 cell-center values, and these 3 values define a plane. Thus, we can approximate $f(x,y)$ by a plane in each of these 8 regions, and define a piecewise linear function $\tilde{f}(x,y)$ such that $\tilde{f}(x_i,y_j) = f(x_i,y_j)$. We may then approximate the (i,j)-th wavelet coefficient as

$$\begin{aligned} \mathcal{C}_{i,j}(f) &:= \langle f\,,\psi_{i,j}\rangle_{L^2(\Omega)} \approx \langle \tilde{f}\,,\psi_{i,j}\rangle_{L^2(\Omega)} \\ &= -\frac{\sqrt{6\,\delta x\,\delta y}}{192}\begin{Bmatrix} f_{2i-1,2j+1} & + & 2f_{2i,2j+1} & + & f_{2i+1,2j+1} & + \\ 2f_{2i-1,2j} & + & -12f_{2i,2j} & + & 2f_{2i+1,2j} & + \\ f_{2i-1,2j-1} & + & 2f_{2i,2j-1} & + & f_{2i+1,2j-1} & \end{Bmatrix}, \end{aligned} \tag{17}$$

where $f_{m,n} = f(x_m,y_n)$. One can verify that the inner product of $\psi_{i,j}$ with an arbitrary plane is identically zero i.e. if $f(x,y) = \alpha x + \beta y$ for some constants α and β, then $\mathcal{C}_{i,j}(u) = 0$ for every $i = 1,\ldots,(M-1)/2$ and $j = 1,\ldots,(N-1)/2$. This is analogous to the result in 1-D that each of the highest frequency wavelets is orthogonal to linear functions.

6.2. Algorithm

Given a noisy function $\tilde{u}(x,y)$, we now present an algorithm for first detecting and then subsequently removing the noise. The algorithm is almost identical to that presented for the 1-D case (see §4 in [18]).

6.2.1. Noise detection in the presence of discontinuities

We first calculate, using formula (17), the $\frac{M-1}{2} \times \frac{N-1}{2}$ wavelet coefficients, each associated with one of the highest frequency wavelets. The coefficients that are largest in magnitude should indicate the location of the noise. However, as in the 1-D case, it is possible that the wavelet coefficients that are largest in magnitude are actually indicating the location of the curve of discontinuity $\Gamma(t)$. Since the C-method is taking care of artificial diffusion in this region, one needs to manually "turn off" the noise detection in a small region surrounding the shock curve $\Gamma(t)$. There are numerous ways to do this, but we employ one of the simplest methods, namely to turn off the noise detection if

$$\frac{C}{\max_\Omega C} > \delta_{\text{off}}\,, \tag{18}$$

where δ_{off} is some value between 0 and 1. A typical range of values for δ_{off} is $\delta_{\text{off}} \in [0.05, 0.25]$.

With noise detection deactivated in the region surrounding $\Gamma(t)$, the wavelet coefficients that are largest in magnitude now indicate the location of the noise. We now "turn on" a noise detector function $\mathbb{1}_{\text{noise}}(x,y)$ in the domain $\mathcal{I}_{i,j}$ if the associated wavelet coefficient $\mathcal{C}_{i,j}$ satisfies $|\mathcal{C}_{i,j}| \geq \mathcal{C}_{\text{ref}}$. The constant $\mathcal{C}_{\text{ref}}$ is the wavelet coefficient obtained from a "typical" high-frequency oscillation, namely a hat function (see Fig. 8) centered in the domain $[-\delta x, +\delta x] \times [-\delta y, +\delta y]$ with amplitude δh. The associated wavelet coefficient may then be calculated as

$$\mathcal{C}_{\text{ref}} = \delta h \frac{\sqrt{6\,\delta x\,\delta y}}{16}\,. \tag{19}$$

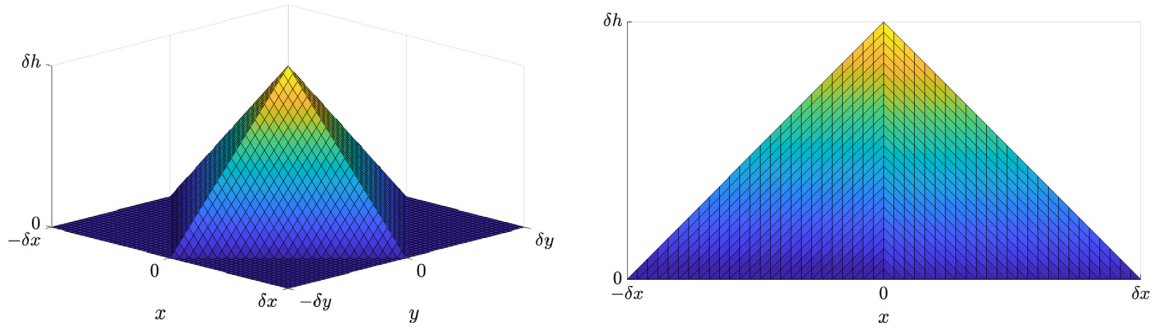
6.2.2. Noise removal with a localized heat equation

The noise removal process in the 2-D case is identical to that in 1-D. We first construct the domain $\mathcal{V}$, given by the union of all domains $\mathcal{I}_{i,j}$ such that the noise detector function $\mathbb{1}_{\text{noise}}(x,y)$ is non-zero in $\mathcal{I}_{i,j}$. We write $\mathcal{V}$ as the union of its connected subsets $\mathcal{V} = \bigcup_k \mathcal{V}_k$, and then define the domains $\tilde{\mathcal{V}}_k$ as the domain $\mathcal{V}_k$ extended by one cell in each outward in each direction (see Fig. 9). For example, if $\mathcal{V}_k = [x_1,x_2] \times [y_1,y_2]$, then $\tilde{\mathcal{V}}_k = [x_1 - \delta x, x_2 + \delta x] \times [y_1 - \delta y, y_2 + \delta y]$.

A localized heat equation with Dirichlet boundary conditions is then solved in each of the domains $\tilde{\mathcal{V}}_k$ for a "de-noised" solution $u(x,y,\tau)$,

$$\partial_\tau u(x,y,\tau) = \eta \cdot \Delta u(x,y,\tau), \quad \text{for } \boldsymbol{x} \in \tilde{\mathcal{V}}_k \text{ and } \tau > 0, \tag{20a}$$

$$u(x,y,0) = \tilde{u}(x,y), \qquad \text{for } \boldsymbol{x} \in \tilde{\mathcal{V}}_k, \tag{20b}$$



(a) reference oscillation: a hat function (b) cross sectional view along $y = 0$

Fig. 8. The 2-D reference oscillation: a hat function with amplitude δh.

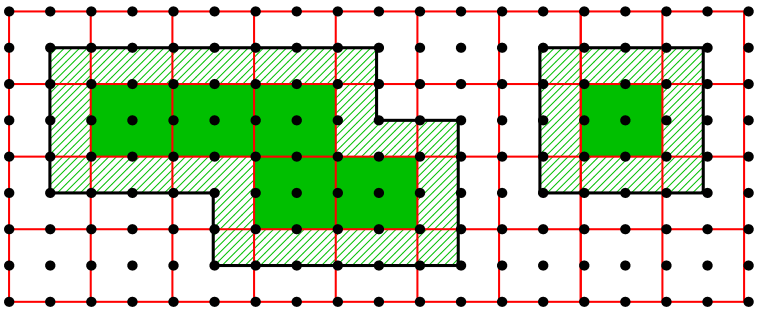


Fig. 9. Construction of the domains $\mathcal{V}_k$ and $\tilde{\mathcal{V}}_k$. The squares bounded by the red lines indicate the support of each of the highest frequency wavelets $\psi_{i,j}$. The shaded green regions indicate where the noise indicator algorithm detects noise, and represent the domains $\mathcal{V}_k$. The hatched regions indicate the extension of each $\mathcal{V}_k$ to $\tilde{\mathcal{V}}_k$. The domains $\tilde{\mathcal{V}}_k$ are the regions bounded by the solid black lines.

$$u(x, y, \tau) = \tilde{u}(x, y), \qquad \text{for } \boldsymbol{x} \in \partial\tilde{\mathcal{V}}_k \text{ and } \tau > 0, \tag{20c}$$

while $u(x, y, \tau) = \tilde{u}(x, y)$ for $\boldsymbol{x} \in \left(\bigcup_k \tilde{\mathcal{V}}_k\right)^{\mathcal{C}}$ and $\tau \geq 0$. The time τ is a fictitious time, introduced solely for the diffusion mechanism, while $0 < \eta \ll 1$ is a small constant, which we refer to as the noise removal viscosity. Equation (20b) is the initial condition, and (20c) is a Dirichlet boundary condition ensuring continuity of $u(x, y, \tau)$ over the domain Ω.

However, as in the 1-D case, it is not necessary to explicitly construct the domains $\tilde{\mathcal{V}}_k$. Instead, one can use the noise detector function $\mathbb{1}_{\text{noise}}(x, y)$ and solve a modified heat equation with Dirichlet boundary conditions, given by

$$\partial_\tau u(x, y, \tau) = \eta \cdot \mathbb{1}_{\text{noise}}(x, y) \cdot \Delta u(x, y, \tau), \quad \text{for } \boldsymbol{x} \in \Omega \text{ and } \tau > 0, \tag{21a}$$

$$u(x, y, 0) = \tilde{u}(x, y), \qquad \text{for } \boldsymbol{x} \in \Omega, \tag{21b}$$

$$u(x, y, \tau) = \tilde{u}(x, y), \qquad \text{for } \boldsymbol{x} \in \partial\Omega \text{ and } \tau > 0. \tag{21c}$$

In practice, the system (21) is solved using a simple, explicit forward Euler time integration scheme, along with a second order central difference approximation for the spatial derivatives. Moreover, in the simulations shown below, a single time-step is sufficient to remove noise; thus, our procedure is the equivalent of the inversion of a Helmholtz elliptic operator, and can hence be viewed as a *filtering* process, in which high frequency noise is eliminated from the solution through a local averaging or localized frequency truncation. However, we remark that for certain simulations with very large amplitude high frequency noise, it may be necessary to solve the localized heat equation for several time steps to completely remove the oscillations.

6.3. Implementation of the algorithm for the Euler equations

We now describe how we implement the noise detection and removal algorithm described above for the particular case of the Euler system (1). Suppose that we are given the solution $\boldsymbol{U} = \boldsymbol{U}(x, y, t_n)$ at time $t = t_n = n \cdot \delta t$, and we wish to calculate the solution $\boldsymbol{U}(x, y, t_{n+1})$ at time $t = t_{n+1} = (n + 1) \cdot \delta t$. The implementation proceeds in two steps:

Table 1
Relevant parameters and variables used in the numerical tests.

Parameter / Variable	Description
β^u, β^E	artificial viscosity coefficients for the momentum and energy, respectively.
β^u_w, β^E_w	wall viscosity coefficients for the momentum and energy, respectively.
δh, δ_{off}, η	amplitude of noise, noise detection deactivation parameter, and noise removal viscosity, respectively.
ε, ε_w	parameters controlling support of C (and C^{τ_i}) and C_w, respectively.
κ, κ_w	parameters controlling smoothness of C (and C^{τ_i}) and C_w, respectively.

1. We first compute in the usual manner the (potentially noisy) solution at time $t = t_{n+1}$. We denote this solution by $\tilde{\boldsymbol{U}}(x, y)$. In the numerical studies below, a simplified WENO-based scheme for the spatial discretization and an explicit Runge-Kutta method for the time integration are used to calculate this solution.
2. We then pass the potentially noisy velocity components $\tilde{u}(x, y)$ and $\tilde{v}(x, y)$ through the noise detection and removal algorithm described in §6.2, to produce de-noised velocity components $u(x, y, t_{n+1})$ and $v(x, y, t_{n+1})$. We then *define* the solution $\boldsymbol{U}(x, y, t_{n+1})$ at time $t = t_{n+1}$ by

$$\boldsymbol{U}(x, y, t_{n+1}) \equiv \begin{bmatrix} \rho(x, y, t_{n+1}) \\ \rho u(x, y, t_{n+1}) \\ \rho v(x, y, t_{n+1}) \\ E(x, y, t_{n+1}) \end{bmatrix} := \begin{bmatrix} \tilde{\rho}(x, y) \\ \tilde{\rho}(x, y) \cdot u(x, y, t_{n+1}) \\ \tilde{\rho}(x, y) \cdot v(x, y, t_{n+1}) \\ \tilde{E}(x, y) \end{bmatrix}.$$

This algorithm mimics the 1-D version in [18]. We apply this procedure to the Rayleigh-Taylor instability in §12 and demonstrate its ability to suppress spurious high frequency noise that otherwise corrupts the solution.

Remark 1. We note that the noise removal procedure for the horizontal velocity component $\tilde{u}$ is completely independent from the noise removal procedure for the vertical velocity component $\tilde{v}$. It is also perhaps more useful to view our algorithm as a predictor-corrector method, in which we first compute auxiliary quantities using the WENO-based portion of the algorithm, and then "correct" these quantities by removing high frequency noise from the solutions in the corrector portion of the method. Extensive testing of the method in the 1-D setting [18] shows that the noise reduction algorithm successfully eliminates high-frequency noise from the auxiliary solution, thereby producing "corrected" solutions with smaller errors. In the 2-D setting, we provide error analysis and convergence tests for the Sedov problem in Table 3. These tests indicate that the noise removal algorithm decreases the error of the computed solution; furthermore, our numerical experiments for the Noh problem in §8 and Rayleigh-Taylor problem in §12 demonstrate that, qualitatively, the noise removal algorithm greatly reduces the numerical error of solutions. Further quantitative evidence in the form of error analysis and convergence tests is presented in [19].

7. The numerical algorithm for the C-method

The systems (1), (8), (13), and (15) are discretized using a simplified finite-differencing WENO scheme for the nonlinear flux terms, and a standard central difference approximation for the diffusion terms. Time integration is done using a k^{th} order Runge-Kutta scheme. For each simulation, a fixed Δt is used at every time step, so that the CFL number can change at each time level; however, for each of the problems presented, the time step Δt is chosen so that the CFL condition is not violated at any time t during the simulation. For convenience, we provide full details of the numerical discretization of these systems in Appendix A. In particular, we refer the reader to Table 5 listing the various methods (and combinations of the methods) that we will use for the numerical tests.

We stress that the WENO-type discretization we use is highly simplified, and is not meant to be representative of the class of full WENO solvers. However, we note that, for certain problems, our simplified WENO-type discretization produces solutions with similar errors and convergence rates to those produced using a standard WENO scheme (see §5.2.5 in [18]).

As with any artificial viscosity scheme, parameters must be chosen for the particular problem under consideration. All of the relevant parameters for the schemes considered are listed in Table 1. In [18], we suggest some practical guidelines on choosing these parameters; a brief summary of the discussion in [18] is the following. We choose the artificial viscosity parameters $\beta^{(\cdot)}$, $\beta^{(\cdot)}_w$, and η large enough to damp post-shock oscillations and high frequency noise both pre and post shock-wall collision. The parameters ε and κ in the C-equations control the support and smoothness of the C-functions. For a more thorough discussion, we refer the reader to §5 of [18].

Table 2
Relative L^1 and L^∞ errors at $t = 4$ of the computed solution minus the exact solution at and convergence for the linear advection problem. The errors are given in percentage form.

Scheme	Error		Cells			
			25×25	50×50	100×100	200×200
WENO	L^1	Error in %	2.894×10^{-2}	9.014×10^{-4}	2.820×10^{-5}	8.821×10^{-7}
		Order	–	5.005	4.998	4.999
WENO	L^∞	Error in %	5.254×10^{-2}	1.929×10^{-3}	6.253×10^{-5}	1.970×10^{-6}
		Order	–	4.767	4.947	4.988

7.1. Accuracy study: linear advection

For the purposes of demonstrating the high order convergence of the base WENO-type scheme, we consider the following linear advection equation [13,8]

$$\partial_t \varphi(\boldsymbol{x}, t) + \operatorname{div}(\boldsymbol{a}\varphi(\boldsymbol{x}, t))) = 0, \quad \boldsymbol{x} \in [-1, 1]^2, t > 0, \tag{22a}$$

$$\varphi(\boldsymbol{x}, 0) = 1 + 0.2 \sin(\pi(x + y)), \quad \boldsymbol{x} \in [-1, 1]^2, t = 0, \tag{22b}$$

with $\boldsymbol{a} = (1, -0.5)$. Periodic boundary conditions are employed, and the exact solution at time t is given by $\varphi(\boldsymbol{x}, t) = 1 + 0.2 \sin(\pi(x + y - 0.5t))$. The problem is run on grids with 25×25, 50×50, 100×100, and 200×200 cells until the final time $t = 4$, at which time the sinusoidal wave has been advected one full wavelength. We choose the time-step so that CFL $= 0.8$. Following [13], the relative L^1 and relative L^∞ errors are listed in percentage form in Table 2. Our simplified WENO-type scheme achieves the advertised fifth-order convergence rate, and the errors are similar to those produced with an "industry-standard" WENO method [13].

8. The Noh infinite strength shock problem

We begin our numerical experiments by considering a radially symmetric version of a classic test of Noh [15,13]. This problem simulates an infinite strength shock wave formed by uniformly compressing a cold gas with constant velocity 1 directed towards the origin. The initial pressure is identically zero, but following [13] we use the initial value $p_0 = 10^{-6}$. The domain is $\Omega = [-1, 1]^2 \subset \mathbb{R}^2$, the adiabatic constant is $\gamma = 5/3$, and the initial data is

$$\begin{bmatrix} \rho_0 \\ (\rho u)_0 \\ (\rho v)_0 \\ E_0 \end{bmatrix} = \begin{bmatrix} 1 \\ -\cos(\theta) \\ -\sin(\theta) \\ 0.5 + 10^{-6}/(\gamma - 1) \end{bmatrix}, \tag{23}$$

where $\theta \in [0, 2\pi)$ is the polar angle. The final time is $t = 2.0$.

The exact solution consists of a shock front moving radially outwards into the cold gas with speed $1/3$. The shock is of "infinite strength", since the sound speed satisfies $c = 0$ in the cold gas. The numerical solution is computed in the positive quadrant $[0, 1]^2$ and then reflected appropriately to yield the solution on all of $[-1, 1]^2$. Consequently, reflecting boundary conditions are employed on the x and y axes. The exact solution is used to enforce the boundary conditions at the boundaries $x = 1$ and $y = 1$ (see [13] for the details).

The Noh test is a difficult problem; among the numerical methods considered in [13], only the PPM and CFLFh schemes produce somewhat satisfactory solutions with sharp shock fronts, though the solutions are still noisy and have large errors at the origin due to the phenomenon of anomalous *wall-heating* [21,18,13,15]. In particular, the WENO scheme considered in [13] fails for this problem. Consequently, it is not so surprising that our grossly simplified WENO scheme also fails for this problem. That is to say, our stand-alone WENO scheme is unable to run until the final time $t = 2.0$; the solution develops noise in the region with the cold gas, which eventually causes a violation of the positivity of the density and, subsequently, blow-up of the solution. This is the case even for a very small time-step δt; using CFL $= 5.0 \times 10^{-3}$ still results in blow-up. We propose the use of the C-method and the noise detection and removal algorithm to deal with these issues.

8.1. Application of WENO-C-N to the Noh problem

We will apply the WENO-C-N scheme on a grid with 200×200 cells in the domain $[0, 1]^2$ with a time-step $\delta t = 5 \times 10^{-4}$, giving a CFL number of approximately 0.25. A modified noise detection algorithm is employed, which we now describe. The "input functions" in the noise detection procedure described in §6.3 are the (potentially noisy) horizontal and vertical velocities $\tilde{u}(x, y)$ and $\tilde{v}(x, y)$. The noise detection procedure for each of these functions produces two different "noise indicator functions", denoted by $\mathbb{1}^u_{\text{noise}}(x, y)$ and $\mathbb{1}^v_{\text{noise}}(x, y)$. The function $\mathbb{1}^u_{\text{noise}}(x, y)$ is then used to de-noise the velocity field $\tilde{u}$, while $\mathbb{1}^v_{\text{noise}}(x, y)$ is used to de-noise $\tilde{v}$. For the Noh problem, we exploit the radial symmetry available by using the

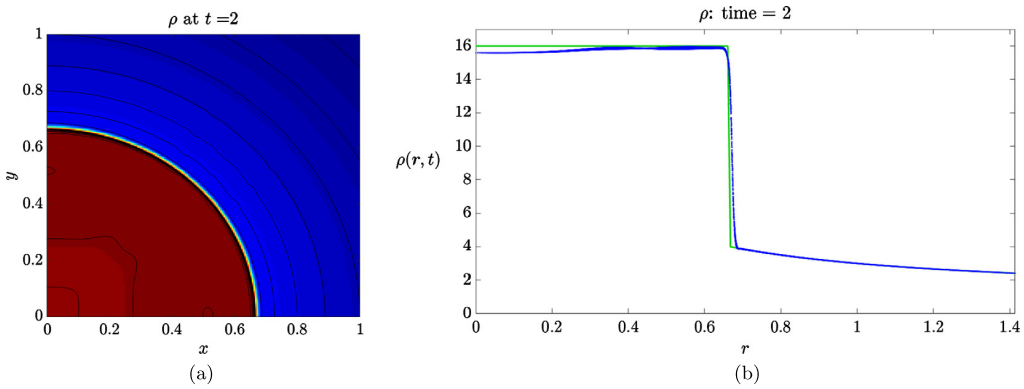


Fig. 10. Application of WENO-C-N to the Noh problem on the grid $[0,1]^2$ with 200×200 cells. Shown on the left is a heatmap of the density ρ at time $t = 2.0$. This is overlaid by 23 density contours, from 2.5 to 4 with step 0.25, and 14 to 17 with step 0.2. On the right is a scatter plot of the density versus radius. The green curve is the exact solution.

radial velocity $\tilde{u}_r(x,y) := \cos(\theta)\tilde{u}(x,y) + \sin(\theta)\tilde{v}(x,y)$ to produce a noise indicator function $\mathbb{1}^r_{\text{noise}}(x,y)$, where θ is the polar angle. The de-noising procedure for $\tilde{u}$ and $\tilde{v}$ then uses the function $\mathbb{1}^r_{\text{noise}}(x,y)$, but is otherwise identical to the algorithm described in §6.2 and §6.3.

The parameters for the WENO-C-N scheme are chosen as

$$\beta^u = 50.0, \qquad \beta^E = 350.0, \qquad \varepsilon = 200.0, \qquad \kappa = 0.5,$$
$$\eta \cdot \delta\tau/|\delta \boldsymbol{x}|^2 = 5 \times 10^{-2}, \qquad \delta h = 10^{-5}, \qquad \delta_{\text{off}} = 0.2\,.$$

The artificial viscosity term on the right-hand side of the energy equation (8d) serves to correct for the wall-heating error. The local heat equation solver for noise removal is iterated for a single time step only.

We provide in Fig. 10 a heatmap plot of the density computed using WENO-C-N, as well as a scatter plot of the density versus radius. We see that the space-time smooth artificial diffusion provided by the C-method stabilizes the strong shock wave and prevents spurious oscillations from developing behind the solution, while the artificial heat conduction term on the right-hand side of the energy equation (8d) significantly reduces the wall-heating error. The noise detection and removal procedure prevents high frequency noise from corrupting the solution and does not affect the sharpness of the shock front. Moreover, the solution maintains, for the most part, angular symmetry, though there are minor variations in the azimuthal direction; this should be contrasted with the results presented in [13], which show a much more obvious lack of symmetry by other schemes.

8.2. Comparison with Noh's artificial viscosity scheme

For the purposes of comparison, we also implement our WENO scheme together with a modification of Noh's artificial viscosity scheme [15], which is designed specifically for the Noh problem. The resulting scheme is referred to as WENO-Noh. Noh's scheme couples the classical artificial viscosity method of Von Neumann and Richtmeyer with a heat conduction term for the energy equation. In particular, the diffusion terms on the right-hand sides of (8b), (8c), and (8d) are replaced with the terms

$$\operatorname{div}\left(\tilde{\beta}^u_N \rho\, |\nabla u_r| \nabla u\right) + \tilde{\alpha}^u_N \Delta u\,, \tag{24a}$$
$$\operatorname{div}\left(\tilde{\beta}^u_N \rho\, |\nabla u_r| \nabla v\right) + \tilde{\alpha}^u_N \Delta v\,, \tag{24b}$$
$$\operatorname{div}\left(\tilde{\beta}^e_N \rho\, |\nabla u_r| \nabla e\right) + \tilde{\alpha}^e_N \Delta e\,, \tag{24c}$$

respectively, where u_r is the radial velocity, $e = p/\rho(\gamma - 1)$ is the internal energy of the system, and the artificial viscosity parameters $\tilde{\beta}_N$ and $\tilde{\alpha}_N$ are defined by

$$\tilde{\beta}_N = \frac{|\delta \boldsymbol{x}|^2}{\max_\Omega |\nabla u_r|}\beta_N \quad \text{and} \quad \tilde{\alpha}_N = |\delta \boldsymbol{x}|\alpha_N\,.$$

The parameter β_N controls the amount of *classical artificial viscosity* added to the system, while the parameter α_N controls the amount of *linear viscosity* added to the system. We refer the reader to [18] for a discussion on the differences between the diffusion terms used in WENO-Noh and the diffusion terms used in WENO-C-N.

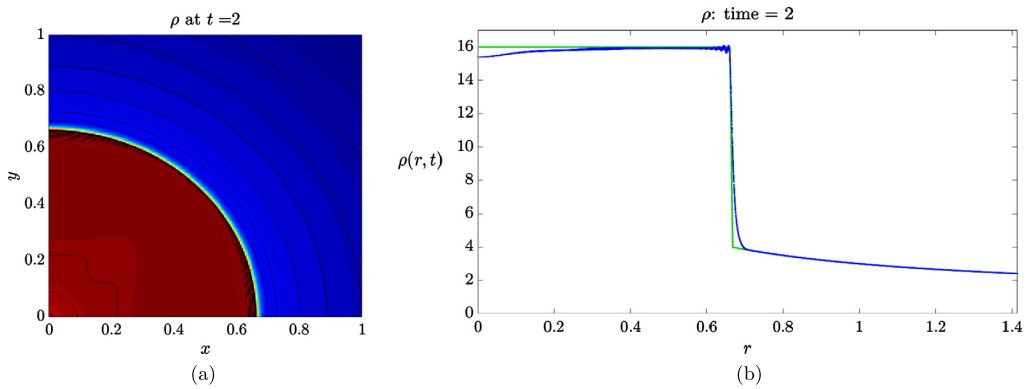


Fig. 11. Application of WENO-Noh to the Noh problem on the grid $[0,1]^2$ with 200×200 cells. Shown on the left is a heatmap of the density ρ at time $t = 2.0$. This is overlaid by 23 density contours, from 2.5 to 4 with step 0.25, and 14 to 17 with step 0.2. On the right is a scatter plot of the density versus radius. The green curve is the exact solution.

We implement the above WENO-Noh scheme for the Noh problem, with the following choices of viscosity parameters: $\beta_N^u = 50.0$, $\beta_N^e = 350.0$, $\alpha_N^u = 0.5$, and $\alpha_N^e = 1.5$. The results are shown in Fig. 11. The WENO-Noh scheme produces a solution that is, for the most part, oscillation-free; however, there are still some oscillations in the density profile behind the shock, and the shock curve is overly smeared. We remark here that the use of the linear viscosity term in (24) was needed to allow the WENO-Noh scheme to run; indeed, when that linear viscosity term was removed, the numerical simulation could not run as the solution blew-up. This is due primarily to the extremely oscillatory nature of the localizing function $|\nabla u_r|$; the additional linear viscosity stabilized the solution, at the cost of a very smeared shock curve and a loss of accuracy.

The solution produced with WENO-C-N has a much sharper shock front. In fact, a simple computation shows that $\max_\Omega |\nabla \rho(x,y,2)| \approx 856$ and $\max_\Omega |\nabla \tilde{\rho}(x,y,2)| \approx 774$, where ρ is the WENO-C-N solution for the density and $\tilde{\rho}$ is the WENO-Noh solution for the density. Consequently, we see that WENO-C-N produces a less oscillatory, more accurate solution with a sharper shock front.

9. The Sedov blast wave

The Sedov problem [22,31,1] models the self-similar evolution of a cylindrical blast wave, arising from a point-source explosion in a cold, uniform density fluid. The computational domain is $\Omega = [0, 1.2] \times [0, 1.2] \subset \mathbb{R}^2$. The initial density and velocity are, respectively, $\rho_0 = 1$, $u_0 = v_0 = 0$, and the initial energy is set to $E_0 = 10^{-12}$ everywhere except in the lower left corner cell, where it takes the value $\frac{0.244816}{\delta x \cdot \delta y}$. The adiabatic constant is $\gamma = 1.4$, reflecting boundary conditions are employed at the bottom and left boundaries, while inhomogeneous Dirichlet boundary conditions are enforced on the top boundary and the right boundary; in particular, the velocity, density, and energy are set equal to the corresponding values of the initial data on those boundaries.

The solution consists of an infinite strength radially symmetric shock front propagating outwards from the origin, leaving behind a near vacuum state. The simulation is run until time $t = 1.0$, at which time the shock front is a circle with radius equal to 1. A semi-exact solution exists for this problem [23,10,11], and we shall use the code made available by [33] to compute this solution. Our WENO and WENO-C-N schemes are employed on a grid with 96 cells in both directions, and with a time-step $\delta t = 10^{-4}$. This fixed time-step was chosen as the largest possible value for which the WENO simulation runs on the desired time interval. For the noise detection portion of our WENO-C-N scheme, we utilize the modified algorithm for radially symmetric flow, described in §8.1 for the Noh problem. The parameters for the C-N-method are as follows:

$$\beta^u = 1.0, \qquad \beta^E = 10.0, \qquad \varepsilon = 1.0, \qquad \kappa = 0.5,$$
$$\eta \cdot \delta\tau / |\delta \boldsymbol{x}|^2 = 10^{-2}, \qquad \delta h = 10^{-4}, \qquad \delta_{\text{off}} = 0.02\,.$$

The results are shown in Figs. 12 and 13. Our stand-alone WENO scheme produces a solution with inaccurate shock speed and location, and is corrupted by a large amount of high-frequency oscillations (or noise) behind the shock front. On the other hand, our WENO-C-N algorithm provides a solution with accurate wave speed and location, stabilizes the dynamics during early time-steps, and removes high-frequency oscillations behind the shock front.

We next conduct L^1 error analysis and convergence tests for the Sedov problem. Due to the fact that the exact solution is known only in radial coordinates, and our solution is defined on a rectangular mesh, we shall compute the errors of the density ρ and radial velocity u_r along the cut $y = 0$. We thus define the quantities $\tilde{\rho} = \rho(x, 0, t) - \rho^*(x, t)$ and $\tilde{u}_r = u_r(x, 0, t) - u_r^*(x, t)$, where ρ^* and u_r^* are the exact solutions. The L^1 norm for a function $f(x)$ defined on a one-dimensional computational grid of M cells with cell centers x_i is defined as

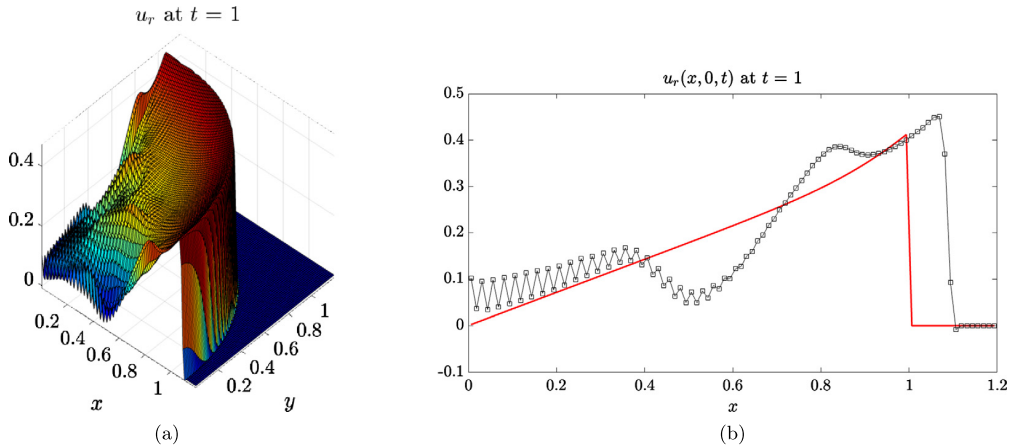


Fig. 12. Application of our standalone WENO scheme to the Sedov problem on the grid $[0, 1.2]^2$ with 96×96 cells. Shown are (a) a surface plot of the radial velocity u_r, (b) u_r along the cut $y = 0$. The red curves are the exact solutions.

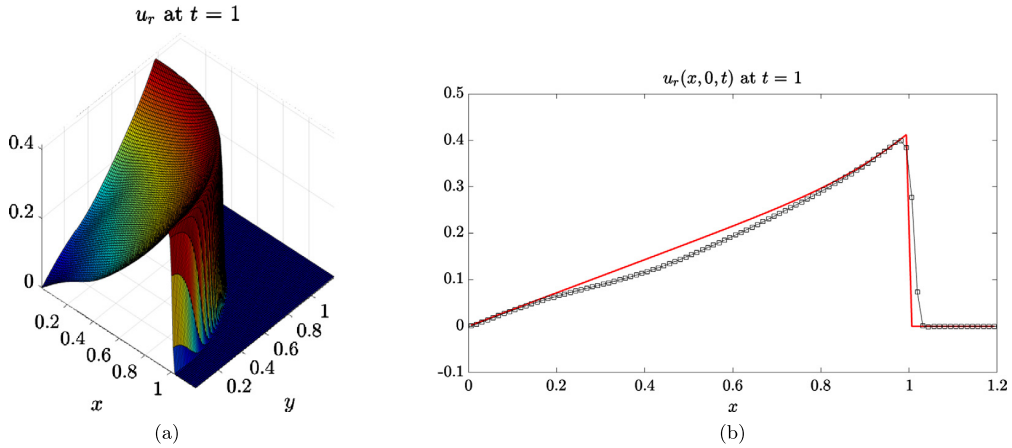


Fig. 13. Application of WENO-C-N to the Sedov problem on the grid $[0, 1.2]^2$ with 96×96 cells. Shown are (a) a surface plot of the radial velocity u_r, (b) u_r along the cut $y = 0$. The red curves are the exact solutions.

$$\|f\|_{L^1} = \frac{1}{M}\sum_{i=1}^{M} |f(x_i)| \,. \tag{25}$$

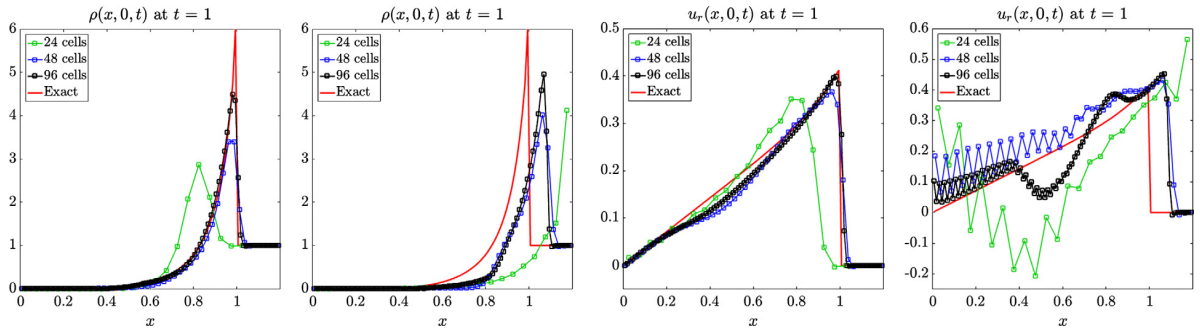
In Table 3, we list the L^1 norms and order of convergence for the errors $\tilde{\rho}$ and $\tilde{u}_r$. For the solution produced with the stand-alone WENO algorithm, the presence of the computational noise and the incorrect shock speed result in large errors and poor convergence rates.[4] The shock stabilization and noise removal provided by the WENO-C-N algorithm produces solutions with much smaller errors than those computed with stand-alone WENO, as well as better rates of convergence.

In Fig. 14, we show plots of the computed density ρ and radial velocity u_r along the cut $y = 0$ at various grid resolutions. It is clear from these plots, as well as the data in Table 3, that WENO-C-N produces solutions that are both quantitatively and qualitatively better than those produced with WENO.

[4] We note the "super-convergence" [8] of the WENO solutions on the coarse grids; this is due to large errors on coarser meshes, rather than smaller errors on finer meshes, and is therefore superficial.

Table 3
L^1 error analysis and convergence tests for the Sedov problem at $t = 1.0$.

Norm	Scheme		Cells		
			24	48	96
$\|\tilde{\rho}\|_{L^1}$	WENO	Error	6.347×10^{-1}	4.722×10^{-1}	4.648×10^{-1}
		Order	–	0.427	0.023
	WENO-*C*-*N*	Error	3.939×10^{-1}	1.081×10^{-1}	5.765×10^{-2}
		Order	–	1.866	0.907
$\|\tilde{u}_r\|_{L^1}$	WENO	Error	2.113×10^{-1}	8.993×10^{-2}	7.266×10^{-2}
		Order	–	1.232	0.308
	WENO-*C*-*N*	Error	4.695×10^{-2}	1.979×10^{-2}	1.482×10^{-2}
		Order	–	1.247	0.417



(a) WENO-*C*-*N* (b) WENO (c) WENO-*C*-*N* (d) WENO

Fig. 14. Plots of (a,b) the density and (c,d) the radial velocity along the cut $y = 0$. Subfigures (a) and (c) show the solutions computed with WENO-*C*-*N*, while Subfigures (b) and (d) show the solutions computed with WENO.

10. The Sod circular explosion problem

The explosion problem proposed in [34,13] is a radially symmetric version of the classic Sod shock tube problem [28].

$$\begin{bmatrix} \rho_0 \\ (\rho u)_0 \\ (\rho v)_0 \\ E_0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 2.5 \end{bmatrix} \mathbb{1}_{[0,0.4)}(r) + \begin{bmatrix} 0.125 \\ 0 \\ 0 \\ 0.25 \end{bmatrix} \mathbb{1}_{[0.4,\infty)}(r)\,, \tag{26}$$

where $r = \sqrt{x^2 + y^2}$, and $\mathbb{1}_\Sigma$ denotes the indicator function on the set Σ. We remark that the initial conditions are implemented in [13,34] by assigning area-weighted initial values for the cells which are crossed by the initial curve of discontinuity. We omit this modification, noting that an instability develops at the contact surface regardless of whether the area-weighted initial conditions are implemented. Since we wish to capture the evolution of the unstable contact, we do not smear the initial data, and instead employ our anisotropic artificial viscosity method.

Reflecting boundary conditions are employed on the x and y axes, and free-flow boundary conditions are employed at the boundaries $x = 1.5$ and $y = 1.5$. The free-flow boundary conditions are implemented using the characteristic form of the Euler equations so as to minimize the reflection from the outgoing waves (see [32] for the details). Nonetheless, there are numerical boundary effects occurring at the top and right boundaries that we were unable to eliminate.

The solution consists of a circular shock front and contact curve traveling outwards from the origin, and a circular expansion wave traveling inwards to the origin. The shock front and contact surface become weaker as time evolves, with the contact eventually coming to rest before traveling inwards, while the shock front passes through the free-flow boundary. The rarefaction wave traveling inwards collides with itself at the origin and reflects as an outwards traveling expansion wave. This results in an inward traveling shock forming and subsequently imploding into the origin. This shock then reflects outwards from the origin and eventually passes through the contact curve.

We shall demonstrate the ability of the *C*-method to allow the artificial viscosity operator associated with the shock wave to "communicate" with the artificial viscosity operator associated to the contact discontinuity. The objective of the scheme is to allow the shock wave to pass through the contact discontinuity, while leaving the small-scale KH structure of the contact undisturbed by over-diffusion. Our results compare favorably to those produced by PPM, CLAW, and WAFT [13].

10.1. The WENO-C-$\hat{C}$ scheme applied to the Sod explosion problem

We employ a combination of our WENO-C and WENO-C^τ schemes for this problem. More precisely, we use isotropic artificial viscosity (as provided by the WENO-C scheme) to stabilize shock fronts, and anisotropic tangential artificial viscosity (as provided by the WENO-C^τ scheme) to add diffusion to the unstable contact curve. We utilize a combination of compression and expansion switches to track the contact curve and shock fronts. In particular, we remark that, since shock implosion is a highly singular phenomenon, the shock that reflects from the origin requires stabilization. However, this shock passes through the contact curve, and consequently we ensure that the stabilization of the shock does not result in an overly diffused contact discontinuity. We do so by turning-off the artificial viscosity on the shock front as it passes through the contact curve by using compression and expansion switches (which are detailed below).

More precisely, we consider the following Euler-C-$\hat{C}$ system:

$$\partial_t \rho + \operatorname{div}(\rho \boldsymbol{u}) = 0\,, \tag{27a}$$

$$\partial_t(\rho u) + \partial_x(\rho u^2 + p) + \partial_y(\rho u v) = \operatorname{div}\left(\tilde{\beta}^u\, \rho\, C\, \nabla u\right) + \partial_i\left(\tilde{\alpha}\, \rho\, \hat{C}\, C^{\tau_i}\, C^{\tau_j}\, \partial_j u\right)\,, \tag{27b}$$

$$\partial_t(\rho v) + \partial_x(\rho u v) + \partial_y(\rho v^2 + p) = \operatorname{div}\left(\tilde{\beta}^u\, \rho\, C\, \nabla v\right) + \partial_i\left(\tilde{\alpha}\, \rho\, \hat{C}\, C^{\tau_i}\, C^{\tau_j}\, \partial_j v\right)\,, \tag{27c}$$

$$\partial_t E + \operatorname{div}(\boldsymbol{u}(E + p)) = \operatorname{div}\left(\tilde{\beta}^E\, \rho\, C\, \nabla(E/\rho)\right)\,, \tag{27d}$$

$$\partial_t C - \mathscr{L}[C\,;\varepsilon,\kappa] = \frac{S(\boldsymbol{u})}{\varepsilon|\delta\boldsymbol{x}|} G_\rho \tag{27e}$$

$$\partial_t \hat{C} - \mathscr{L}[\hat{C}\,;\varepsilon,\kappa] = \frac{S(\boldsymbol{u})}{\varepsilon|\delta\boldsymbol{x}|} \hat{G_\rho}\,, \quad \partial_t C^{\tau_i} - \mathscr{L}[C^{\tau_i}\,;\varepsilon,\kappa] = \frac{S(\boldsymbol{u})}{\varepsilon|\delta\boldsymbol{x}|}\hat{\tau}_i \text{ for } i = 1, 2\,. \tag{27f}$$

The forcing functions to the C-equations are given by

$$G_\rho = \left[1 - \mathbb{1}_{(-\infty,0)}(\partial_r e\, \partial_r \rho)\right] \cdot \mathbb{1}_{(-\infty,0)}(\operatorname{div}\boldsymbol{u}) \cdot |\nabla\rho|\,, \tag{28a}$$

$$\hat{G_\rho} = \mathbb{1}_{(-\infty,0)}(\partial_r e\, \partial_r \rho) \cdot |\nabla\rho|\,, \tag{28b}$$

$$\hat{\tau_1} = -\mathbb{1}_{(-\infty,0)}(\partial_r e\, \partial_r \rho) \cdot \partial_y \rho\,, \tag{28c}$$

$$\hat{\tau_2} = \mathbb{1}_{(-\infty,0)}(\partial_r e\, \partial_r \rho) \cdot \partial_x \rho\,, \tag{28d}$$

where $\partial_r = \cos\theta\, \partial_x + \sin\theta\, \partial_y$ denotes the radial derivative. The function $\mathbb{1}_{(-\infty,0)}(\operatorname{div}\boldsymbol{u})$ is a compression switch that localizes C to shocks, while the function $\mathbb{1}_{(-\infty,0)}(\partial_r e\, \partial_r \rho)$ localizes $\hat{C}$ and $\hat{\tau}_i$ to the contact curve. Consequently, the use of the function $\left[1 - \mathbb{1}_{(-\infty,0)}(\partial_r e\, \partial_r \rho)\right]$ in (28a) ensures that C deactivates during the short time interval that the shock front passes through the contact, so that isotropic diffusion is not added during this time interval, which prevents the smearing of the contact curve.

The artificial viscosity parameters $\tilde{\beta}$ and $\tilde{\alpha}$ are defined by

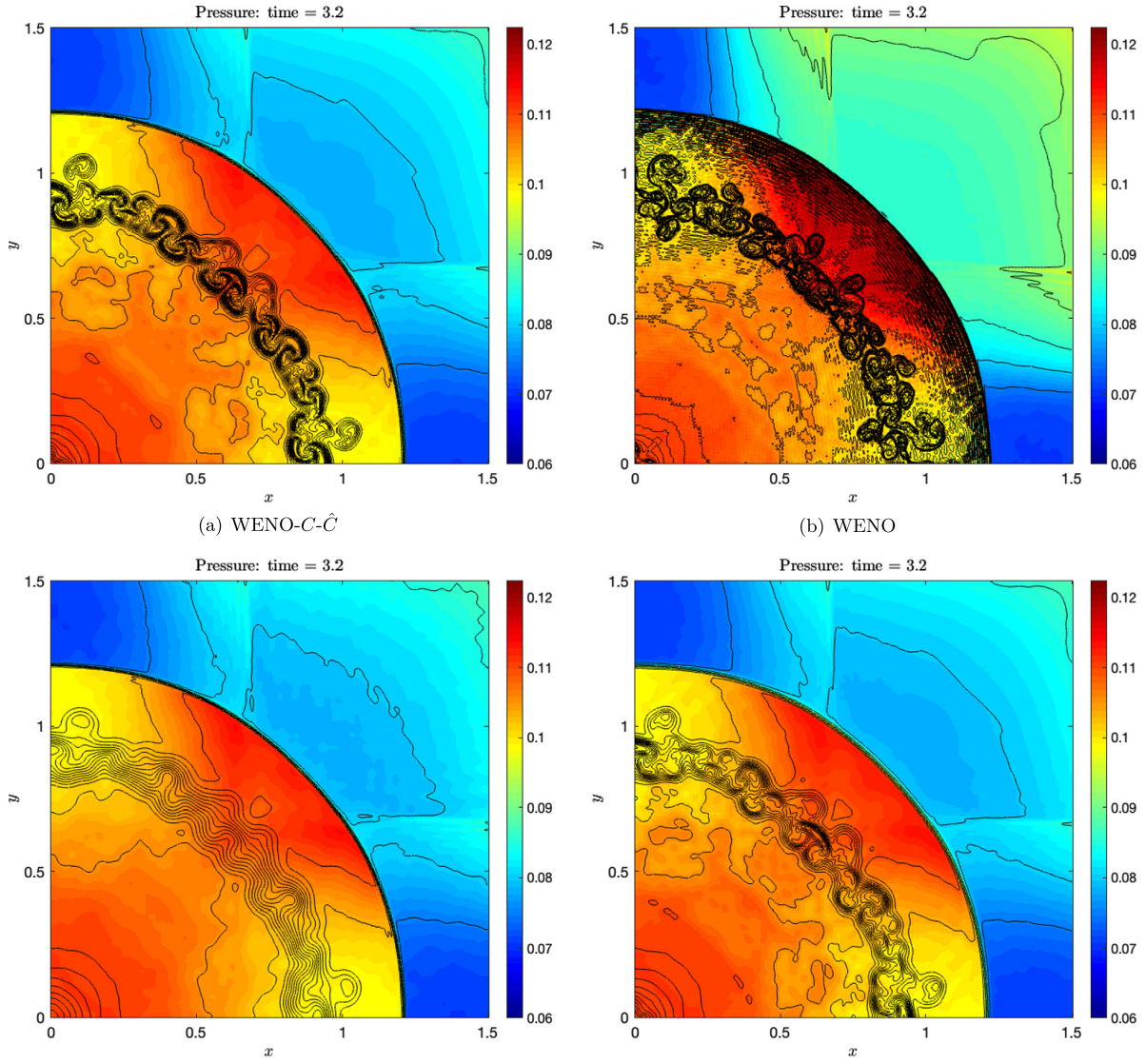
$$\tilde{\beta^\cdot} = \frac{|\delta\boldsymbol{x}|^2}{\max_\Omega C}\beta^\cdot \quad \text{and} \quad \tilde{\alpha} = \frac{|\delta\boldsymbol{x}|^2}{\mu^2 \max_\Omega \hat{C}}\alpha\,,$$

with $\mu = \max_\Omega\left\{\max\left\{|C^{\tau_1}|\,, |C^{\tau_2}|\right\}\right\}$.

The Euler-C-$\hat{C}$ system is numerically discretized in an identical fashion to the other schemes presented; we will refer to the discretized method as the WENO-C-$\hat{C}$ scheme. We employ the method to the problem on a grid with 400×400 cells with a time-step of $\delta t = 6.4 \times 10^{-4}$, giving an initial CFL number of approximately 0.75. The parameters for the WENO-C-$\hat{C}$ method are chosen as

$$\beta^u = 15.0, \qquad \beta^E = 200.0, \qquad \alpha = 2.0, \qquad \varepsilon = 1.0, \qquad \kappa = 10.0\,.$$

The results are presented in Fig. 15, which show heatmap plots of the pressure function p overlaid with density contours. These figures should be contrasted with those presented in [13]. The stand-alone WENO scheme produces a highly oscillatory solution behind the shock front (see Fig. 15(b)), whereas the C-method allows for the stabilization of the shock front. Additionally, we may see the role that the function $\left[1 - \mathbb{1}_{(-\infty,0)}(\partial_r e\, \partial_r \rho)\right]$ plays by comparing with the solution computed without the deactivation of C during shock-contact collision; this solution, shown in Fig. 15(c), is noticeably more smeared at the contact discontinuity. Finally, in Fig. 15(d) we show the solution computed using isotropic diffusion for both the contact discontinuity as well as shock fronts, with all the relevant parameters identical to those used in the WENO-C-$\hat{C}$ simulation. It is clear that the contact discontinuity is not as sharp as the contact curve for the solution computed using WENO-C-$\hat{C}$; this is due to the addition of diffusion in the direction normal to the contact curve. The WENO-C-$\hat{C}$ scheme produces a non-oscillatory solution with minimal noise, a sharp shock front, and a sharp contact curve.



(a) WENO-C-$\hat{C}$ (b) WENO

(c) WENO-C-$\hat{C}$ with C active during shock-contact collision (d) isotropic diffusion for the contact and shocks

Fig. 15. Comparison of WENO and WENO-C-$\hat{C}$ for the Sod explosion problem. The figures shown are heatmap plots of the pressure p overlaid with 27 density contours from 0.08 to 0.21 with step 0.005. Results are presented at time $t = 3.2$.

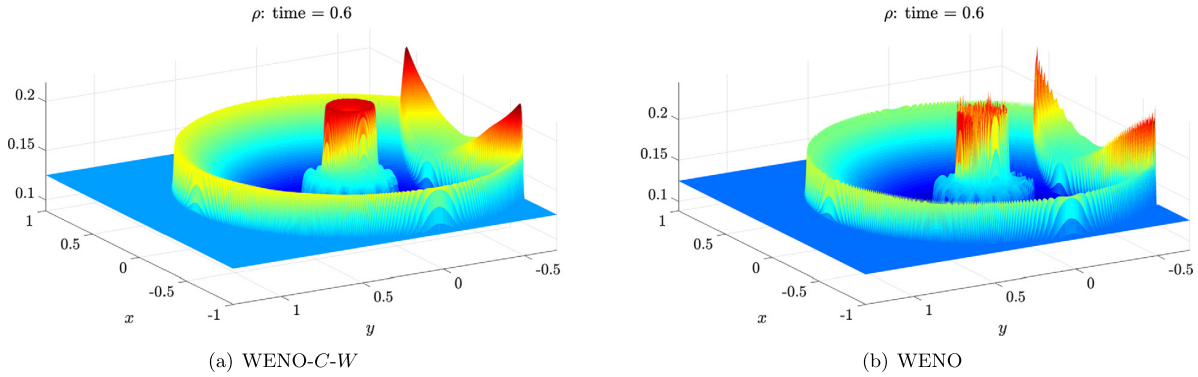
10.2. Shock-wall collision for a Sod-type explosion problem

To test the shock-collision scheme presented in §5, we consider a modified version of the Sod circular explosion problem. The domain is $\Omega = [-1, 1] \times [-0.7, 1.3] \subset \mathbb{R}^2$, the adiabatic constant is $\gamma = 1.4$, and the initial data is

$$\begin{bmatrix} \rho_0 \\ (\rho u)_0 \\ (\rho v)_0 \\ E_0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 2.5 \end{bmatrix} \mathbb{1}_{[0,0.1)}(r) + \begin{bmatrix} 0.125 \\ 0 \\ 0 \\ 0.25 \end{bmatrix} \mathbb{1}_{[0.1,\infty)}(r)\,, \tag{29}$$

where $r = \sqrt{x^2 + y^2}$, and $\mathbb{1}_\Sigma$ denotes the indicator function on the set Σ.

We are interested in the problem of shock-wall collision, and so treat the bottom boundary as a fixed wall, so that solid-wall (or reflecting) boundary conditions are enforced at $y = -0.7$. Free-flow (or symmetric) boundary conditions are implemented at the other three boundaries. The final time is $t = 0.6$. The outwards traveling shock front collides with the bottom boundary $y = -0.7$ at time $t \approx 0.4$. As the gas is compressed, an increase in the density leads to a reversal in the



(a) WENO-C-W (b) WENO

Fig. 16. Comparison of WENO and WENO-C-W for the Sod explosion and bounce-back problem. The figures shown are surface plots of the density ρ after shock-wall collision.

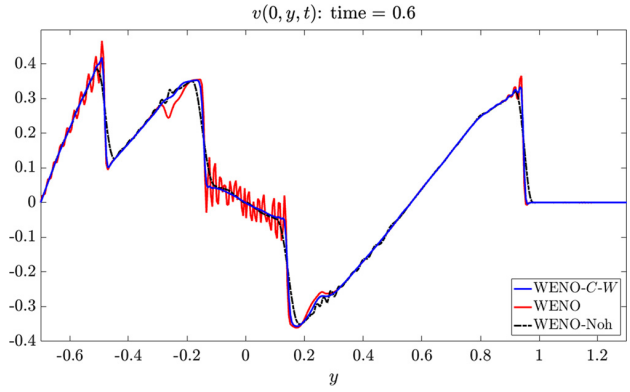


Fig. 17. Comparison of WENO, WENO-Noh, and WENO-C-W for the Sod explosion and bounce-back problem. Shown are cross-sections of the vertical velocity $v(0, y, t)$ along $x = 0$, at time $t = 0.6$ after the shock-wall collision.

direction of travel of the shock front. The collision with the bottom boundary breaks the radial symmetry of the problem; since the first point of contact of the shock front with the bottom boundary occurs at $x = 0$, the gas is forced outwards along the bottom boundary $y = -0.7$.

We employ our WENO-C-W scheme (see §5), which couples the C-method with a shock collision scheme. Due to the symmetry of the problem, we compute the solution in the half domain $[0, 1] \times [-0.7, 1.3]$, and then reflect appropriately to obtain the solution on all of Ω. The WENO-C-W scheme is applied on a grid with 200×400 cells in the half domain $[0, 1] \times [-0.7, 1.3]$, with a time-step $\delta t = 5 \times 10^{-4}$, giving a CFL number of approximately 0.4. The parameters in the WENO-C-W method are chosen as

$$\beta^u = 20.0, \qquad \beta^E = 0.0, \qquad \varepsilon = 1.0, \qquad \kappa = 1.0,$$
$$\beta^u_w = 50.0, \qquad \beta^E_w = 100.0, \qquad \varepsilon_w = 10.0, \qquad \kappa_w = 1.0\,.$$

For the purposes of comparison, we also implement our stand-alone WENO scheme as well as the WENO-Noh scheme with the following modification: due to the loss of radial symmetry when the shock front collides with the bottom boundary $y = 0$, we do not use the (normalized) gradient of the radial velocity $|\nabla u_r| / \max_\Omega |\nabla u_r|$ to track the shock front; instead, we use the function $\mathbb{1}_{\mathrm{div}\boldsymbol{u}<0} |\nabla \rho| / \max_\Omega |\nabla \rho|$, where $\mathbb{1}_{\mathrm{div}\boldsymbol{u}<0}$ is a compression switch. The artificial viscosity parameters in (24) are chosen as

$$\alpha^u_N = \alpha^e_N = 0 \quad \text{and} \quad \beta^u_N = 70.0\,, \beta^e_N = 200.0\,.$$

The results are provided in Fig. 16, which shows surface plots of the density, and Fig. 17, which is a plot of the vertical velocity $v(0, y, t)$ along $x = 0$.

The stand-alone WENO scheme produces an oscillatory solution both pre and post shock-wall collision. In particular, the shock implosion at the origin results in a large amount of noise (see Fig. 17). The solution computed using WENO-Noh is also oscillatory, albeit to a lesser extent, and the shock fronts are overly smeared. On the other hand, the WENO-C-W scheme produces a solution that is completely noise-free while retaining sharp shock fronts and correct wave speeds.

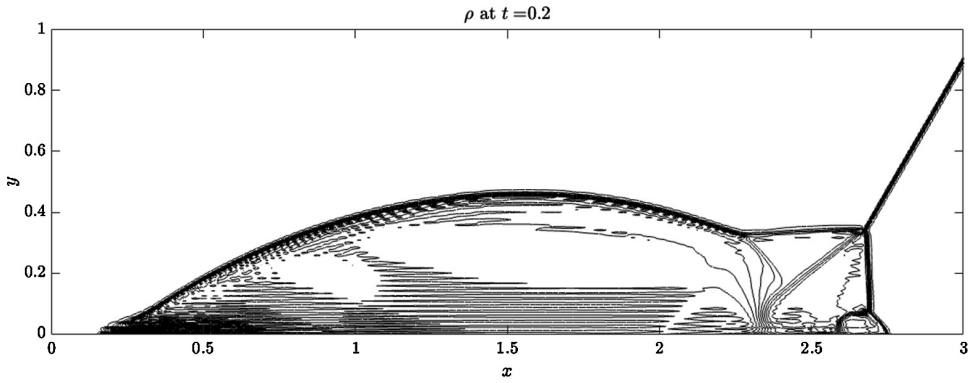


Fig. 18. A density contour plot of the stand-alone WENO solution computed on a grid with $\delta x = \delta y = 1/120$. Shown are 30 equally spaced density contours from $\rho = 1.5$ to $\rho = 22.9705$. The carbuncle instability leads to a kink at the leading shock front, causing a spurious triple point to form. There are high-frequency oscillations behind the reflected shock front, especially in the regions where shock curves intersect with the reflecting wall.

11. The Mach 10 shock reflection problem

The double Mach shock reflection problem introduced by Woodward & Colella [5] features a Mach 10 shock in a $\gamma = 1.4$ gas reflecting from a wedge inclined at an angle of 60°. The computational domain is $\Omega = [0, 3.25] \times [0, 1]$, and the initial data is given by

$$\begin{bmatrix} \rho_0 \\ (\rho u)_0 \\ (\rho v)_0 \\ E_0 \end{bmatrix} = \begin{bmatrix} 8 \\ 66\cos(\pi/6) \\ -66\sin(\pi/6) \\ 563.5 \end{bmatrix} \mathbb{1}_{\Sigma}(x, y) + \begin{bmatrix} 1.4 \\ 0 \\ 0 \\ 1.0 \end{bmatrix} \mathbb{1}_{\Omega\setminus\Sigma}(x, y)\,, \tag{30}$$

where $\Sigma = \left\{(x, y) \in \Omega \mid y > \sqrt{3}(x - 1/6)\right\}$ is the region behind the initial position of the shock. On the left boundary, the density, velocity, and energy are prescribed the values of the corresponding initial conditions, while free-flow boundary conditions are imposed at the right boundary. The conditions at the top boundary are set to describe the exact motion of the initial Mach 10 shock. At time t, the shock intersects with the top boundary $y = 1$ at the point $(s(t), 1)$, where $s(t) = 1/6 + (1 + 20t)/\sqrt{3}$. The pre- and post-shock conditions are then imposed for $x \geq s(t)$ and $x < s(t)$, respectively. At the bottom boundary, reflecting boundary conditions are prescribed, except for the short region $0 \leq x < 1/6$, along which the exact initial conditions are imposed. This condition forces the reflected shock to remain "attached" to the wedge.

The solution develops a complex self-similar flow structure featuring two triple points and a jet attached to bottom wall. Numerical methods applied to this problem often produce solutions suffering from the "carbuncle phenomenon" [16]. In such solutions, the wave speed of the leading Mach stem is incorrect, leading to a "kink" in the stem, and often the formation of a spurious additional triple point. In fact, our simplified WENO-type scheme produces a solution, shown in Fig. 18, exhibiting this error; moreover, the solution is corrupted by a large amount of high frequency noise, especially in the regions where the shock fronts meet with the reflecting wall.

To eliminate the carbuncle error and the high-frequency noise in the computed solution near the reflecting wall, we shall employ our WENO-C-W algorithm for this problem, with the following simple modification: since there is noise in the solution near the wall-jet, we do not use the compression switch $\mathbb{1}_{\mathrm{div}\boldsymbol{u}<0}$ in the C-equation (15e). This allows the C-function to remain active at the wall-jet, so that the artificial viscosity damps the high-frequency oscillations in this region. The discontinuities are stabilized through the use of directionally isotropic viscosity, and the wall C-method is used to implement additional viscosity in the regions where the shock fronts meet with the bottom reflecting boundary. We note that the compression switch $\mathbb{1}_{\mathrm{div}\boldsymbol{u}<0}$ remains active for the wall C equation (15f); this forces the wall function $\overline{C}(x, t)$ to vanish at the wall-jet, which subsequently prevents the over-smearing of the contact discontinuity.

We implement the WENO-C-W scheme, with the small modification mentioned above, on a grid with 781×241 cells, giving a mesh resolution of $\delta x = \delta y = 1/240$. The time-step is set as $\delta t = 4 \times 10^{-5}$, giving CFL ≈ 0.4. This time-step was chosen as the largest possible value for which the stand-alone WENO simulation runs until the final time $t = 0.2$. The relevant parameters for the WENO-C-W method are chosen as

$$\beta^u = 3 \times 10^2, \qquad \beta^E = 0.0, \qquad \varepsilon = 10.0, \qquad \kappa = 1.0,$$
$$\beta^u_w = 1 \times 10^3, \qquad \beta^E_w = 0, \qquad \varepsilon_w = 10.0, \qquad \kappa_w = 1.0\,.$$

A contour plot of the density computed using WENO-C-W is shown in Fig. 19. The artificial viscosity stabilizes the shock fronts and corrects the wave speeds, resulting in the elimination of the spurious carbuncle instability at the leading

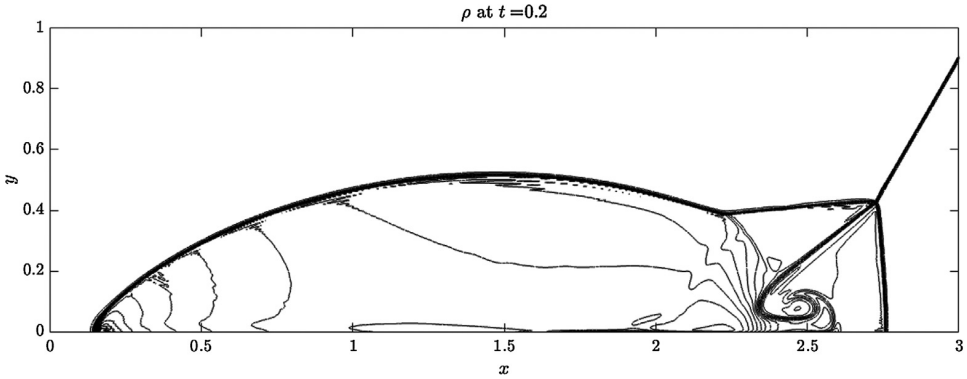


Fig. 19. A density contour plot of the WENO-C-W solution computed on a grid with $\delta x = \delta y = 1/240$. Shown are 30 equally spaced density contours from $\rho = 1.5$ to $\rho = 22.9705$.

shock. Moreover, the high-frequency oscillations at the intersections of shock fronts with the bottom boundary, and near the wall-jet, are suppressed by the additional viscosity provided by the wall C-method active in those regions. Our result is comparable to the simulations presented, for example, by Shi, Zhang, & Shu [25], for their developed WENO schemes.

12. The Rayleigh-Taylor instability

The Rayleigh-Taylor (RT) instability [17,30] is an instability of a heavy fluid layer supported by a light one. Specifically, RT occurs when a perturbed interface, between two fluids of different density, is subjected to a normal pressure gradient; the light fluid *bubbles* into the heavy fluid, while the heavy fluid *spikes* into the lighter fluid, which in turn initiates the Kelvin-Helmholtz (KH) instability. The RT instability occurs in a wide range of physical phenomena; see [24,12] and the references therein for an overview.

We use the following setup for the RT problem considered by Liska & Wendroff [13]: the domain is $\Omega = [-1/6, +1/6] \times [0, 1]$, the adiabatic constant is $\gamma = 1.4$, and the initial data is

$$\begin{bmatrix} \rho_0 \\ (\rho u)_0 \\ (\rho v)_0 \\ E_0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ p_0/(\gamma-1) \end{bmatrix} \mathbb{1}_{[0,h_0)}(y) + \begin{bmatrix} 2 \\ 0 \\ 0 \\ p_0/(\gamma-1) \end{bmatrix} \mathbb{1}_{[h_0,1]}(y)\,. \tag{31}$$

Here, the initial interface Γ_0 is parameterized by $(x, h_0(x))$ with $h_0(x) = 0.5 + 0.01\cos(6\pi x)$, and p_0 is the initial pressure, defined as

$$p_0 = \begin{cases} P_0 + g\,(h_0(x) - y) + 2g\,(1 - h_0(x)) & ,\ \text{if } y < h_0(x) \\ P_0 + 2g\,(1 - y) & ,\ \text{if } y \geq h_0(x) \end{cases}, \tag{32}$$

with $g = 0.1$ the gravitational acceleration, and P_0 the reference pressure. While Liska & Wendroff [13] do not provide P_0, we follow the ATHENA code [29] and use the value $P_0 = 2.4$ so as to produce a sound speed $c = \sqrt{3.5}$ in the lower density gas at the interface. The equations of motion governing the flow are given by the Euler system (1) with the addition of the gravity forcing on the right-hand side of (1a): $\boldsymbol{f} = [0\,, 0\,, -g\rho\,, -g\rho v]^{\mathcal{T}}$. Due to the symmetry of the problem, it is sufficient to calculate the solution in the half-domain $[0, 1/6] \times [0, 1]$, and then use reflection.

For a very short time, when the amplitude of the perturbation remains very small compared with the wavelength of the perturbation, the motion of the interface can be analyzed using linear stability analysis [17,30,3,14]. Quickly, however, the nonlinearity is activated, and the interface evolves to the classical "mushroom-shaped" profile [35]. The flow at later times is characterized by the development of "roll-up" regions, driven by the KH shear instability [12].

The particular problem set up described above results in a low Mach number flow. For explicit time-integration schemes, the CFL stability condition requires the time-step to be very small for such flows, due to the fact that, in this regime, sound waves are much faster than the advection of the flow. We do not switch to implicit time-integration, but note that one of the authors has extensively compared implicit and explicit temporal differencing for low Mach number flows and found little differences between the two approaches, provided the time step size is below the sound speed in the explicit approach; the time step size can be at least a factor of 10 bigger in the implicit approach. Likewise, for time-split methods, it has been shown that time-split errors grow as the time step begins to exceed the fastest time scale of the problem, i.e. the speed of sound waves. Hence, it is presumed for the current algorithm, in which the time steps utilized are below this fast time scale, that the temporal error growth is small. Moreover, we note that explicit time integration is used by Liska & Wendroff [13], and so for the purposes of comparison we utilize the same approach. We also note that the Navier-Stokes equations (rather

than the Euler equations) are often used for low Mach flow calculations; however, for the numerical tests presented here, the mesh resolution is so low that the kinematic viscosity coefficient appearing in the Navier-Stokes equations is negligible, and consequently algorithms for either sets of equations will produce similar results.

To highlight the applicability of our noise reduction algorithm to RT problems, we shall artificially generate noise in the solution by using explicit forward Euler time integration, rather than the usual third order Runge-Kutta method used for the other test problems. As we shall show in §12.2.2, the use of the explicit first order in time method results in the generation of spurious high frequency oscillations in solutions computed without any noise reduction algorithm. This test allows us to model a typical scenario in computational physics, in which high resolution and long time simulations require the use of an extremely small Δt and, consequently, a prohibitively large number of time steps. Our aim is to show that the noise generated by using a larger time step (or lower order time integration method) can be removed by means of our noise algorithm, resulting in a noise-free and accurate solution.

Numerical simulations of the RT instability often suffer from the development of spurious small-scale structure due to the discretization of the problem (to be described in more detail below); see [13] for example. On the one hand, as described in [13], the numerical methods with the least amount of implicit diffusion, such as the Piecewise Parabolic Method (PPM), Virginia Hydrodynamics 1 (VH1) and WENO schemes, produce interface break-up at early times which corrupts the solution (even if the initial density and pressure are smoothed over a few cells). The solutions produced using these schemes generally do not possess the classical mushroom-shaped profile; the spurious small-scale structures give rise to a complex interface and a significant amount of mixing. On the other hand, the more dissipative methods in [13], such as the scheme of Liu and Lax (LL) and the CFLF hybrid (CFLFh) scheme, suppress this small-scale spurious structure from developing but, in doing so, produce solutions with overly smeared interfaces and overly diffused mixing zones with very little KH roll-up.

Our aim, therefore, is to produce a noise-free solution with a sharp interface and the classical mushroom-shaped profile, while ensuring that the KH-driven roll-up regions are not significantly affected. We do so by implementing the *anisotropic* version of the C-method (that we detail below).

12.1. Tangential spikes and the need for anisotropic diffusion

We now consider numerical simulations of the RT problem with data given by (31). Let us motivate the need for strictly tangential artificial viscosity operators for the long-time motion of contact discontinuities. In Fig. 20, we show results of RT simulations, run with our simplified WENO scheme (without the use of any artificial viscosity); due to interpolation errors of the cosine function onto the uniform mesh, together with a lack of artificial viscosity, extremely large *tangential spikes* are produced in the vertical velocity, shown in Fig. 20(a). These oscillations are, of course, non-physical. The interpolation error is demonstrated in Fig. 20(b); the initial density profile has a tangential "staircase" whose jumps produce the tangential spikes in the vertical velocity, which in turn cause the interface to develop thin fingers as shown in Fig. 20(c). The fingers continue to grow and, in the absence of any artificial diffusion, eventually interact with each other and corrupt the solution as shown in Fig. 20(d).

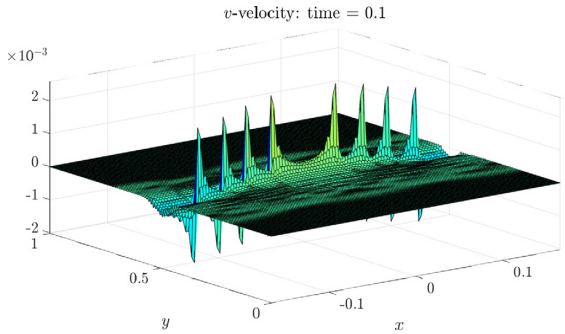
Liska and Wendroff [13] smooth the initial density and pressure functions over a region of width $\mathcal{O}(\delta y)$ in the vertical direction; using smoothed initial data mitigates the development of the tangential spikes (although not entirely), but has the undesirable effect of overly diffusing the contact discontinuity, and modifying wave speeds in a non-transient manner. Moreover, this smoothing is not enough to suppress the development of spurious small-scale structure for the highest order schemes with the least amount of numerical diffusion; in particular, as shown in [13], both WENO and PPM break-up the interface.

Our strategy, therefore, is to implement the explicit anisotropic diffusion term in (13) for use with our simplified WENO scheme. The tangential artificial viscosity operators smooth the solution only in tangential directions with no diffusion added in directions normal to the contact curve. This is extremely important for RT problems which take a very long time to fully develop (for example, in the runs presented here, 425,000 time-steps are used for the numerical simulation). The equations of motion are thus given by the Euler-C^τ system (13) with the addition of the gravity term $\boldsymbol{f} = [0\,, 0\,, -g\rho\,, -g\rho v]^{\mathcal{T}}$ to the right-hand side of the equations.

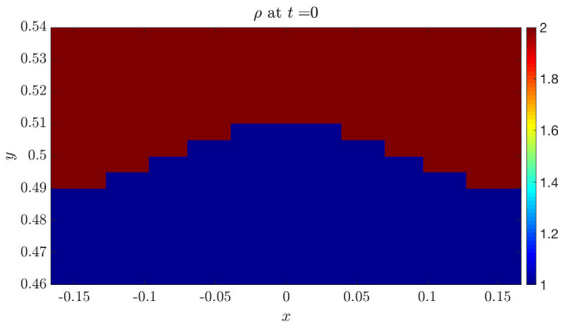
Fig. 21(a) and Fig. 21(b) show the effect of tangential artificial viscosity: the contact discontinuity remains sharp, while the spurious tangential spikes are suppressed.

12.1.1. Boundary conditions for the RT problem

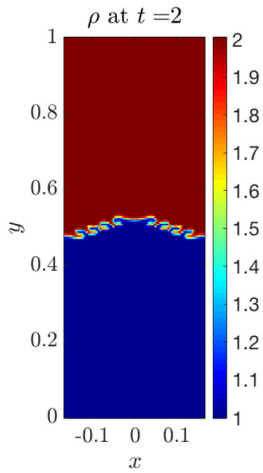
We implement reflecting boundary conditions for the RT problem on all four boundaries. Let $\vec{\nu}$ denote the normal vector to the boundary $\partial\Omega$. The reflecting boundary condition mean that $\vec{\nu} \cdot \boldsymbol{u}|_{\partial\Omega} = 0$ for all time $t \geq 0$. In the numerical discretization of the problem, this condition is enforced through the choice of *ghost node* values via an even or odd extension of each of the conservative variables. We provide further details in Appendix A.4, but mention here that the presence of the gravity terms in (13) requires that the pressure be extended in a linear fashion at the top and bottom boundaries so as to ensure that the vertical velocity v satisfies $v = 0$ there.



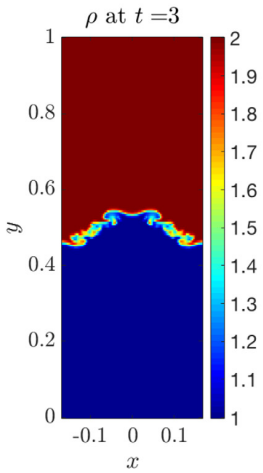
(a) tangential spikes in the vertical velocity v.



(b) "jumps" in the initial data for the density ρ. Figure is zoom-in on the initial interface Γ_0.

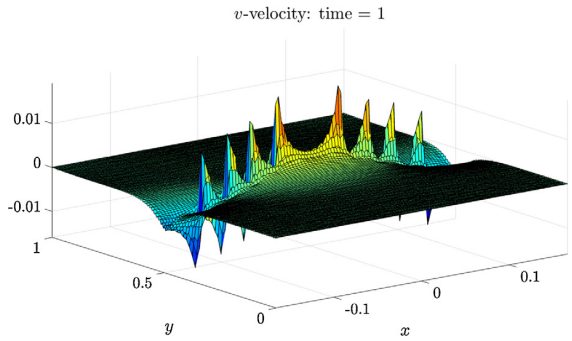


(c) development of fingers, caused by tangential spikes. Shown is a heatmap of the density ρ.

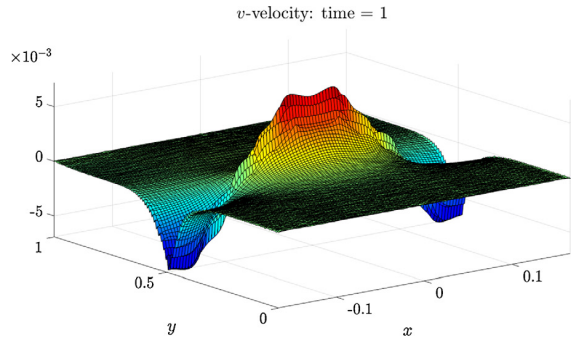


(d) mixing caused by interaction between fingers. Shown is a heatmap of the density ρ.

Fig. 20. Figures demonstrating the occurrence of tangential spikes due to the discretization of the initial conditions, which leads to the development of fingers and subsequent mixing. The solution is computed on a 50×200 grid on $[0, 1/6] \times [0, 1]$ using stand-alone WENO i.e. without any artificial diffusion.



(a) vertical velocity v computed with WENO-N at time $t = 1.0$; there are oscillations in the tangential direction.



(b) vertical velocity v computed with WENO-C^τ-N at time $t = 1.0$; the tangential spikes are removed.

Fig. 21. Demonstration of the removal of tangential spikes using anisotropic diffusion. Solutions are computed using WENO-N and WENO-C^τ-N in the domain $[0, 1/6] \times [0, 1]$ with 50×200 cells.

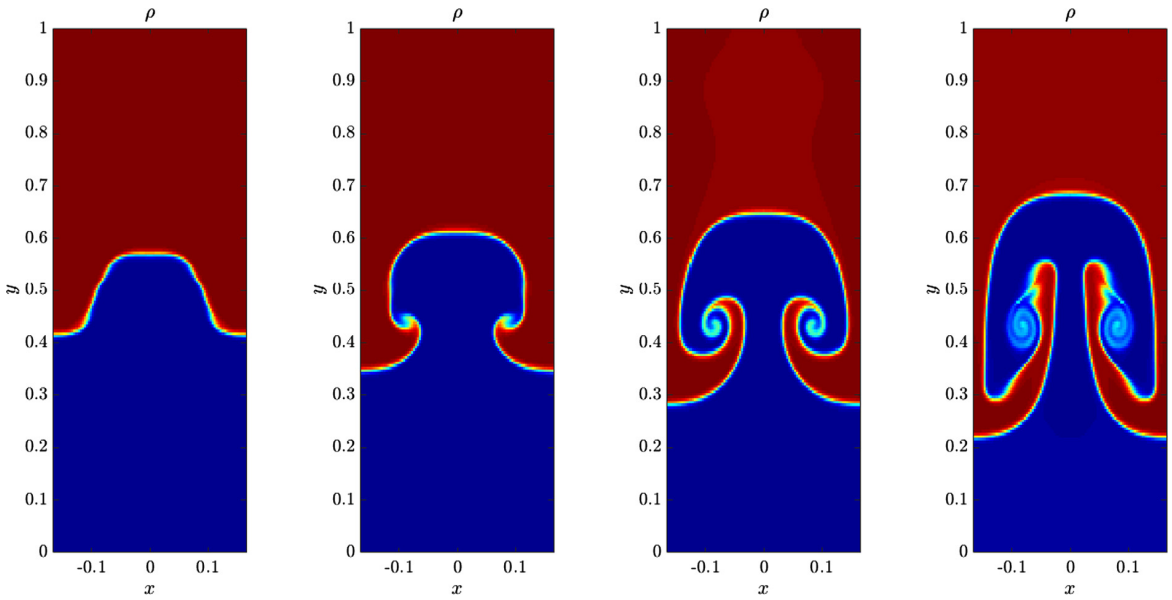


Fig. 22. Application of WENO-C^τ-N to the RT instability in the domain $[0, 1/6] \times [0, 1]$ with 50×200 cells; figures are heatmap plots of the density ρ at various instances of time.

12.2. Application of WENO-C^τ-N to the RT instability

We now apply the WENO-C^τ-N scheme (detailed in Appendix A.3) to the RT instability. The domain $[0, 1/6] \times [0, 1]$ is discretized using a 50×200 cell grid with a time-step of $\delta t = 2 \times 10^{-5}$, and the problem is run up to a final time of $t = 8.5$. The artificial viscosity parameter in (14) is set as $\beta = 20.0$, the C-equation parameters are chosen as $\varepsilon = 0.4$ and $\kappa = 10.0$, and we employ our noise detection and removal algorithm, with $\delta h = 2 \times 10^{-5}$ in (19), $\delta_{\text{off}} = 0.05$ in (18), the noise removal viscosity η in (21) chosen such that $\eta \cdot \delta\tau / |\delta \boldsymbol{x}|^2 = 5 \times 10^{-4}$, and the local heat equation solver iterated for only a single time-step.

The density heatmap at various instances of time are presented in Fig. 22. At time $t = 4.0$, the anisotropic diffusion acting in the tangential direction to the interface Γ prevents grid noise from developing into fingers and corrupting the solution, while ensuring that the interface Γ itself remains sharp, see Fig. 22(a). By time $t = 5.5$, the solution begins to assume the classical "mushroom"-shaped profile (Fig. 22(b)), and at time $t = 7.0$, the roll-up begins to occur, see Fig. 22(c). The anisotropic diffusion ensures that the solution in the roll-up region does not become overly diffused, and this is demonstrated in Fig. 22(d). This final figure should be compared with those figures in [13], which were produced with a finer mesh of 100×400 cells.

The WENO-C^τ-N solution shown in Fig. 22 compares very favorably with those in [13], which are either overly diffused (such as LL, CFLFh), or are corrupted by the small-scale structure at the interface (PPM, VH1). We note here that low resolution simulations generally exhibit a significant amount of mixing due to numerical diffusion [2]. This is in contrast with the results presented here, where the low resolution 50×200 WENO-C^τ-N simulation produces a solution with a sharper interface and more development in the roll-up region than the solutions obtained from the high resolution 100×400 simulations in [13].

12.2.1. Comparison with isotropic artificial viscosity

If we instead use isotropic artificial viscosity, we cannot produce an RT simulation which is as good as that produce by the anisotropic scheme. Fig. 23 shows the density profile for the RT simulation using the *isotropic* C-method as described by the Euler-C system (8) with $\beta^E = 0$ and $\beta^u = \beta^v = \beta$ with four different values of β.

The choice of $\beta = 1.0$ is insufficient to smooth the density oscillations in the tangential direction, so that the fingers develop, interact with each other and eventually corrupt the solution. Setting $\beta = 5.0$ removes the density oscillations and produces a solution that is closer to the classic mushroom-shaped profile, but still results in a depression at the top of the mushroom. Moreover, there is very little roll-up occurring, as can be seen from Fig. 23(b). Increasing to $\beta = 10.0$ removes the depression at the top of the mushroom, but results in almost no roll-up, see Fig. 23(c). In Fig. 23(d), we show the result computed with $\beta = 20.0$, which is the value of β used for the anisotropic C-method simulation shown Fig. 22. The isotropic solution with $\beta = 20$ shows no roll-up at all.

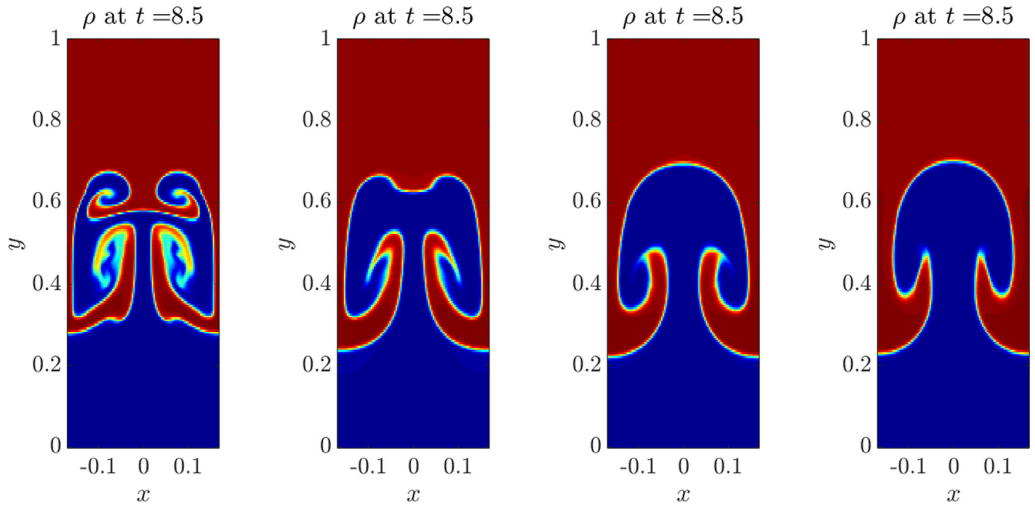


Fig. 23. Application of WENO-C-N (with isotropic viscosity) to the RT instability in $[0, 1/6] \times [0, 1]$ with 50×200 cells; figures are of the density ρ at the final time $t = 8.5$.

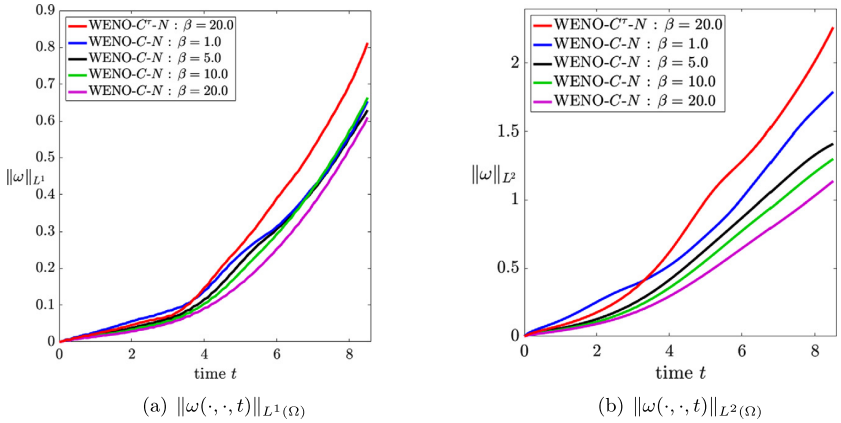


Fig. 24. Comparison of the L^1 and L^2 norms of vorticity ω for solutions computed with anisotropic or isotropic diffusion.

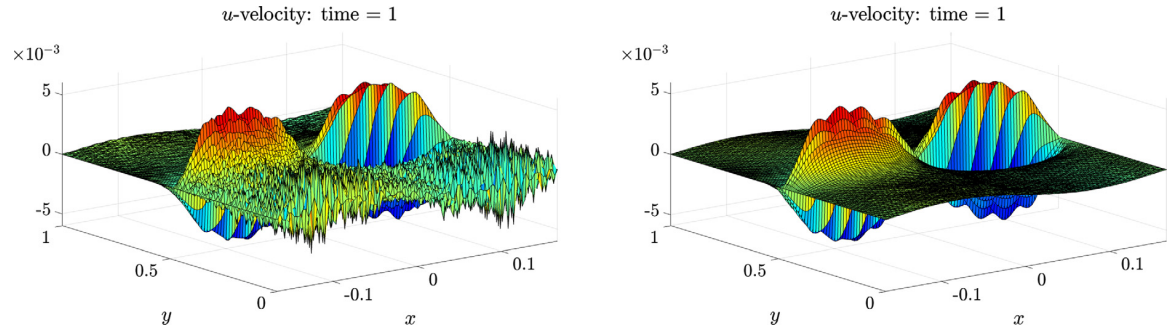
In Fig. 24, we compute the L^q norm (with $q = 1, 2$) of the *vorticity* $\omega := \partial_x v - \partial_y u$, which is a measure of the local rotation of the fluid, and hence provides a measure of the amount of interface roll-up in the flow. It is evident that the solution computed with the anisotropic C-method displays the largest growth (in time) of the norm $\|\omega(\cdot, \cdot, t)\|_{L^q(\Omega)}$. The simulation with isotropic diffusion and $\beta = 1.0$ has a distinct increase in $\|\omega(\cdot, \cdot, t)\|_{L^1(\Omega)}$ at around time $t = 4.5$. This is due to the fingers that develop and roll-up; the growth then slows down due to the interaction between the fingers which causes mixing. The isotropic $\beta = 5.0$, $\beta = 10.0$ and $\beta = 20.0$ simulations share similar profiles for $\|\omega(\cdot, \cdot, t)\|_{L^q(\Omega)}$, with all three displaying significantly less growth of vorticity than the simulation run using WENO-C^τ-N.

12.2.2. Suppression of noise with the noise detection and removal algorithm

In this section, we briefly discuss the role of the noise detection and removal algorithm described in §6 in suppressing noise that would otherwise occur and corrupt the solution. In Fig. 25, we compare the results from two simulations: one run using WENO-C^τ-N, which includes the noise detection and removal algorithm, and one run using WENO-C^τ, which does not include the noise detection and removal algorithm. The figures shown are of the horizontal velocity u at time $t = 1.0$. We recall that explicit forward Euler time integration is used for both simulations.

It is clear that the WENO-C^τ-N solution is much better than the WENO-C^τ solution. The noise that develops[5] corrupts the WENO-C^τ solution at time $t = 1.0$, as shown in Fig. 25(a). This is in contrast to the WENO-C^τ-N solution, which remains

[5] The use of explicit forward Euler time-stepping to simulate low Mach flow results in extremely severe stability constraints and, although the CFL condition is not violated, solutions are quickly corrupted by high-frequency noise.



(a) WENO-C^τ (b) WENO-C^τ-N

Fig. 25. Figures demonstrating the ability of the wavelet-based noise detection and removal algorithm to suppress high-frequency spurious noise that can occur in the solution. Figures are of the horizontal velocity u. Fig. 25(a) shows the solution computed with WENO-C^τ, while Fig. 25(b) shows the solution computed with WENO-C^τ-N. Both solutions were solved in $[0, 1/6] \times [0, 1]$ with 50×200 cells and explicit forward Euler time integration, and are shown at time $t = 1.0$.

noise-free. We note here that the WENO-C^τ-N solution is unaffected at the interface Γ, due to the deactivation of the noise detection near the interface.

For the purpose of benchmarking our noise detection and removal procedure, we compare the algorithm with a simple alternative, namely the addition of a linear viscosity term to the right-hand side of the momentum equations (13b) and (13c) of the form

$$\beta_l |\delta \boldsymbol{x}|^2 \Delta \boldsymbol{u}\,, \tag{33}$$

with β_l the linear viscosity parameter. The resulting scheme is referred to as the WENO-C^τ-$\Delta \boldsymbol{u}$ scheme.

Setting $\beta_l = 25.0$ corresponds to the choice of $\eta \cdot \delta\tau / |\delta \boldsymbol{x}|^2 = 5 \times 10^{-4}$, which is the value used for the simulation shown in Fig. 22. We present in Fig. 26 the solutions obtained when the noise detection and removal algorithm is replaced by the linear viscosity (33), with various choices of β_l.

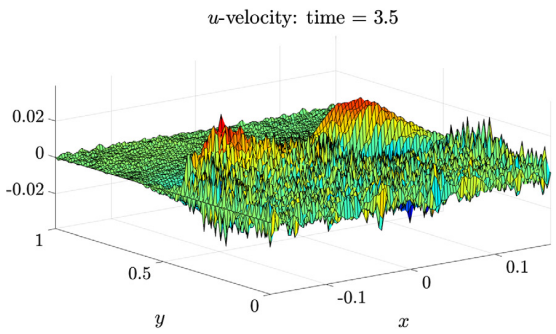
The issue with the uniform application of linear viscosity (33) is that the parameter β_l controlling the amount of diffusion needs to be large enough so as to suppress noise, while simultaneously small enough to allow some physical structure to develop in the KH mixing zones. For the RT problem considered here, this is not possible; for instance, the simulation run with $\beta_l = 25.0$, shown in Fig. 26(d), is overly diffused with no structure at all in the roll-up region. The same is true, albeit to a lesser extent, with the choice of $\beta_l = 3.0$, shown in Fig. 26(c). On the other hand, setting β_l too small by choosing, for example, $\beta_l = 1.0$ means that there is not enough viscosity to suppress the high frequency oscillations. This is demonstrated in Fig. 26(a). The same is true, again to a lesser extent, when β_l is set as $\beta_l = 2.0$, see Fig. 26(b). In this case, the noise is suppressed at the early stages, but begins to develop near the boundaries at later times, and again corrupts the solution. On the other hand, the use of the noise detection and removal algorithm allows for the use of a large amount of viscosity, due to the localized nature of the heat equation solver. The solution produced is noise-free, does not require the use of a smaller time-step, and retains the physical structure in the KH roll-up regions.

To quantitatively demonstrate the improvement in accuracy, we compare the solutions shown above with a "reference" solution $\boldsymbol{U}^*$. This reference solution is computed using WENO-C^τ with 50×200 cells and explicit third order Runge-Kutta time integration; the use of a higher order in time method results in a noise-free solution. We define the L^1 error norm of a computed quantity f as

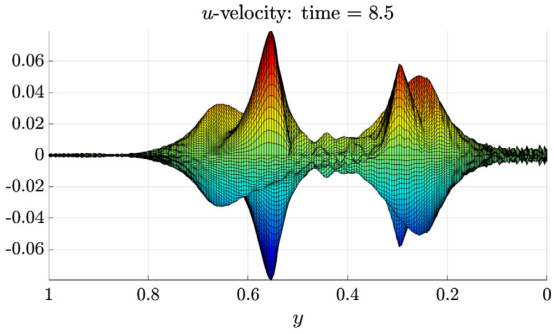
$$E_f(t) = \Delta x \cdot \Delta y \sum_{i=1}^{M} \sum_{j=1}^{N} |f(x_i, y_j, t) - f^*(x_i, y_j, t)|\,, \tag{34}$$

where f^* is the reference solution, and M and N are the number of cells in the x and y directions, respectively. For this test, $M = 50$ and $N = 200$.

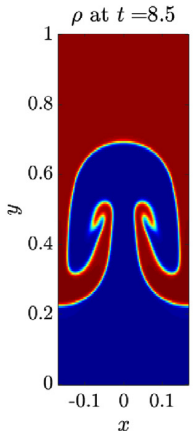
In Table 4, we provide $E_\rho(t)$, $E_u(t)$, and $E_v(t)$ at various times t for solutions computed using either WENO-C^τ, WENO-C^τ-N, or WENO-C^τ-$\Delta \boldsymbol{u}$. We see that the WENO-C^τ-N method produces solutions with smaller errors in all of the quantities and at each of the times t, which establishes the high accuracy of the method. Finally, we compare in Fig. 27 WENO-C^τ-N with WENO-C^τ-$\Delta \boldsymbol{u}$ for $\beta_l = 2$, which is the optimal artificial viscosity parameter value for the linear viscosity scheme. It is clear from the figure that the WENO-C^τ-N solution is much closer in appearance to the reference solution than the WENO-C^τ-$\Delta \boldsymbol{u}$ solution is; the latter solution is overly smeared in the mixing regions due to the uniform application of viscosity, whereas the former solution retains the KH roll-up and matches almost exactly with the reference solution. These qualitative and quantitative observations establish the superiority of the noise detection and removal algorithm over the simple linear viscosity method.



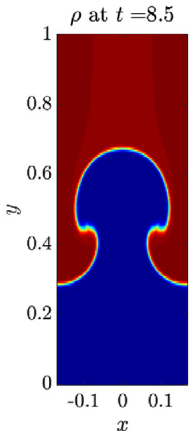
(a) $\beta_l = 1.0$: noise develops at early times. Shown is the horizontal velocity u



(b) $\beta_l = 2.0$: noise develops at late times, particularly near the bottom boundary $y = 0$; shown is a cross-section of the horizontal velocity u along $x = 0$



(c) $\beta_l = 3.0$: heatmap plot of the density ρ

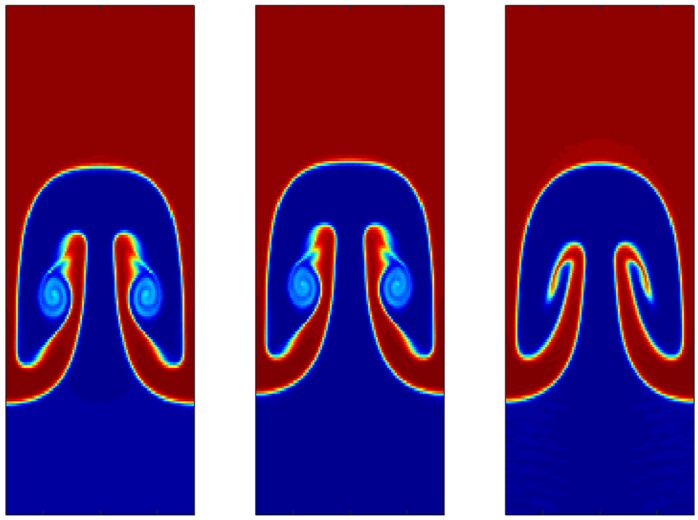


(d) $\beta_l = 25.0$: heatmap plot of the density ρ

Fig. 26. Figures showing solutions computed with linear viscosity replacing the noise detection and removal algorithm. Solutions computed in $[0, 1/6] \times [0, 1]$ with 50×200 cells using WENO-C^τ and a linear viscosity term to suppress the noise.

Table 4
L^1 error analysis for the RT problem at various times t.

Error	Scheme	**Time**		
		3.0	7.0	8.5
$E_\rho(t)$	WENO-C^τ-N	1.066×10^{-4}	6.286×10^{-3}	9.180×10^{-3}
	WENO-C^τ	5.812×10^{-3}	fail	fail
	WENO-C^τ-$\Delta \boldsymbol{u}$: $\beta_l = 1.0$	3.914×10^{-4}	8.033×10^{-3}	fail
	WENO-C^τ-$\Delta \boldsymbol{u}$: $\beta_l = 2.0$	4.475×10^{-4}	6.349×10^{-3}	9.222×10^{-3}
	WENO-C^τ-$\Delta \boldsymbol{u}$: $\beta_l = 3.0$	5.694×10^{-4}	7.197×10^{-3}	9.479×10^{-3}
	WENO-C^τ-$\Delta \boldsymbol{u}$: $\beta_l = 25.0$	1.852×10^{-3}	2.238×10^{-2}	2.952×10^{-2}
$E_u(t)$	WENO-C^τ-N	1.938×10^{-5}	5.753×10^{-4}	8.094×10^{-4}
	WENO-C^τ	5.569×10^{-3}	fail	fail
	WENO-C^τ-$\Delta \boldsymbol{u}$: $\beta_l = 1.0$	1.302×10^{-4}	2.691×10^{-3}	fail
	WENO-C^τ-$\Delta \boldsymbol{u}$: $\beta_l = 2.0$	6.694×10^{-5}	6.013×10^{-4}	8.819×10^{-4}
	WENO-C^τ-$\Delta \boldsymbol{u}$: $\beta_l = 3.0$	7.921×10^{-5}	6.205×10^{-4}	9.393×10^{-4}
	WENO-C^τ-$\Delta \boldsymbol{u}$: $\beta_l = 25.0$	2.040×10^{-4}	1.085×10^{-3}	1.160×10^{-3}
$E_v(t)$	WENO-C^τ-N	8.603×10^{-5}	1.046×10^{-3}	1.355×10^{-3}
	WENO-C^τ	4.794×10^{-3}	fail	fail
	WENO-C^τ-$\Delta \boldsymbol{u}$: $\beta_l = 1.0$	2.469×10^{-4}	2.776×10^{-3}	fail
	WENO-C^τ-$\Delta \boldsymbol{u}$: $\beta_l = 2.0$	1.015×10^{-4}	1.109×10^{-3}	1.480×10^{-3}
	WENO-C^τ-$\Delta \boldsymbol{u}$: $\beta_l = 3.0$	1.137×10^{-4}	1.172×10^{-3}	1.587×10^{-3}
	WENO-C^τ-$\Delta \boldsymbol{u}$: $\beta_l = 25.0$	2.651×10^{-4}	2.253×10^{-3}	2.820×10^{-3}



(a) WENO-C^τ-N (b) WENO-C^τ (c) WENO-C^τ-$\Delta \boldsymbol{u}$

Fig. 27. Comparison of (a) WENO-C^τ-N, (b) WENO-C^τ with 3rd order Runge-Kutta, and (c) WENO-C^τ-$\Delta \boldsymbol{u}$ with $\beta_l = 2.0$. Figures are of the density ρ at the final time $t = 8.5$.

13. Concluding remarks

This paper presents the two-dimensional generalization of the 1-D C-method described in [18]. The 2-D C-method is a space-time smooth artificial viscosity scheme for adding *isotropic* diffusion to shock curves and *anisotropic* diffusion to contact discontinuities. We have applied the isotropic C-method to four difficult problems, namely the Noh problem, the Sedov point blast test, the Sod circular explosion problem, and the Mach 10 reflection test, and demonstrated that our method compares favorably to stand-alone WENO as well as the WENO-Noh scheme, the latter using the artificial viscosity method of Noh [15].

We have also demonstrated the efficacy of the anisotropic C-method. For problems requiring long-time evolution of contact discontinuities with hundreds of thousands of time-steps, the *anisotropic* C-method works extremely well. This method was applied to the RT instability problem of Liska-Wendroff [13], and was shown to be able to suppress tangential spikes, while keeping the interface sharp and not overly diffusing the KH roll-up regions.

For problems with shock-contact interaction, our C-method uses a C-function to track the shock and another $\hat{C}$-function to track the contact discontinuity. These two functions interact with one another: when the shock interacts with the contact, the C tracking the shock goes to zero, allowing the $\hat{C}$ of the contact to stabilize the contact curve without dissipation from the shock front.

We have also described a simple shock-wall collision scheme for stabilizing shock fronts during wall collision and bounce-back, and demonstrated its ability to suppress post-collision noise for a Sod-type explosion problem and the double Mach 10 reflection test.

For high-frequency noise detection and removal, we have devised and implemented a simple 2-D wavelet-based scheme for detecting these oscillations followed by a highly localized (in both space and time) removal algorithm based on a local explicit heat equation solver. This noise detection and removal scheme has been applied to the RT and Noh problems, wherein it suppressed high-frequency noise while retaining sharp shock and contact fronts.

The qualitative and quantitative observations presented in this paper, together with the extensive accuracy studies presented in [18] in the 1-D case, indicate that the schemes are high-order and produce accurate solutions, even with the use of our highly simplified finite-difference WENO method.

Acknowledgements

Research reported in this publication was supported by the Office of Defense Nuclear Nonproliferation Research and Development and by the Defense Threat Reduction Agency under Interagency Agreement number HDTRA1825370 (DTRA10027 – 25370) as work for others. SS was partially supported by DTRA HDTRA11810022.

We would like to express our gratitude to the anonymous referees for their numerous suggestions that have greatly improved the manuscript.

Table 5
Various numerical schemes used in the simulations.

Scheme	Description
WENO	standard fifth-order WENO procedure for the usual Euler equations (1).
WENO-Noh	WENO scheme with Noh's artificial viscosity method (24).
WENO-C	WENO scheme with the *isotropic* C-method (8).
WENO-C^τ	WENO scheme with the *anisotropic* C-method (13).
WENO-C-N and WENO-C^τ-N	WENO-C and WENO-C^τ with the noise detection and removal algorithm described in §6.
WENO-C-$\hat{C}$	WENO scheme with the *isotropic* C-method for shock fronts and the *anisotropic* C-method for contact discontinuities (see §10.1).
WENO-C-W	WENO scheme with the shock-wall collision scheme (15).

Appendix A. Numerical discretization and schemes

In this section, we provide details for the WENO-based schemes used to produce the results in this paper. We first describe the simplified WENO procedure used for the spatial discretization of the nonlinear flux terms in the Euler equations. We then describe the implementation for the Euler-C and Euler-C^τ systems.

For readability, we will use Table 5 to refer to these various schemes.

A.1. WENO reconstruction procedure

Our WENO reconstruction procedure is formally fifth-order with upwinding performed based only on the sign of the velocity at the cell edges. In particular, no approximate Riemann solvers are used.

The spatial domain $\Omega = [x_1, x_M] \times [y_1, y_N]$ is subdivided into $M \cdot N$ cells, each with area $\delta x \cdot \delta y$. We employ the notation

$$x_{i+\frac{1}{2}} = \frac{x_i + x_{i+1}}{2} \quad \text{and} \quad y_{j+\frac{1}{2}} = \frac{y_j + y_{j+1}}{2} .$$

Any quantity w evaluated at a cell center (x_i, y_j) shall be denoted by $w_{i,j}$. Similarly, any quantity w evaluated at the center of the cell edge $(x_{i\pm\frac{1}{2}}, y_j)$ shall be denoted by $w_{i\pm\frac{1}{2},j}$, and at the cell edge $(x_i, y_{j\pm\frac{1}{2}})$ by $w_{i,j\pm\frac{1}{2}}$. The quantity w evaluated at the cell corners $(x_{i\pm\frac{1}{2}}, y_{j\pm\frac{1}{2}})$ is denoted by $w_{i\pm\frac{1}{2},j\pm\frac{1}{2}}$.

Given an array $(a_{r,s})$ corresponding to cell-center values of the quantity a, and arrays $f_{r\pm\frac{1}{2},s}$ and $g_{r,s\pm\frac{1}{2}}$ corresponding to cell-edge values of the quantities f and g, respectively, we define the $(i,j)^{\text{th}}$ component of the array $\text{WENO}\left(a_{r,s}, f_{r\pm\frac{1}{2},s}, g_{r,s\pm\frac{1}{2}}\right)$ by

$$\left[\text{WENO}\left(a_{r,s}, f_{r\pm\frac{1}{2},s}, g_{r,s\pm\frac{1}{2}}\right)\right]_{i,j} = \left[\text{WENO}_x\left(a_{r,s}, f_{r\pm\frac{1}{2},s}\right)\right]_{i,j} + \left[\text{WENO}_y\left(a_{r,s}, g_{r,s\pm\frac{1}{2}}\right)\right]_{i,j} ,$$

with

$$\left[\text{WENO}_x\left(a_{r,s}, f_{r\pm\frac{1}{2},s}\right)\right]_{i,j} = \frac{1}{\delta x}\left(\tilde{a}_{i+\frac{1}{2},j} f_{i+\frac{1}{2},j} - \tilde{a}_{i-\frac{1}{2},j} f_{i-\frac{1}{2},j}\right)$$
$$\left[\text{WENO}_y\left(a_{r,s}, g_{r,s\pm\frac{1}{2}}\right)\right]_{i,j} = \frac{1}{\delta y}\left(\tilde{a}_{i,j+\frac{1}{2}} g_{i,j+\frac{1}{2}} - \tilde{a}_{i,j-\frac{1}{2}} g_{i,j-\frac{1}{2}}\right) ,$$

where the cell-edge values $\tilde{a}_{i\pm\frac{1}{2},j}$ and $\tilde{a}_{i,j\pm\frac{1}{2}}$ are calculated using a standard fifth-order WENO reconstruction procedure (see [8,26]) with upwinding performed based on the sign of $f_{i\pm\frac{1}{2},j}$ and $g_{i,j\pm\frac{1}{2}}$, respectively.

Then, defining the vector $\boldsymbol{U} = [\rho, \rho u, \rho v, E]^{\mathcal{T}}$, we construct the operator $\mathcal{F}_{\text{WENO}}$ as

$$\left[\mathcal{F}_{\text{WENO}}(\boldsymbol{U}_{r,s})\right]_{i,j} = -\begin{bmatrix} \left[\text{WENO}\left(\rho_{r,s}, \hat{u}_{r\pm\frac{1}{2},s}, \hat{v}_{r,s\pm\frac{1}{2}}\right)\right]_{i,j} \\ \left[\text{WENO}\left((\rho u)_{r,s}, \hat{u}_{r\pm\frac{1}{2},s}, \hat{v}_{r,s\pm\frac{1}{2}}\right)\right]_{i,j} + \tilde{\partial}_{x,4} p_{i,j} \\ \left[\text{WENO}\left((\rho v)_{r,s}, \hat{u}_{r\pm\frac{1}{2},s}, \hat{v}_{r,s\pm\frac{1}{2}}\right)\right]_{i,j} + \tilde{\partial}_{y,4} p_{i,j} \\ \left[\text{WENO}\left((E+p)_{r,s}, \hat{u}_{r\pm\frac{1}{2},s}, \hat{v}_{r,s\pm\frac{1}{2}}\right)\right]_{i,j} \end{bmatrix} .$$

Here, we use the notation $\tilde{\partial}_{x,4} p_{i,j}$ and $\tilde{\partial}_{y,4} p_{i,j}$ to denote the fourth-order central difference approximation for $\partial_x p$ and $\partial_y p$, respectively, at the cell-center (x_i, y_j). The cell-edge velocities $\hat{u}_{i\pm\frac{1}{2},j}$ and $\hat{v}_{i,j\pm\frac{1}{2}}$ are calculated using a fourth-order averaging:

$$\hat{u}_{i-\frac{1}{2},j} = \frac{-u_{i-2,j} + 7u_{i-1,j} + 7u_{i,j} - u_{i+1,j}}{12},$$
$$\hat{v}_{i,j-\frac{1}{2}} = \frac{-v_{i,j-2} + 7v_{i,j-1} + 7v_{i,j} - v_{i,j+1}}{12}.$$

The operator $\mathcal{F}_{\text{WENO}}$ provides the discretization of the nonlinear flux terms in the Euler systems (1), (8), (13) and (15). Below, we describe the discretization and implementation of the various forms of artificial viscosity used in this paper. If no artificial viscosity is implemented, then we have the semi-discrete equations

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{U}_{i,j} = \left[\mathcal{F}_{\text{WENO}}(\boldsymbol{U}_{r,s})\right]_{i,j}$$

corresponding to the Euler system (1). Given the solution at a time-step $t_n = n \cdot \delta t$, denoted by $\boldsymbol{U}^n_{i,j}$, time integration is done using an explicit k^{th} order Runge-Kutta method:

$$\boldsymbol{U}^{n+1}_{i,j} = \text{RK}(\boldsymbol{U}^n_{i,j}, \left[\mathcal{F}_{\text{WENO}}(\boldsymbol{U}^n_{r,s})\right]_{i,j}; k). \tag{35}$$

We will refer to the scheme (35) as "WENO", or "stand-alone WENO".

A.2. The WENO-C and WENO-C-N schemes

We first describe the discretization of the Euler-C system (8). Given arrays $(w_{r,s})$, $(C_{r,s})$ and $(\rho_{r,s})$ corresponding to cell-center values of the quantities w, C and ρ, and a time-dependent function $\tilde{\beta}$, the $(i,j)^{\text{th}}$ component of the array $\text{DIFF}(w_{r,s}, C_{r,s}, \rho_{r,s}; \tilde{\beta})$ is defined as

$$\begin{aligned}\left[\text{DIFF}(w_{r,s}, C_{r,s}, \rho_{r,s}; \tilde{\beta})\right]_{i,j} :=& \frac{\tilde{\beta}}{\delta x}\left(\rho_{i+\frac{1}{2},j} C_{i+\frac{1}{2},j} \tilde{\partial} w_{i+\frac{1}{2},j} - \rho_{i-\frac{1}{2},j} C_{i-\frac{1}{2},j} \tilde{\partial} w_{i-\frac{1}{2},j}\right) \\ &+ \frac{\tilde{\beta}}{\delta y}\left(\rho_{i,j+\frac{1}{2}} C_{i,j+\frac{1}{2}} \tilde{\partial} w_{i,j+\frac{1}{2}} - \rho_{i,j-\frac{1}{2}} C_{i,j-\frac{1}{2}} \tilde{\partial} w_{i,j-\frac{1}{2}}\right),\end{aligned}$$

where

$$\tilde{\partial} w_{i+\frac{1}{2},j} = \frac{w_{i+1,j} - w_{i,j}}{\delta x} \quad \text{and} \quad \tilde{\partial} w_{i,j+\frac{1}{2}} = \frac{w_{i,j+1} - w_{i,j}}{\delta y},$$

and the notation $z_{i\pm\frac{1}{2},j}$ and $z_{i,j\pm\frac{1}{2}}$ denote the quantity evaluated at the cell edges $(x_{i\pm\frac{1}{2}}, y_j)$ and $(x_i, y_{j\pm\frac{1}{2}})$, respectively, using a standard averaging e.g.

$$z_{i+\frac{1}{2},j} = \frac{z_{i+1,j} + z_{i,j}}{2}.$$

The operator $\mathcal{D}_{\text{DIFF}}$ is then defined by

$$\left[\mathcal{D}_{\text{DIFF}}(\boldsymbol{U}_{r,s}, C_{r,s})\right]_{i,j} = \begin{bmatrix} 0 \\ \left[\text{DIFF}\left(u_{r,s}, C_{r,s}, \rho_{r,s}; \tilde{\beta}^u\right)\right]_{i,j} \\ \left[\text{DIFF}\left(v_{r,s}, C_{r,s}, \rho_{r,s}; \tilde{\beta}^u\right)\right]_{i,j} \\ \left[\text{DIFF}\left((E/\rho)_{r,s}, C_{r,s}, \rho_{r,s}; \tilde{\beta}^E\right)\right]_{i,j} \end{bmatrix},$$

where the artificial viscosity coefficients are given by (9).

We also define the operator $\mathcal{L}(C_{r,s}; \varepsilon, \kappa)$ by

$$\left[\mathcal{L}(C_{r,s}; \varepsilon, \kappa)\right]_{i,j} = \frac{S(\boldsymbol{u}_{r,s})}{\varepsilon |\delta \boldsymbol{x}|}([G_\rho]_{i,j} - C_{i,j}) + \kappa S(\boldsymbol{u}_{r,s}) |\delta \boldsymbol{x}| \left(\tilde{\partial} C_{i+\frac{1}{2},j} - \tilde{\partial} C_{i-\frac{1}{2},j} + \tilde{\partial} C_{i,j+\frac{1}{2}} - \tilde{\partial} C_{i,j-\frac{1}{2}}\right),$$

where $S(\boldsymbol{u}_{r,s})$ is given by (3).

The isotropic Euler-C system (8) is then approximated by the semi-discrete equations

$$\begin{cases} \frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{U}_{i,j} = \left[\mathcal{F}_{\mathrm{WENO}}(\boldsymbol{U}_{r,s}) + \mathcal{D}_{\mathrm{DIFF}}(\boldsymbol{U}_{r,s}, C_{r,s})\right]_{i,j}, \\ \frac{\mathrm{d}}{\mathrm{d}t}C_{i,j} = \left[\mathcal{L}(C_{r,s};\varepsilon,\kappa)\right]_{i,j}, \end{cases}$$

and time integration is again done using a k^{th} order Runge-Kutta method:

$$\boldsymbol{U}_{i,j}^{n+1} = \mathrm{RK}\left(\boldsymbol{U}_{i,j}^{n}, \left[\mathcal{F}_{\mathrm{WENO}}(\boldsymbol{U}_{r,s}^{n}) + \mathcal{D}_{\mathrm{DIFF}}(\boldsymbol{U}_{r,s}^{n}, C_{r,s}^{n})\right]_{i,j}; k\right), \tag{36a}$$

$$C_{i,j}^{n+1} = \mathrm{RK}\left(C_{i,j}^{n}, \left[\mathcal{L}(C_{r,s}^{n};\varepsilon,\kappa)\right]_{i,j}; k\right). \tag{36b}$$

We will refer to the scheme (36) as the WENO-C scheme. When coupled with the noise detection and removal algorithm as detailed in §6.3, the scheme will be referred to as the WENO-C-N scheme.

A.3. The WENO-C^τ and WENO-C^τ-N schemes

Next, we describe the discretization procedure for the Euler-C^τ system (13). The anisotropic diffusion operator DIFF_τ is defined by

$$\begin{aligned}
&\left[\mathrm{DIFF}_\tau(w_{r,s}, C_{r,s}, C_{r,s}^{\tau_1}, C_{r,s}^{\tau_2}, \rho_{r,s}; \tilde{\beta})\right]_{i,j} \coloneqq \\
&\frac{\tilde{\beta}}{\delta x}\left(\rho_{i+\frac{1}{2},j} C_{i+\frac{1}{2},j} C^{\tau_1}_{i+\frac{1}{2},j} C^{\tau_1}_{i+\frac{1}{2},j} \tilde{\partial} w_{i+\frac{1}{2},j} - \rho_{i-\frac{1}{2},j} C_{i-\frac{1}{2},j} C^{\tau_1}_{i-\frac{1}{2},j} C^{\tau_1}_{i-\frac{1}{2},j} \tilde{\partial} w_{i-\frac{1}{2},j}\right) \\
&+\frac{\tilde{\beta}}{\delta y}\left(\rho_{i,j+\frac{1}{2}} C_{i,j+\frac{1}{2}} C^{\tau_2}_{i,j+\frac{1}{2}} C^{\tau_2}_{i,j+\frac{1}{2}} \tilde{\partial} w_{i,j+\frac{1}{2}} - \rho_{i,j-\frac{1}{2}} C_{i,j-\frac{1}{2}} C^{\tau_2}_{i,j-\frac{1}{2}} C^{\tau_2}_{i,j-\frac{1}{2}} \tilde{\partial} w_{i,j-\frac{1}{2}}\right) \\
&+\frac{\tilde{\beta}}{\delta x}\left(\rho_{i+\frac{1}{2},j} C_{i+\frac{1}{2},j} C^{\tau_1}_{i+\frac{1}{2},j} C^{\tau_2}_{i+\frac{1}{2},j} \hat{\partial} w_{i+\frac{1}{2},j} - \rho_{i-\frac{1}{2},j} C_{i-\frac{1}{2},j} C^{\tau_1}_{i-\frac{1}{2},j} C^{\tau_2}_{i-\frac{1}{2},j} \hat{\partial} w_{i-\frac{1}{2},j}\right) \\
&+\frac{\tilde{\beta}}{\delta y}\left(\rho_{i,j+\frac{1}{2}} C_{i,j+\frac{1}{2}} C^{\tau_1}_{i,j+\frac{1}{2}} C^{\tau_2}_{i,j+\frac{1}{2}} \hat{\partial} w_{i,j+\frac{1}{2}} - \rho_{i,j-\frac{1}{2}} C_{i,j-\frac{1}{2}} C^{\tau_1}_{i,j-\frac{1}{2}} C^{\tau_2}_{i,j-\frac{1}{2}} \hat{\partial} w_{i,j-\frac{1}{2}}\right),
\end{aligned}$$

where $\hat{\partial} w_{i\pm\frac{1}{2},j}$ and $\hat{\partial} w_{i,j\pm\frac{1}{2}}$ are approximations to the derivatives $\partial_y w$ at $(x_{i\pm\frac{1}{2}}, y_j)$ and $\partial_x w$ at $(x_i, y_{j\pm\frac{1}{2}})$, respectively:

$$\hat{\partial} w_{i\pm\frac{1}{2},j} = \frac{w_{i\pm\frac{1}{2},j+\frac{1}{2}} - w_{i\pm\frac{1}{2},j-\frac{1}{2}}}{\delta y},$$

$$\hat{\partial} w_{i,j\pm\frac{1}{2}} = \frac{w_{i+\frac{1}{2},j\pm\frac{1}{2}} - w_{i-\frac{1}{2},j\pm\frac{1}{2}}}{\delta x}.$$

We use the notation $w_{i\pm\frac{1}{2},j\pm\frac{1}{2}}$ to mean the quantity w evaluated at the cell corner $(x_{i\pm\frac{1}{2}}, y_{j\pm\frac{1}{2}})$ using a simple averaging. For example,

$$w_{i+\frac{1}{2},j+\frac{1}{2}} = \frac{w_{i,j} + w_{i+1,j} + w_{i,j+1} + w_{i+1,j+1}}{4}.$$

Defining the vector $\boldsymbol{C} = [C, C^{\tau_1}, C^{\tau_2}]^{\mathcal{T}}$, we construct the operator $\mathcal{D}^{\tau}_{\mathrm{DIFF}}$ as

$$\left[\mathcal{D}^{\tau}_{\mathrm{DIFF}}(\boldsymbol{U}_{r,s}, \boldsymbol{C}_{r,s})\right]_{i,j} = \begin{bmatrix} 0 \\ \left[\mathrm{DIFF}_\tau\left(u_{r,s}, C_{r,s}, C^{\tau_1}_{r,s}, C^{\tau_2}_{r,s}, \rho_{r,s}; \tilde{\beta}\right)\right]_{i,j} \\ \left[\mathrm{DIFF}_\tau\left(v_{r,s}, C_{r,s}, C^{\tau_1}_{r,s}, C^{\tau_2}_{r,s}, \rho_{r,s}; \tilde{\beta}\right)\right]_{i,j} \\ 0 \end{bmatrix},$$

where the artificial viscosity coefficient $\tilde{\beta}$ is defined by (14).

The anisotropic Euler-C^τ system (13) may then be approximated by the semi-discrete equations

$$\begin{cases} \frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{U}_{i,j} = \left[\mathcal{F}_{\mathrm{WENO}}(\boldsymbol{U}_{r,s}) + \mathcal{D}^{\tau}_{\mathrm{DIFF}}(\boldsymbol{U}_{r,s}, \boldsymbol{C}_{r,s})\right]_{i,j}, \\ \frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{C}_{i,j} = \left[\mathcal{L}(\boldsymbol{C}_{r,s};\varepsilon,\kappa)\right]_{i,j}, \end{cases}$$

and time-integration is done using a k^{th} order Runge-Kutta solver:

$$\boldsymbol{U}_{i,j}^{n+1} = \mathrm{RK}\left(\boldsymbol{U}_{i,j}^{n}, \left[\mathcal{F}_{\mathrm{WENO}}(\boldsymbol{U}_{r,s}^{n}) + \mathcal{D}_{\mathrm{DIFF}}^{\tau}(\boldsymbol{U}_{r,s}^{n}, \boldsymbol{C}_{r,s}^{n})\right]_{i,j}; k\right), \tag{37a}$$

$$\boldsymbol{C}_{i,j}^{n+1} = \mathrm{RK}\left(\boldsymbol{C}_{i,j}^{n}, \left[\mathcal{L}(\boldsymbol{C}_{r,s}^{n}; \varepsilon, \kappa)\right]_{i,j}; k\right). \tag{37b}$$

We refer to the anisotropic scheme (37) as the WENO-C^{τ} scheme or, when coupled with the noise detection and removal algorithm in §6.3, as the WENO-C^{τ}-N scheme.

A.4. Boundary conditions and ghost node values

In general, the boundary conditions for the various problems considered here are imposed through the assigning of values to the *ghost nodes*. Generally, either an even extension or an odd extension of the variable in question is employed. For our (formally) fifth-order WENO reconstruction procedure, 7 nodes in each direction are used to reconstruct the flux at a cell-center, so that three ghost node values are required in each direction. By an even extension at the left and right boundaries of the variable w, we mean that

$$w_{1-i,j} = w_{1+i,j} \quad \text{and} \quad w_{M+i,j} = w_{M-i,j} \quad \text{for } i = 1, 2, 3,$$

while an odd extension at the left and right boundaries means

$$w_{1-i,j} = -w_{1+i,j} \quad \text{and} \quad w_{M+i,j} = -w_{M-i,j} \quad \text{for } i = 1, 2, 3.$$

An analogous identity holds for the top and bottom boundaries.

A.4.1. Boundary conditions for the RT problem

The numerical solution to the Rayleigh-Taylor problem 12 is computed on the half-domain $[0, 1/6] \times [01]$ and then reflected appropriately to yield the solution on all of $\Omega = [-1/6, 1/6] \times [0, 1]$. Consequently, reflecting boundary conditions are used on the left boundary $x = 0$ and right boundary $x = 1/6$. These are imposed through the ghost-node values; an even extension of ρ, E, ρv, C, and C^{τ_1} is imposed, while an odd extension of ρu and C^{τ_2} is enforced. At the top and bottom boundaries, ρ, ρu, C, C^{τ_1}, and C^{τ_2} are extended in an even fashion, ρv is extended in an odd fashion, and the pressure p is extended linearly so as to preserve hydrostatic equilibrium:

$$p_{i,1-j} = 2p_{i,1-j+1} - p_{i,1-j+2} \quad \text{and} \quad p_{i,N+j} = 2p_{i,N+j-1} - p_{i,N+j-2}$$

for $j = 1, 2, 3$ and $i = 1, \ldots, M$.

References

[1] H.A. Bethe, K. Fuchs, J.O. Hirschfelder, J.L. Magee, R.E. Peierls, J. von Neumann, Blast Wave, Los Alamos Scientific Laboratory Report LA-2000, 1947.
[2] A.C. Calder, et al., On validating an astrophysical simulation code, Astrophys. J. Suppl. Ser. 143 (1) (2002) 201.
[3] S. Chandrasekhar, Hydrodynamic and Hydromagnetic Stability, Dover Books Phys. Ser., Dover Publications, 1961.
[4] D. Coutand, S. Shkoller, On the finite-time splash and splat singularities for the 3-D free-surface Euler equations, Commun. Math. Phys. 325 (1) (2014) 143–183.
[5] P. Colella, P.R. Woodward, The numerical simulation of two-dimensional fluid flow with strong shocks, J. Comput. Phys. 54 (1984) 115–173.
[6] M. Crandall, A. Majda, The method of fractional steps for conservation laws, Numer. Math. 34 (1980) 285–314.
[7] C. Hu, C.-W. Shu, Weighted essentially non-oscillatory schemes on triangular meshes, J. Comput. Phys. 150 (1999) 97–127.
[8] G.-S. Jiang, C.-W. Shu, Efficient implementation of weighted ENO schemes, J. Comput. Phys. 126 (1996) 202–228.
[9] G.-S. Jiang, E. Tadmor, Nonoscillatory central schemes for multidimensional hyperbolic conservation laws, SIAM J. Sci. Comput. 19 (1998) 1892–1917.
[10] J.R. Kamm, Evaluation of the Sedov-von Neumann-Taylor Blast Wave Solution, Technical Report LA-UR-00-6055, Los Alamos National Laboratory, 2000.
[11] James R. Kamm, F.X. Timmes, On Efficient Generation of Numerically Robust Sedov Solutions, Technical report, 2007.
[12] H.J. Kull, Theory of the Rayleigh-Taylor instability, Phys. Rep. 206 (5) (1991) 197–325.
[13] R. Liska, B. Wendroff, Comparison of several difference schemes on 1D and 2D test problems for the Euler equations, SIAM J. Sci. Comput. 25 (2003) 995–1017.
[14] D. Livescu, Compressibility effects on the Rayleigh-Taylor instability growth between immiscible fluids, Phys. Fluids 16 (1) (2004) 118–127.
[15] W.F. Noh, Errors for calculations of strong shocks using an artificial viscosity and an artificial heat flux, J. Comput. Phys. 72 (1) (1987) 78–120.
[16] J.J. Quirk, A contribution to the great Riemann solver debate, Int. J. Numer. Methods Fluids 18 (1994) 555–574.
[17] Lord Rayleigh, Investigation of the character of the equilibrium of an incompressible heavy fluid of variable density, Proc. Lond. Math. Soc. s1–14 (1) (1882) 170–177.
[18] R. Ramani, J. Reisner, S. Shkoller, A space-time smooth artificial viscosity method with wavelet noise indicator and shock collision scheme, Part 1: the 1-D case, J. Comput. Phys. (2019), https://doi.org/10.1016/j.jcp.2019.02.049.
[19] R. Ramani, J. Reisner, S. Shkoller, A space-time smooth artificial viscosity method for shock-shock and shock-contact collision, in preparation.
[20] J. Reisner, J. Serencsa, S. Shkoller, A space-time smooth artificial viscosity method for nonlinear conservation laws, J. Comput. Phys. 235 (2013) 912–933.
[21] W.J. Rider, Revisiting wall heating, J. Comput. Phys. 162 (2) (2000) 395–410.
[22] L.I. Sedov, The movement of air in a strong explosion, Dokl. Akad. Nauk SSSR 52 (1946) 17–20.
[23] L.I. Sedov, Similarity and dimensional methods in mechanics, in: Translation by Morris Friedman (translation edited by Maurice Holt), Academic Press, New York-London, 1959.
[24] D.H. Sharp, An overview of Rayleigh-Taylor instability, Phys. D: Nonlinear Phenom. 12 (1) (1984) 3–18.
[25] Jing Shi, Yong-Tao Zhang, Chi-Wang Shu, Resolution of high order weno schemes for complicated flow structures, J. Comput. Phys. 186 (2) (2003) 690–696.

[26] C.-W. Shu, High-order finite difference and finite volume weno schemes and discontinuous galerkin methods for cfd, Int. J. Comput. Fluid Dyn. 17 (2) (2003) 107–118.
[27] T.C. Sideris, Formation of singularities in three-dimensional compressible fluids, Commun. Math. Phys. 101 (4) (1985) 475–485.
[28] G.A. Sod, A survey of several finite difference methods for systems of nonlinear hyperbolic conservation laws, J. Comput. Phys. 27 (1978) 1–31.
[29] J.M. Stone, The Athena Code Test Page, https://www.astro.princeton.edu/~jstone/Athena/tests/. (Accessed 17 December 2018).
[30] G.I. Taylor, The instability of liquid surfaces when accelerated in a direction perpendicular to their planes. i, Proc. R. Soc. Lond., Ser. A, Math. Phys. Eng. Sci. 201 (1065) (1950) 192–196.
[31] G.I. Taylor, The formation of a blast wave by a very intense explosion i. theoretical discussion, Proc. R. Soc. Lond. Ser. A, Math. Phys. Sci. 201 (1065) (1950) 159–174.
[32] K.W. Thompson, Time-dependent boundary conditions for hyperbolic systems, J. Comput. Phys. 68 (1) (1987) 1–24.
[33] F. Timmes, Sedov blast wave verification problem, http://cococubed.asu.edu/research_pages/sedov.shtml. (Accessed 17 December 2018).
[34] E.F. Toro, Riemann Solvers and Numerical Methods for Fluid Dynamics, Springer-Verlag, Berlin, Heidelberg, 2009.
[35] J.T. Waddell, C.E. Niederhaus, J.W. Jacobs, Experimental study of Rayleigh-Taylor instability: Low Atwood number liquid systems with single-mode initial perturbations, Phys. Fluids 13 (5) (2001) 1263–1273.
[36] J. Von Neumann, R.D. Richtmyer, A method for the numerical calculation of hydrodynamic shocks, J. Appl. Phys. 21 (1950) 232–237.

CHAPTER 2

The z-model and multiscale algorithms

This chapter was published in Journal of Computational Physics, Vol 405, R. Ramani & S. Shkoller, A multiscale model for Rayleigh-Taylor and Richtmyer-Meshkov instabilities, 109177,

Journal of Computational Physics 405 (2020) 109177

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

A multiscale model for Rayleigh-Taylor and Richtmyer-Meshkov instabilities

Raaghav Ramani, Steve Shkoller *

Department of Mathematics, University of California, Davis, CA 95616 USA

ARTICLE INFO

Article history:
Received 7 May 2019
Received in revised form 29 October 2019
Accepted 2 December 2019
Available online 10 December 2019

Keywords:
Fluid instability
Vortex sheets
Multiscale methods
Rayleigh-Taylor
Richtmyer-Meshkov

ABSTRACT

We develop a novel multiscale model of interface motion for the Rayleigh-Taylor instability (RTI) and Richtmyer-Meshkov instability (RMI) for two-dimensional, inviscid, compressible flows with vorticity, which yields a fast-running numerical algorithm that produces both qualitatively and quantitatively similar results to a resolved gas dynamics code, while running approximately two orders of magnitude (in time) faster. Our multiscale model is founded upon a new compressible-incompressible decomposition of the velocity field $u = v + w$. The incompressible component w of the velocity is also irrotational and is solved using a new asymptotic model of the Birkhoff-Rott singular integral formulation of the incompressible Euler equations, which reduces the problem to one spatial dimension. This asymptotic model, called the higher-order z-model, is derived using small nonlocality in the asymptotic parameter, allows for interface turn-over and roll-up, and yields a significant simplification for the equation describing the evolution of the amplitude of vorticity. This incompressible component w of the velocity controls the small scale structures of the interface and can be solved efficiently on fine grids. Meanwhile, the compressible component of the velocity v remains continuous near contact discontinuities and can be computed on relatively coarse grids, while receiving subgrid scale information from w. We first validate the incompressible higher-order z-model by comparison with classical RTI experiments as well as full point vortex simulations. We then consider both the RTI and the RMI problems for our multiscale model of compressible flow with vorticity, and show excellent agreement with our high-resolution gas dynamics solutions.

Contents

* Corresponding author.
E-mail addresses: rramani@math.ucdavis.edu (R. Ramani), shkoller@math.ucdavis.edu (S. Shkoller).

https://doi.org/10.1016/j.jcp.2019.109177
0021-9991/

1. Introduction

The instability that occurs when an interface separating two fluids of different densities is perturbed and subjected to an acceleration force is a fundamental problem in fluid mechanics. The Rayleigh-Taylor instability (RTI) [67,96] occurs when the lighter fluid is accelerated towards the heavier fluid (under the action of gravity, for instance). The Richtmyer-Meshkov instability (RMI) [81,71] is initiated by the passage of a shock wave across the perturbed interface separating the two fluids. In either case, perturbations of the interface initially grow according to the linear theory, before the system enters the nonlinear regime, in which the light fluid *bubbles* into the heavy fluid, while the heavy fluid *spikes* into the light fluid. The velocity of the resulting flow is discontinuous at the material interface (or contact discontintuity), which initiates the Kelvin-Helmholtz instability (KHI) [89,50]. This causes the interface to *roll up* into complex vortical structures, and eventually leads to turbulent mixing. Each of these instabilities arises in numerous important applications, including in astrophysics [51], inertial confinement fusion [13], and ocean mixing [90]. We refer the reader to the works [85,58,15] and the references therein for further details.

The fundamental mathematical model for the RT and RM instabilities is the Euler system of hydrodynamics equations, consisting of the conservation of mass, momentum, and energy. The mathematical analysis of the Euler equations is extremely challenging due to the ill-posed nature of the equations in the absence of stabilizing mechanisms such as surface tension or viscosity, with the RTI and RMI causing growth of perturbations at the smallest scales available. The highly unstable nature of both the RTI and RMI also poses significant difficulties for numerical methods, and the development of algorithms to study these instabilities has been the subject of intensive research over the last several decades [32,9,40,98,41,36,1,59], and continues to remain a challenge.

As the linear theory shows, the highest frequency perturbations of the interface have the largest growth rates; numerical solutions thus often suffer from the development of spurious small scale structure [62], which does not appear to agree with laboratory experiments [104]. Numerical methods with a large amount of implicit diffusion suppress these small scale eddies, but, in doing so, prevent the development of the KHI mixing zones.

Moreover, even when numerical schemes can be manipulated into producing better solutions [78], simulations can be prohibitively (computationally) expensive. Direct Numerical Simulations (DNS) solve the complete governing equations with exact physical parameters and sufficient resolution to represent all the scales of the flow [73], but the requirement that all the spatial and temporal scales be numerically resolved results in overwhelmingly expensive calculations, both in terms of computational runtime, as well as other basic computational resources, such as memory. As such, simulations are currently generally limited to small Reynolds number flows and simple geometries [102]. These observations indicate a great need for fast algorithms that can be used to accurately predict the RTI and RMI mixing layers and associated growth rates.

In this work, we develop a *multiscale* model for interface evolution during RTI and RMI for two-dimensional, inviscid, compressible flow with vorticity. Our multiscale model is founded upon a new compressible-incompressible decomposition of the velocity field $u = v + w$, which is, in turn, based upon a two-phase elliptic system of Hodge type [19]. The incompressible component w of the velocity is also irrotational and is solved using a new asymptotic model of the Birkhoff-Rott singular-integral formulation of the incompressible Euler equations, which already reduces the problem to one spatial dimension. This asymptotic model, called the *higher-order z-model*, is derived using small nonlocality in the asymptotic parameter, allows for interface turn-over and roll up, and yields a significant simplification for the equation describing the

evolution of the amplitude of vorticity. This incompressible component w of the velocity controls the small scale structures of the interface and can be solved efficiently on fine grids. Meanwhile, the compressible component of the velocity v field remains smooth near contact discontinuities and can be computed on relatively coarse grids, while receiving subgrid scale information from w.

Specifically, our higher-order z-model, approximates the Birkhoff-Rott (BR) equations [26] of interface evolution in two-dimensional multiphase incompressible and irrotational flow, which are, in turn, a reduction of the incompressible and irrotational Euler equation to one-dimensional evolution for the parameterization of the interface $z(\alpha,t) = (z_1(\alpha,t), z_2(\alpha,t))$ and the amplitude of vorticity $\varpi(\alpha,t)$. The original (low-order) z-model was derived by Granero-Belinchón and Shkoller [44] using an asymptotic expansion in a small *non-locality* parameter, and its main advantage over the full BR evolution is a drastic simplification of the dynamics for the amplitude of vorticity ϖ. Without this simplification, the BR dynamics of ϖ is nonlinear, non-local, and is in fact a Fredholm integro-differential equation of the second kind. Numerical methods for this type of equation are thus often quite complex and computationally expensive. On the other hand, the ϖ dynamics given by the z-model allow us to implement an extremely simple numerical method which avoids costly upwinding and iterative procedures [7,92]; in particular, we use a simple Fourier collocation method to evolve ϖ. For the evolution of the interface z, the original (low-order) z-model of [44] used a local equation, while our new (high-order) z-model instead uses Krasny's δ-desingularization [56] of the singular integral kernel. The solution of z and ϖ then provide us with the incompressible velocity field w via an efficient kernel computation. The compressible velocity field v is solved on a very coarse grid (while receiving small-scale information from w) using a very simple WENO scheme together with a nonlinear, spacetime smooth, artificial viscosity method termed the C-method [80,77,78].

We first validate our incompressible and irrotational (high-order) z-model by performing a number of numerical experiments, including both the single-mode and multi-mode RTI, to demonstrate the accuracy of the z-model and its numerical implementation. The computed z-model solutions are compared with observations from laboratory experiments [104,79,107], and are shown to achieve very similar growth rates of the bubbles and spikes, as well as the mixing layer. We additionally compare our z-model solutions with "reference" solutions computed using a sophisticated numerical method for the complete Birkhoff-Rott equations [92], and show that the two solutions are in excellent agreement, thereby demonstrating the validity of the z-model. Moreover, our simplified model equations allow for a numerical computation that is a factor of *at least* 75 times faster than the reference solution calculation. We also compare the simple Krasny desingularization used in the numerical implementation of the z-model with two other higher-order regularizations that smooth the singular integral kernel via convolution with Gaussian-type functions which satisfy certain moment conditions. We demonstrate that all three numerical methods for smoothing the singular integral produce similar numerical solutions (in a sense to be made precise below); as will be shown, these solutions are in reasonable agreement in the asymptotic limit as the mesh spacing $\triangle\alpha$ and viscosity parameter δ converge to zero.

Then, we use our multiscale model, designed to simulate interface evolution in compressible flows with vorticity. As we noted above, we decompose $u = v + w$, where w is both divergence-free and curl-free, but has a discontinuity in its tangential component across the contact discontinuity, while v is continuous across the contact, but is forced by the bulk compression and vorticity of the fluid. By analogy with turbulence models, such as large-eddy simulation (LES) [70], Reynolds-averaged Navier-Stokes (RANS), and Lagrangian-averaged Navier-Stokes (LANS-α) [72], our multiscale model involves the decomposition of the flow into a part which can be solved on a coarse grid (the mean flow), and a part which must be solved on a fine grid (the sub-grid scale fluctuations). The novelty of our approach is that, for the RTI and RMI, the fine grid coincides with the interface itself, and is thus one-dimensional. This means that fine structures can be simulated with much less computational expense than is required for fully two-dimensional calculations on similarly fine meshes.

We describe a simple Eulerian-Lagrangian algorithm for our multiscale model that couples the equations on a coarse two-dimensional mesh with the equations on the high resolution one-dimensional interface. For modeling the RMI, a modified set of equations is used, in which we account for both the effects of shock-contact interaction, as well as the classical Taylor "frozen turbulence" hypothesis [95]. We then discuss the numerical implementation of the algorithm, which uses our incompressible z-model, as well as simple interpolation and integral-kernel calculation techniques. A number of numerical experiments for the RTI and RMI are performed to demonstrate the efficacy of our multiscale model and algorithm for compressible flows with vorticity. In particular, we show that our algorithm produces solutions that agree both qualitatively and quantitatively with (relatively) high-resolution reference solutions. We again perform some basic convergence studies, and find good agreement between the multiscale solutions and high-resolution reference solutions in the limit as the interfacial mesh spacing $\triangle\alpha$ and desingularization parameter δ converge to zero. Moreover, the run times of our multiscale algorithm are two orders of magnitude (or more) faster than those of the corresponding high-resolution reference solutions.

Outline of the paper. Section 2 is devoted to the notation and definitions that will be used throughout the paper. In Section 3, we introduce the full system of Euler equations for compressible flow, followed by the incompressible and irrotational simplification. For the latter, we explain how those equations can be solved using the Birkhoff-Rott *singular integral-kernel* equations for the interface parameterization and amplitude of vorticity. We then describe our asymptotic (in nonlocality) z-model. We next consider the full compressible Euler equations as a two-phase elliptic system for the velocity, and derive a novel compressible-incompressible decomposition of the velocity. This decomposition is the foundation of our multiscale model and algorithm.

In Section 4, we consider the numerical implementation of the incompressible z-model. A simple numerical method is introduced, and results for several numerical experiments are shown, including comparisons with laboratory experiments, theoretical predictions and models, and benchmark numerical simulations. We then present, in Section 5, our multiscale model and algorithms for the compressible RTI and RMI, and give details about their numerical implementations.

Our multiscale algorithm is then applied to two RTI and two RMI test problems Section 6, and compared against both high-resolution simulations and low-resolution simulations. Finally, our conclusions are in Section 7. Two short sections of the Appendix are provided: the first concerns mesh refinement studies for the multiscale algorithm and the second summarizes our numerical method for gas dynamics.

2. Preliminaries

2.1. Some notation and definitions

2.1.1. Derivatives

We write

$$\partial_i f = \frac{\partial f}{\partial x_i} \text{ for } i = 1, 2\,, \quad \partial_t f = \frac{\partial f}{\partial t}\,, \quad \nabla = (\partial_1, \partial_2)\,, \quad \nabla^\perp = (-\partial_2, \partial_1)\,,$$

and for a vector F,

$$\operatorname{div} F = \nabla \cdot F \text{ and } \operatorname{curl} F = \nabla^\perp \cdot F\,.$$

The Laplace operator is defined as $\Delta = \partial_1^2 + \partial_2^2$. Given a transport velocity $u(x,t)$, we shall denote the material derivative $\partial_t + u \cdot \nabla$ by $\frac{\mathrm{D}}{\mathrm{D}t}$.

2.1.2. Fourier series

Let $\mathbb{T}_L$ denote the interval $[-L/2, L/2]$. If $f : \mathbb{T}_L \to \mathbb{R}$ is a square-integrable L-periodic function, then it has the Fourier series representation $f(\alpha) = \sum_{k=-\infty}^{\infty} \widehat{f}(k) e^{\frac{2\pi i k\alpha}{L}}$ for all $\alpha \in \mathbb{T}_L$, where the complex Fourier coefficients are defined by $\mathcal{F}\{f\}(k) \equiv \widehat{f}(k) = \frac{1}{L}\int_{\mathbb{T}_L} f(\alpha) e^{-\frac{2\pi i k\alpha}{L}}\,\mathrm{d}\alpha$. We have the following standard identity:

$$\mathcal{F}\{\partial_\alpha^n f\}(k) = \left(\frac{2i\pi}{L}k\right)^n \widehat{f}(k)\,, \tag{1}$$

where $\partial_\alpha = \frac{\partial}{\partial\alpha}$. We shall sometimes write $\widehat{f}_k$ for $\widehat{f}(k)$.

2.1.3. Principal value integral

The *principal value integral* of a function $f : \mathbb{R} \to \mathbb{R}$ is defined as

$$\mathrm{P}\int_{\mathbb{R}} f(\beta)\,\mathrm{d}\beta := \lim_{\varepsilon \to 0^+} \int_{(-1/\varepsilon, -\varepsilon)\cup(\varepsilon, 1/\varepsilon)} f(\beta)\,\mathrm{d}\beta\,. \tag{2}$$

2.1.4. Hilbert transform

The *Hilbert transform* of a function $f : \mathbb{R} \to \mathbb{R}$ is defined as

$$\mathcal{H} f(\alpha) = \frac{1}{\pi}\,\mathrm{P}\int_{\mathbb{R}} \frac{f(\beta)}{\alpha - \beta}\,\mathrm{d}\beta\,. \tag{3}$$

If f is an L-periodic function on $\mathbb{T}_L$, then

$$\mathcal{H} f(\alpha) = \frac{1}{L}\,\mathrm{P}\int_{\mathbb{T}_L} \frac{f(\beta)}{\tan(\frac{\pi}{L}(\alpha - \beta))}\,\mathrm{d}\beta\,. \tag{4}$$

Equivalently, using the Fourier representation, the Hilbert transform $\mathcal{H}$ can be defined as

$$\widehat{\mathcal{H} f}(k) = -i\,\mathrm{sgn}(k)\,\widehat{f}(k)\,. \tag{5}$$

In particular, we note that $\mathcal{H}^2 = -1$.

2.1.5. Discrete operators in Fourier space

Let $\alpha \in \mathbb{T}_L$. We discretize the parameter α with $N+1 = 2^r + 1$ nodes,

$$\alpha_k = -L/2 + (k-1)\triangle\alpha\,,$$

with $\triangle\alpha = L/N$. Given an L-periodic function $f(\alpha)$, we denote by $f_k = f(\alpha_k)$ the function f evaluated at a point $\alpha_k \in \mathbb{T}_L$ Let $\tilde{\mathcal{F}}$ and $\tilde{\mathcal{F}}^{-1}$ denote the discrete Fourier and inverse Fourier transforms, respectively, defined for sequences (f_k) of length $N = 2^r$ by

$$\tilde{\mathcal{F}}\{f_k\}_m = \sum_{l=1}^{N} f_l \cdot e^{-\frac{2i\pi}{N}(m-1)(l-1)} \quad \text{and} \quad \tilde{\mathcal{F}}^{-1}\left\{\widehat{f}_m\right\}_k = \frac{1}{N}\sum_{l=1}^{N} \widehat{f}_l \cdot e^{\frac{2i\pi}{N}(k-1)(l-1)}\,.$$

We define the discrete Fourier operators (H_k), (D_k), $(D_k^2) \in \mathbb{C}^N$ as

$$H_k = \begin{cases} 0 & \text{if } k = 1\,, \\ -i & \text{if } 2 \le k \le (N+1)/2\,, \\ i & \text{if } k > (N+1)/2\,, \end{cases} \tag{6}$$

$$D_k = \begin{cases} \frac{2\pi i}{L}(k-1) & \text{if } k < (N+1)/2\,, \\ 0 & \text{if } k = (N+1)/2\,, \\ -\frac{2\pi i}{L}(N-k) & \text{if } k > (N+1)/2\,, \end{cases} \tag{7}$$

$$D_k^2 = \begin{cases} -\left(\frac{2\pi}{L}\right)^2 (k-1)^2 & \text{if } k \le (N+1)/2\,, \\ -\left(\frac{2\pi}{L}\right)^2 (N-k)^2 & \text{if } k > (N+1)/2\,. \end{cases} \tag{8}$$

Formula (6) is the discrete Hilbert transform in Fourier space, while (7) and (8) are the discrete derivative operators ∂_α and ∂_α^2, respectively, in Fourier space.

2.2. Computational platform and code optimization

All of the numerical simulations conducted in this work were run on a Macbook Pro laptop using a 2.4 GHz Intel Core i5 processor with 8 GB of RAM. The operating system is macOS High Sierra 10.13.6, and the GFortran F90 compiler is used.

The codes for the numerical methods described in the paper are implemented in the same programming framework, but are not otherwise specially optimized (apart from a specific calculation described in the paper). The same input, output, and timing routines are used in all of the codes. This consistency allows for a reliable comparison of the different algorithms and their associated imposed computational burden.

3. The Euler equations

3.1. The compressible Euler equations

The fundamental mathematical model for the motion of an inviscid two-dimensional fluid is given by the compressible Euler equations:

$$\partial_t \rho + \operatorname{div}(\rho u) = 0\,, \tag{9a}$$

$$\partial_t(\rho u) + \operatorname{div}(\rho u \otimes u) + \nabla p + \rho g e_2 = 0\,, \tag{9b}$$

$$\partial_t E + \operatorname{div}(u(E+p)) + \rho g u_2 = 0\,, \tag{9c}$$

where $\otimes$ denotes the tensor product, and $\operatorname{div} M$ denotes the row-wise divergence of a matrix M. The velocity vector is $u = (u_1\,, u_2)$ with horizontal component u_1 and vertical component u_2, $\rho > 0$ is the fluid density (assumed strictly positive), E denotes the energy, and p is the pressure defined by an equation of state. These equations are, in fact, the basic conservation laws of fluid dynamics: (9a) is conservation of mass, (9b) is conservation of linear momentum, and (9c) is conservation of energy.

The system (9) can be written in classical conservation-law form as the Cauchy problem

$$\partial_t \mathbf{U}(x,t) + \partial_{x_1}\mathbf{F}(\mathbf{U}(x,t)) + \partial_{x_2}\mathbf{G}(\mathbf{U}(x,t)) = \mathbf{H}(x,t), \quad x \in \mathbb{R}^2\,, t > 0, \tag{10a}$$

$$\mathbf{U}(x,0) = \mathbf{U}_0(x), \quad x \in \mathbb{R}^2\,, t = 0, \tag{10b}$$

where the 4-vector $\mathbf{U}$ and the *flux functions* $\mathbf{F}(\mathbf{U})$ and $\mathbf{G}(\mathbf{U})$ are defined as

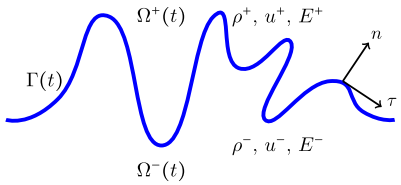


Fig. 1. An example of a time-dependent contact discontinuity $\Gamma(t)$ separating the two fluid regions $\Omega^+(t)$ and $\Omega^-(t)$.

$$\mathbf{U} = \begin{pmatrix} \rho \\ \rho u_1 \\ \rho u_2 \\ E \end{pmatrix} \quad \text{and} \quad \mathbf{F}(\mathbf{U}) = \begin{pmatrix} \rho u_1 \\ \rho u_1^2 + p \\ \rho u_1 u_2 \\ u_1(E+p) \end{pmatrix} \quad \text{and} \quad \mathbf{G}(\mathbf{U}) = \begin{pmatrix} \rho u_2 \\ \rho u_1 u_2 \\ \rho u_2^2 + p \\ u_2(E+p) \end{pmatrix}. \tag{11}$$

The space coordinate is $x = (x_1, x_2)$, with x_1 denoting the horizontal component, x_2 denoting the vertical component, and $t \geq 0$ denoting time. The function $\mathbf{H}$ denotes the forcing function due to gravity, and so will be given as $\mathbf{H} = (0\,, 0\,, -\rho g\,, -\rho g u_2)$, where g is a gravitational acceleration constant. The pressure p is defined by the ideal gas law,

$$p = (\gamma - 1)\left(E - \frac{1}{2}\rho|u|^2\right), \tag{12}$$

where γ is the adiabatic constant, which we will assume takes the value $\gamma = 1.4$, unless otherwise stated. We also define the specific internal energy per unit mass of the fluid as $e = p/(\rho(\gamma - 1))$. Once the initial data $u_0(x)$, $\rho_0(x)$, $E_0(x)$ are specified, solutions of (10) provide the velocity, density, and energy for each instant of time for which the solution exits.

3.2. The incompressible and irrotational Euler equations

In the absence of sound waves, the system (9) can be simplified to model incompressible flows. The incompressible Euler equations are written as

$$\rho\left[\partial_t w(x,t) + (w \cdot \nabla) w\right] + \nabla p + g\rho e_2 = 0, \quad x \in \mathbb{R}^2\,, t > 0, \tag{13a}$$

$$\operatorname{div} w = 0, \quad x \in \mathbb{R}^2\,, t \geq 0, \tag{13b}$$

$$w(x,0) = w_0(x), \quad x \in \mathbb{R}^2\,, t = 0\,, \tag{13c}$$

where $w = (w_1, w_2)$ denotes a divergence-free velocity vector field, the density ρ is assumed to be a constant (or piecewise constant as we shall consider below), and the pressure p is a Lagrange multiplier which enforces the incompressibility constraint (13b). We define the two-dimensional vorticity function $\omega = \operatorname{curl} w$, where

$$\operatorname{curl} w := \nabla^\perp \cdot w = \partial_1 w_2 - \partial_2 w_1\,.$$

Computing the curl of (13a) and using (13b) shows that the two-dimensional vorticity is transported by incompressible flows,

$$\partial_t \omega + w \cdot \nabla \omega = 0\,,$$

and hence if the initial velocity $w_0(x)$ is chosen to be irrotational such that $\omega_0(x) = 0$, then $\omega(x,t) = 0$ for all time t for which the solution exists. Thus, for such data, we supplement (13) with

$$\operatorname{curl} w = 0, \quad x \in \mathbb{R}^2\,, t \geq 0\,. \tag{13d}$$

3.3. The compressible Euler equations as a two-phase hyperbolic system

We are particularly interested in two-dimensional discontinuous solutions of the Euler equations (10) which propagate curves of discontinuity, whose evolution is determined by the Rankine-Hugoniot conditions (see, for example, [31]). Specifically, our focus is on two-dimensional solutions $\mathbf{U} = (\rho, \rho u_1, \rho u_2, E)$ to (10) which have jump discontinuities across a time-dependent, space-periodic material interface $\Gamma(t)$ (see Fig. 1).

The two-dimensional fluid domain is written as

$$\mathbb{R}^2 = \Omega^+(t) \cup \Omega^-(t) \cup \Gamma(t)\,,$$

where $\Omega^+(t)$ denotes the time-dependent open domain lying above $\Gamma(t)$, while $\Omega^-(t)$ denotes the open domain lying below $\Gamma(t)$. We let $n(\cdot, t)$ denote the unit normal vector to $\Gamma(t)$ pointing into $\Omega^+(t)$ and let $\tau(\cdot, t)$ denote the unit tangent vector

to $\Gamma(t)$, so that the pair (τ, n) denote a right-handed basis. We denote by $\mathbf{U}^+$ the solution in the domain $\Omega^+(t)$ and by $\mathbf{U}^-$, the solution in $\Omega^-(t)$. The jump of a function $\mathbf{U}$ across $\Gamma(t)$ is denoted by

$$[\![\mathbf{U}]\!] = \mathbf{U}^+ - \mathbf{U}^- \text{ on } \Gamma(t)\,.$$

The Rankine-Hugoniot conditions relate the speed of propagation $\sigma(t)$ of the curve of discontinuity $\Gamma(t)$ with the jump discontinuity in the variables $\mathbf{U} = (\rho, \rho u_1, \rho u_2, E)$ via the relation

$$\sigma\,[\![\rho]\!] = [\![\rho u \cdot n]\!]\,, \tag{14a}$$

$$\sigma\,[\![\rho u]\!] = [\![(\rho u \cdot n)u + pn]\!]\,, \tag{14b}$$

$$\sigma\,[\![E]\!] = [\![(E+p)u \cdot n]\!]\,, \tag{14c}$$

which represent, respectively, the conservation of mass, linear momentum, and energy across the discontinuity. Notice that (14b) admits solutions with $[\![p]\!] = 0$ and $[\![(\sigma - u\cdot n)\rho u]\!] = 0$, and that the latter condition is satisfied if $u^\pm \cdot n = \sigma$. Such discontinuities are known as *contact discontinuities*, in which case the interface $\Gamma(t)$ is transported by the fluid velocity u, and the pressure p is continuous across $\Gamma(t)$. Contact discontinuities are the class of two-dimensional discontinuous solutions solving the following coupled *two-phase* system of hyperbolic equations:

$$\partial_t \rho^\pm + \operatorname{div}(\rho u^\pm) = 0, \qquad \text{in } \Omega^\pm(t)\,, \tag{15a}$$

$$\partial_t(\rho^\pm u^\pm) + \operatorname{div}(\rho^\pm u^\pm \otimes u^\pm) + \nabla p^\pm + \rho g e_2 = 0, \qquad \text{in } \Omega^\pm(t)\,, \tag{15b}$$

$$\partial_t E^\pm + \operatorname{div}(u^\pm(E^\pm + p^\pm)) + \rho g u_2^\pm = 0, \qquad \text{in } \Omega^\pm(t)\,, \tag{15c}$$

$$[\![u \cdot n]\!] = 0, \qquad \text{on } \Gamma(t)\,, \tag{15d}$$

$$[\![u \cdot \tau]\!] \neq 0, \qquad \text{on } \Gamma(t)\,, \tag{15e}$$

$$[\![p]\!] = 0, \qquad \text{on } \Gamma(t)\,, \tag{15f}$$

$$\big(u^\pm(x,0), \rho^\pm(x,0), E^\pm(x,0), \Gamma(0)\big) = \big(u_0^\pm, \rho_0^\pm, E_0^\pm, \Gamma_0\big)\,, \qquad \text{at } t = 0\,. \tag{15g}$$

As already noted the interface $\Gamma(t)$ is transported by the velocity u and we will make the dynamics of $\Gamma(t)$ precise, once we introduce a parameterization for $\Gamma(t)$. The tangential velocity jump discontinuity is the primary mechanism that initiates the Kelvin-Helmholtz instability . The densities $\rho^\pm$ and the energies $E^\pm$ are, in general, also discontinuous across $\Gamma(t)$. The initial data is specified in (15g).

3.4. The two-phase incompressible and irrotational Euler equations

The incompressible and irrotational Euler equations for two-phase flow are written as

$$\rho^\pm\left(\partial_t w^\pm + w^\pm \cdot \nabla w^\pm\right) + \nabla p^\pm + \rho^\pm g e_2 = 0 \qquad \text{in } \Omega^\pm(t)\,, \tag{16a}$$

$$\operatorname{curl} w = \operatorname{div} w = 0 \qquad \text{in } \Omega^\pm(t)\,, \tag{16b}$$

$$[\![w \cdot n]\!] = 0 \qquad \text{on } \Gamma(t)\,, \tag{16c}$$

$$[\![w \cdot \tau]\!] \neq 0 \qquad \text{on } \Gamma(t)\,, \tag{16d}$$

$$[\![p]\!] = 0 \qquad \text{on } \Gamma(t)\,, \tag{16e}$$

$$\big(w^\pm(x,0), \Gamma(0)\big) = \big(w_0^\pm, \Gamma_0\big) \qquad \text{at } t = 0\,, \tag{16f}$$

and ρ^+ and ρ^- are constant in each phase. Again, the interface $\Gamma(t)$ is transported by the velocity w, and for incompressible and irrotational flows, the interface $\Gamma(t)$ is called a *vortex sheet*, because the vorticity is restricted to the one-dimensional interface as a measure, as will be made precise.

Incompressibility and irrotationality of the flow allow for a reduction of the system (16) to a coupled system of evolution equations in one space dimension. We let $\mathbb{T}_L = [-L/2\,, L/2]$ denote a (periodic) interval of length L, and introduce a *parameterization* of the interface $\Gamma(t)$ by a mapping $z : \mathbb{T}_L \to \mathbb{R}^2$, so that for each α in $\mathbb{T}_L$, the vector $z(\alpha, t) = (z_1(\alpha, t), z_2(\alpha, t))$ represents a point on the interface $\Gamma(t)$. Moreover, for any α_0, the vector $\partial_\alpha z(\alpha_0, t)$ is tangent to $\Gamma(t)$ at the point $z(\alpha_0, t)$, and $\tau = \partial_\alpha z/|\partial_\alpha z|$ is the unit tangent vector at that point (as shown in Fig. 2).

Now, since $\Gamma(t)$ moves with speed $w \cdot n$, it follows that $\partial_t z(\alpha, t) = [w(z(\alpha, t), t) \cdot n]n$ and that the tangential motion of the interface $\partial_t z \cdot \partial_\alpha z$ has no constraints at all. The dynamics of the interface $\Gamma(t)$ are governed by the evolution equation

$$\partial_t z(\alpha, t) = w(z(\alpha, t), t)\,. \tag{17}$$

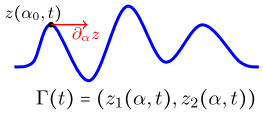


Fig. 2. The curve $\Gamma(t)$ is parameterized by $z(\alpha, t) = (z_1(\alpha, t), z_2(\alpha, t))$. A tangent vector at a point $z(\alpha, t)$ on $\Gamma(t)$ is given by $\partial_\alpha z(\alpha, t)$.

The vorticity ω vanishes in each of $\Omega^{\pm}(t)$, and is in fact a *measure* supported on $\Gamma(t)$, written as

$$\omega = \varpi\, \delta_{\Gamma(t)}\,,$$

where $\delta_{\Gamma(t)}$ is the Dirac delta distribution supported on $\Gamma(t)$, and the function ϖ is the *amplitude of vorticity* along $\Gamma(t)$. More precisely, if φ is any smooth test function with compact support in $\mathbb{R}^2$, then

$$\langle \omega\,, \varphi\rangle = \int_{\mathbb{R}} \varpi\,(\beta, t)\varphi(z(\beta, t))\,\mathrm{d}\beta\,.$$

The amplitude of vorticity ϖ may be computed in terms of the jump in the velocity as

$$\begin{aligned}\int_{\mathbb{R}} \varpi\,(\beta, t)\varphi(z(\beta, t))\,\mathrm{d}\beta = \langle \omega\,, \varphi\rangle &:= -\int_{\Omega} w\cdot\nabla^{\perp}\varphi\,\mathrm{d}x \\ &= \int_{\Omega^+\cup\Omega^-} \varphi\nabla^{\perp}\cdot w\,\mathrm{d}x - \int_{\Gamma} [\![w\cdot\tau]\!]\varphi\,\mathrm{d}S\,.\end{aligned}$$

Since the vorticity $\omega = \nabla^{\perp}\cdot w = 0$ in $\Omega^+(t)\cup\Omega^-(t)$, it then follows that

$$\int_{\mathbb{T}_L} \varpi\,(\beta, t)\varphi(z(\beta, t))\,\mathrm{d}\beta = -\int_{\Gamma} [\![w\cdot\tau]\!]\varphi\,\mathrm{d}S = -\int_{\mathbb{R}} [\![w\cdot\tau]\!]\varphi(z(\beta, t))|\partial_\alpha z(\beta, t)|\,\mathrm{d}\beta$$

for any smooth test function φ with compact support in $\mathbb{R}^2$, which implies that

$$\varpi = -[\![w\cdot\tau]\!]|\partial_\alpha z| = -[\![w\cdot\partial_\alpha z]\!]\,.$$

Due to the fact that the flow is both irrotational and incompressible, there exist scalar *stream functions* $\psi^{\pm}(x, t)$ such that $\Delta\psi^{\pm} = 0$ in $\Omega^{\pm}(t)$ and $w^{\pm} = \nabla^{\perp}\psi^{\pm}$.

Following [83,14], we next reduce (16) to a system of coupled evolutionary integro-differential equations in one space dimension. The incompressible and irrotational velocity w can be reconstructed from the vorticity measure ϖ using the well-known Biot-Savart kernel $\mathcal{K}_{\mathbb{R}}(x)$, which is an integral representation for $\nabla^{\perp}\Delta^{-1}$ in $\mathbb{R}^2$. The kernel is defined by

$$\mathcal{K}_{\mathbb{R}}(x) = \frac{x^{\perp}}{2\pi|x|^2} = \frac{1}{2\pi}\left(\frac{-x_2}{x_1^2+x_2^2}\,,\,\frac{x_1}{x_1^2+x_2^2}\right). \tag{18}$$

Away from the interface, the velocity w is then given as

$$w(x, t) = \mathrm{P}\int_{\mathbb{R}} \mathcal{K}_{\mathbb{R}}(x - z(\beta, t))\varpi\,(\beta, t)\,\mathrm{d}\beta\,, \tag{19}$$

for $x\in\Omega^+(t)\cup\Omega^-(t)$, where we recall that the integral is to be understood in the principal value sense (2). At the interface $\Gamma(t)$, the velocity $w|_{\Gamma(t)}$ is defined to be the average $(w^+ + w^-)/2$ on $\Gamma(t)$. The Plemelj formulae give

$$w^{\pm}(z(\alpha, t), t) = \mathrm{P}\int_{\mathbb{R}} \mathcal{K}_{\mathbb{R}}(z(\alpha, t) - z(\beta, t))\varpi\,(\beta, t)\,\mathrm{d}\beta \pm \frac{1}{2}\frac{\varpi\,(\alpha, t)}{|\partial_\alpha z(\alpha, t)|}\tau\,,$$

from which it follows that

$$w(z(\alpha, t), t) = \mathrm{P}\int_{\mathbb{R}} \mathcal{K}_{\mathbb{R}}(z(\alpha, t) - z(\beta, t))\varpi\,(\beta, t)\,\mathrm{d}\beta\,.$$

This integral is over the real line. For horizontally periodic flows, the integral can be summed over the periodic images to yield an integral over a single period, with the kernel given by

$$\mathcal{K}_{\mathbb{T}_L}(x) = \frac{(-\sinh(2\pi x_2/L)\,,\,\sin(2\pi x_1/L))}{2L\,(\cosh(2\pi x_2/L) - \cos(2\pi x_1/L))}\,, \tag{20}$$

where $\mathbb{T}_L = [-L/2\,,\,L/2]$ denotes the periodic interval with period L. Hence,

$$w(z(\alpha,t),t) = \mathrm{P}\int_{\mathbb{T}_L} \mathcal{K}_{\mathbb{T}_L}(z(\alpha,t) - z(\beta,t))\varpi(\beta,t)\,\mathrm{d}\beta\,. \tag{21}$$

Together with (17), the system is closed by determining the evolution equation for the amplitude of vorticity ϖ. A lengthy computation [26,44] using the Bernoulli equation, the Plemelj formulae, and (16e) provides the dynamics for ϖ; together with (17) and (21), we obtain the following coupled system:

$$\partial_t z(\alpha,t) = \mathrm{P}\int_{\mathbb{T}_L} \mathcal{K}_{\mathbb{T}_L}(z(\alpha,t) - z(\beta,t))\varpi(\beta,t)\,\mathrm{d}\beta\,, \tag{22a}$$

$$\begin{aligned}\partial_t \varpi(\alpha,t) = -\partial_\alpha \left[\frac{A}{4\pi^2} \left| \mathrm{P}\int_{\mathbb{R}} \varpi(\beta,t) \frac{(z(\alpha,t) - z(\beta,t))^\perp}{|z(\alpha,t) - z(\beta,t)|^2}\,\mathrm{d}\beta \right|^2 - \frac{A}{4}\frac{\varpi(\alpha,t)^2}{|\partial_\alpha z(\alpha,t)|^2} - 2Agz_2 \right] \\ + \frac{A}{\pi}\partial_t \left[\mathrm{P}\int_{\mathbb{R}} \varpi(\beta,t) \frac{(z(\alpha,t) - z(\beta,t))^\perp}{|z(\alpha,t) - z(\beta,t)|^2} \cdot \partial_\alpha z(\alpha,t)\,\mathrm{d}\beta \right],\end{aligned} \tag{22b}$$

where

$$A = (\rho^+ - \rho^-)/(\rho^+ + \rho^-)$$

is the Atwood number. These equations are solved for $\alpha \in \mathbb{T}_L$ and $t > 0$. The coupled equations (22) are the incompressible and irrotational Euler equations, reduced to a one-dimensional problem for the three unknowns (z_1, z_2, ϖ).

The analysis of the BR system (22) is difficult, due to the presence of the Kelvin-Helmholtz instability. Linear stability analysis yields perturbation solutions with arbitrarily large growth rates, so that the problem is ill-posed in the sense of Hadamard [38,54]. Delort [34] proved existence of global weak solutions for initial data that is a signed vorticity measure (concentrated on the interface); see also [68,39,65]. Uniqueness of these solutions has not been proved, and there is evidence to suggest that such solutions are, in fact, not unique [94,76,69,66].

3.5. An asymptotic model for incompressible interface motion: the z-model

As we will explain in Section 4, the numerical solution of the system (22) can be computationally expensive and difficult to implement. Moreover, the equations are sufficiently complex that, in many cases, the dynamics of solutions is extremely difficult to analyze. As such, there has been a sustained effort to develop *model* equations that can suitably approximate the Euler equations in certain asymptotic regimes. For water waves (i.e. $A = -1$), there are a number of such equations (see, for example, [5,30] and references therein), and for the two-fluid case (i.e. $-1 < A < 1$), a number of modal models have been proposed for the evolution of the interface, such as the models of [42] and [46]; we refer the reader to Zhou [108] for an extensive review of the subject.

The fundamental difficulty is the nonlocal nature of the singular integral equations (22), in which the dynamics at a point on the interface require information at all other points on the interface. By developing a new asymptotic procedure in which z and ϖ are expanded in a small non-locality parameter, Granero-Belinchón and Shkoller [44] obtained model equations, approximating the solution to (22), which allow for interface turn-over and place no constraints on the steepness of the interface. These localized equations are

$$\partial_t z(\alpha,t) = \frac{1}{2}\mathcal{H}\varpi(\alpha,t)\frac{\partial_\alpha z^\perp(\alpha,t)}{|\partial_\alpha z(\alpha,t)|^2}\,, \tag{23a}$$

$$\partial_t \varpi(\alpha,t) = -\partial_\alpha \left[\frac{A}{2|\partial_\alpha z(\alpha,t)|^2}\mathcal{H}(\varpi(\alpha,t)\mathcal{H}\varpi(\alpha,t)) - 2Agz_2(\alpha,t) \right], \tag{23b}$$

where $\mathcal{H}$ denotes the Hilbert transform, defined in (5). The equations (23) are called the (lower-order) z-model.

A number of numerical experiments of the (lower-order) z-model were performed in [44], which demonstrated very good agreement with experimental data and theoretical predictions of interface growth, but the localized nature of the evolution for $z(\alpha,t)$ in (23a) can inhibit the initiation of Kelvin-Helmholtz roll-up. On the other hand, the fundamental challenge in simulating the Euler system (22) stems from the evolution equation for ϖ. As such we introduce the *higher-order* z-model as the following system:

$$\partial_t z(\alpha,t) = \mathrm{P}\int_{\mathbb{T}_L} \mathcal{K}_{\mathbb{T}_L}(z(\alpha,t)-z(\beta,t))\varpi(\beta,t)\,\mathrm{d}\beta\,, \tag{24a}$$

$$\partial_t \varpi(\alpha,t) = -\partial_\alpha\left[\frac{A}{2|\partial_\alpha z(\alpha,t)|^2}\mathcal{H}(\varpi(\alpha,t)\mathcal{H}\varpi(\alpha,t)) - 2Agz_2(\alpha,t)\right], \tag{24b}$$

in which the asymptotic model for ϖ evolution is coupled to the integral equation for z.

3.6. The Euler equations as a two-phase elliptic system for velocity

We now reformulate the full compressible Euler equations (15) as a two-phase elliptic system for the compressible velocity vector u. As we have already stated, by using the parameterization $z(\alpha,t)$ for the interface $\Gamma(t)$, the dynamics of the interface $\Gamma(t)$ are governed by the evolution equation $\partial_t z(\alpha,t) = u(z(\alpha,t),t)$, and from the definition of the amplitude of vorticity ϖ, we have the following jump conditions for the velocity:

$$[\![u\cdot\tau]\!] = -\tfrac{\varpi}{|\partial_\alpha z|} \text{ and } [\![u\cdot n]\!] = 0\,.$$

Next, from equation (15a), we have that

$$F^\pm := \operatorname{div} u^\pm = -\frac{\frac{\mathrm{D}\rho^\pm}{\mathrm{D}t}}{\rho^\pm}\,.$$

Letting the operator curl act on (15b), and setting $\omega = \operatorname{curl} u$, we find that ω is the solution of

$$\tfrac{\mathrm{D}\omega}{\mathrm{D}t} + \omega\operatorname{div} u = -\tfrac{\nabla\rho\cdot\nabla^\perp p}{\rho^2}\,.$$

Thus, given z, ϖ, $F^\pm$, and $\omega^\pm$, we can reconstruct $u^\pm$ by solving the following two-phase elliptic system:

$$\operatorname{div} u^\pm = F^\pm, \qquad \text{in } \Omega^\pm(t)\,, \tag{25a}$$

$$\operatorname{curl} u^\pm = \omega^\pm, \qquad \text{in } \Omega^\pm(t)\,, \tag{25b}$$

$$[\![u\cdot n]\!] = 0, \qquad \text{on } \Gamma(t)\,, \tag{25c}$$

$$[\![u\cdot\tau]\!] = -\tfrac{\varpi}{|\partial_\alpha z|}, \qquad \text{on } \Gamma(t)\,. \tag{25d}$$

For the two-dimensional geometry that we are considering, the system (25) is uniquely solvable [19], and thus u obtained from (25) is the velocity field solving the compressible Euler equations (15).

3.7. A compressible-incompressible decomposition of the Euler equations

We substitute the additive decomposition $u = v + w$ into the two-phase elliptic system (25) and define $v^\pm$ and $w^\pm$ to be the solutions of

$$\operatorname{div} v^\pm = F^\pm \quad \text{in} \quad \Omega^\pm(t)\,, \qquad \operatorname{div} w^\pm = 0 \quad \text{in} \quad \Omega^\pm(t)\,, \tag{26a}$$

$$\operatorname{curl} v^\pm = \omega^\pm \quad \text{in} \quad \Omega^\pm(t)\,, \qquad \operatorname{curl} w^\pm = 0 \quad \text{in} \quad \Omega^\pm(t)\,, \tag{26b}$$

$$[\![v\cdot n]\!] = 0 \quad \text{on} \quad \Gamma(t)\,, \qquad [\![w\cdot n]\!] = 0 \quad \text{on} \quad \Gamma(t)\,, \tag{26c}$$

$$[\![v\cdot\tau]\!] = 0 \quad \text{on} \quad \Gamma(t)\,, \qquad [\![w\cdot\tau]\!] = -\tfrac{\varpi}{|\partial_\alpha z|} \quad \text{on} \quad \Gamma(t)\,, \tag{26d}$$

together with

$$\partial_t z(\alpha,t) = v(z(\alpha,t),t) + w(z(\alpha,t),t)\,, \tag{26e}$$

$$\varpi(\alpha,t) = -[\![w(z(\alpha,t),t)\cdot\partial_\alpha z(\alpha,t)]\!]\,. \tag{26f}$$

The velocity w is incompressible and irrotational, but has a discontinuity in its tangential component, while the velocity v is continuous and is forced by the bulk compression and vorticity of the fluid. There are a number of different ways to find velocities v and w such that u solves the full compressible Euler equations (15). We shall simultaneously solve for the pair (v,w) as the solution of the following system:

$$\bar{\rho}^{\pm}\left(\partial_t w^{\pm} + w^{\pm}\cdot\nabla w^{\pm}\right) + \nabla p^{\pm} + \bar{\rho}^{\pm} g e_2 = 0 \qquad \text{in } \Omega^{\pm}(t)\,, \tag{27a}$$

$$\operatorname{curl} w^{\pm} = \operatorname{div} w^{\pm} = 0 \qquad \text{on } \Omega^{\pm}(t)\,, \tag{27b}$$

$$[\![w\cdot n]\!] = 0 \qquad \text{on } \Gamma(t)\,, \tag{27c}$$

$$[\![w\cdot\tau]\!] = -\tfrac{\varpi}{|\partial_\alpha z|} \qquad \text{on } \Gamma(t)\,, \tag{27d}$$

$$[\![p]\!] = 0 \qquad \text{on } \Gamma(t)\,, \tag{27e}$$

$$\left(w^{\pm}(x,0), z(\alpha,0)\right) = (0, z_0(\alpha)) \qquad \text{at } t=0\,, \tag{27f}$$

where $z_0(\alpha)$ is the initial data for the parameterization of the interface, the initial amplitude of vorticity is computed as

$$\varpi(\alpha,0) = -[\![u(z_0(\alpha),0)\cdot\partial_\alpha z_0(\alpha)]\!]\,,$$

and the density functions $\bar{\rho}^{\pm}$ are constants given by $\bar{\rho}^{\pm} = \rho_0^{\pm}|_{\Gamma_0}$. This is coupled to

$$\partial_t \rho^{\pm} + \operatorname{div}(\rho^{\pm} v^{\pm}) = -\operatorname{div}(\rho^{\pm} w^{\pm}) \qquad \text{in } \Omega^{\pm}(t)\,, \tag{28a}$$

$$\begin{aligned}\partial_t(\rho^{\pm} v^{\pm}) + \operatorname{div}(\rho^{\pm} v^{\pm}\otimes v^{\pm}) + \nabla p^{\pm} + \rho^{\pm} g e_2 &= -\partial_t(\rho^{\pm} w^{\pm}) \\ &\quad - \operatorname{div}\left(\rho^{\pm}(w^{\pm}\otimes w^{\pm} + w^{\pm}\otimes v^{\pm} + v^{\pm}\otimes w^{\pm})\right) \qquad \text{on } \Omega^{\pm}(t)\,,\end{aligned} \tag{28b}$$

$$\partial_t E^{\pm} + \operatorname{div}(v^{\pm}(E^{\pm}+p^{\pm})) + \rho^{\pm} g v_2^{\pm} = -\operatorname{div}(w^{\pm}(E^{\pm}+p^{\pm})) - g\rho^{\pm} w_2^{\pm} \qquad \text{on } \Omega^{\pm}(t)\,, \tag{28c}$$

$$[\![v]\!] = 0 \qquad \text{on } \Gamma(t)\,, \tag{28d}$$

$$[\![p]\!] = 0 \qquad \text{on } \Gamma(t)\,, \tag{28e}$$

$$\left(v^{\pm}(x,0), \rho^{\pm}(x,0), E^{\pm}(x,0), z(\alpha,0)\right) = \left(u_0^{\pm}, \rho_0^{\pm}, E_0^{\pm}, z_0(\alpha)\right) \qquad \text{at } t=0\,, \tag{28f}$$

together with (26e) and (26f). This decomposition of the flow into velocities v and w provides a natural setting for a multiscale model of compressible interface evolution. In particular, we shall develop a two-scale solution strategy, in which (27) is solved over small scales using our higher-order incompressible z-model (24), and (28) is solved over large scales.

An equivalent formulation for this (v, w) system is given by replacing (26e), (26f), and (27) with

$$\partial_t z(\alpha,t) = \mathrm{P}\!\int_{\mathbb{T}_L} \mathcal{K}_{\mathbb{T}_L}(z(\alpha,t) - z(\beta,t))\varpi(\beta,t)\,\mathrm{d}\beta + v(z(\alpha,t),t)\,, \tag{29a}$$

$$\begin{aligned}\partial_t \varpi(\alpha,t) = -\partial_\alpha &\left[\frac{A}{4\pi^2}\left|\mathrm{P}\!\int_{\mathbb{R}} \varpi(\beta,t)\frac{(z(\alpha,t)-z(\beta,t))^{\perp}}{|z(\alpha,t)-z(\beta,t)|^2}\,\mathrm{d}\beta\right|^2 - \frac{A}{4}\frac{\varpi(\alpha,t)^2}{|\partial_\alpha z(\alpha,t)|^2} - 2Agz_2\right] \\ &+ \frac{A}{\pi}\partial_t\left[\mathrm{P}\!\int_{\mathbb{R}} \varpi(\beta,t)\frac{(z(\alpha,t)-z(\beta,t))^{\perp}}{|z(\alpha,t)-z(\beta,t)|^2}\cdot\partial_\alpha z(\alpha,t)\,\mathrm{d}\beta\right],\end{aligned} \tag{29b}$$

and coupling these equations with (28). Then w is computed using (19). This will be the basis for our multiscale modeling approach.

We note that both the two-dimensional compressible and incompressible Euler equations are ill-posed in Sobolev spaces for most vortex sheet initial data.[1] On the other hand, these systems of equations become well-posed in Sobolev spaces if either bulk viscosity [35] or surface tension along the contact discontinuity [18] is added. As such, any state-of-the-art high-order numerical discretization of the compressible Euler equations uses some form of regularization to remove small-scale instability and oscillations (see the review paper [62]). In order to generate high-resolution reference solutions for comparison with our multiscale algorithm, we too rely on a regularization scheme that employs a new type of anisotropic artificial viscosity operator which is described in Section 5.4.1. This anisotropic operator adds nonlinear viscosity only in directions tangential to the evolving front while adding virtually zero viscosity in the direction normal to the interface. This approach ensures that the contact discontinuity does not become too smeared (which indeed occurs for more traditional isotropic artificial viscosity operators).

In deriving the multiscale decomposition, we return to the inviscid Euler setting in which all regularization is removed. The inviscid velocity u is decomposed into v and w, which are evolved by the equations (28) and (29). As such v and w are again governed by inviscid systems, and in particular, w is governed by the inviscid incompressible and irrotational Euler

[1] The two-dimensional compressible Euler equations are weakly well-posed in Sobolev spaces if the initial vorticity measure (tangential jump of velocity along the interface) is sufficiently large relative to the Mach number of the flow [29,28].

equations, which are once again ill-posed, while v no longer has a discontinuity and does not suffer from the same small-scale instabilities as the original Euler system it was derived from. Now, the w system must be numerically regularized, but before we do so, we make a significant simplification for the dynamics for the amplitude of vorticity ϖ. The equation (29b) is a highly nonlinear and nonlocal equation. We replace this complicated evolution with our asymptotic z-model (24b). This is a Burgers-type equation which leads to shock formation from initial data consisting of small perturbations of equilibrium. This Burgers-like equation can be stabilized in the same way as the one-dimensional Burgers equation, and the use of artificial viscosity is the most natural (and efficient) method for this purpose. As shown in [5], the ϖ equation (24b) (with and without regularization) is locally well-posed for analytic data; for such data (with periodic boundary conditions), there exists a limit of zero artificial viscosity.

On the other hand, since the foundational work of Chorin and Bernard [22], it is very natural to use the vortex blob method (to be described below in Section 4.1) to solve for z_1 and z_2. While there are many other choices for regularizing the z-model, using this combination of artificial viscosity for ϖ and vortex blobs for z_1 and z_2 produces an efficient and stable algorithm which allows for convergence of the large-scale structures of the flow (such as bubble and spike locations), as will be demonstrated below.

An alternative approach to our multiscale decomposition might have been to decompose solutions of the compressible Navier-Stokes equations into large-scale and small-scale velocities, but special (and very restrictive) interface conditions would then be required to keep the interface sharp, while the standard interface conditions would instead enforce continuity of the velocity across the interface. While vortex methods for viscous flows [27] have been developed by Chorin [20,21] using the random walk method and by Degond and Mas-Gallic [33] using weighted particle methods, we instead rely upon a simple artificial viscosity scheme restricted to the interface for our multiscale algorithm which is computationally less expensive than the vortex methods for viscous flows.

4. Numerical implementation of the z-model

Our multiscale model will rely on a fast-running numerical implementation of the higher-order z-model (24). In this section, we explain the method, and perform some classical numerical experiments to demonstrate the efficacy of our scheme.

4.1. A regularization of the incompressible z-model

A simple method for approximating the singular integral on the right-hand side of (24a) is to use a standard trapezoidal quadrature rule. This is the original *point vortex method* of Rosenhead [82]. Unfortunately, as demonstrated in [55,22], solutions computed using the point vortex method often suffer from irregular point vortex motion due to small perturbation errors introduced by round-off error.[2] Such irregular motion is then amplified by the Kelvin-Helmholtz instability. Moreover, this irregular motion persists as the mesh is refined, and is in fact initiated at earlier times as the number of nodes increases.

In [56,57], the equation (24a) is *desingularized* by smoothing the singular kernel $\mathcal{K}_{\mathbb{T}_L}$ to yield the desingularized kernel

$$\mathcal{K}^{\delta}_{\mathbb{T}_L}(x) = \frac{(-\sinh(2\pi x_2/L)\,,\sin(2\pi x_1/L))}{2L\left(\delta^2+\cosh(2\pi x_2/L)-\cos(2\pi x_1/L)\right)}\,, \tag{30}$$

with $\delta\in\mathbb{R}$ some constant. This yields a more numerically stable set of equations to which the standard trapezoidal quadrature rule can be applied. Computational evidence [56] suggests that this approximation converges beyond the singularity time if the mesh is refined and the smoothing parameter δ is decreased, in the appropriate order. In our numerical experiments, we have found that the scaling

$$\delta^2 = |\triangle\alpha\log\triangle\alpha|\cdot\tilde{\delta}^2\,, \tag{31}$$

with $\tilde{\delta}$ a constant, yields stable solutions with increasing amounts of roll-up as $\triangle\alpha\to 0$. The details of how the parameter $\tilde{\delta}$ is chosen are provided in §4.3.2, in which we provide an example of the procedure applied to a KHI test problem.

The full-space version of the desingularized kernel (30) is given by

$$\mathcal{K}^{\delta}_{\mathbb{R}}(x) = \frac{1}{2\pi}\frac{x^{\perp}}{\left(|x|^2+\delta^2\right)}\,. \tag{32}$$

We will make use of (32) in the numerical experiments in Section 4.3.

Convergence of the point vortex and vortex blob methods for smooth flows is proved in [47,12,43]. For vortex sheets, where the initial data is not smooth, Caflisch and Lowengrub [16] proved global existence of analytic solutions from arbitrary analytic initial data for the desingularized equations in the case $A=0$. They also proved short time convergence of the

[2] Few digits of precision can also be regularizing, as shown by the calculations of Rosenhead [82].

vortex blob method as the desingularization parameter δ, mesh size, and time-step converge to zero. When the sheet is analytic, the error due to the desingularization is $\mathcal{O}(\delta)$ [16], assuming round-off errors are sufficiently small. Convergence for weak solutions was proved by Liu and Xin [63] in the case that the vorticity measure is of distinguished sign (and the Atwood number vanishes, $A = 0$).

Let us note that the full-space desingularized kernel (32) satisfies the sufficient conditions of the theorem of Liu and Xin [63], whereas the periodic desingularized kernel (30) does not satisfy the assumptions. Consequently, it is not known whether the solutions to the system induced by the regularized kernel (30) converge to a weak solution of the incompressible Euler system. Nonetheless, there is numerical evidence to suggest that the numerical solution does indeed converge [56,8].

Next, we turn to the evolution equation for the amplitude of vorticity ϖ. The nonlinearity in (24b) often results in the development of steep profiles of the variable ϖ, analogous to the formation of shocks for solutions to nonlinear conservation laws. The shock formation in ϖ generally occurs at late times, and is followed by the roll-up of the vortex sheet. One can handle this phenomenon by using shock-capturing methods (see for instance Sohn [92]). However, we use the simplest possible technique, namely a linear artificial viscosity operator $\mu\partial_\alpha^2$ with $\mu \geq 0$, to *smear* the shock over a small number of cells and thereby stabilize the solution.

We therefore consider the following *regularization* of the higher-order z-model (24) as follows:

$$\partial_t z(\alpha,t) = \int_{\mathbb{T}_L} \mathcal{K}^\delta_{\mathbb{T}_L}(z(\alpha,t) - z(\beta,t))\varpi(\beta,t)\,\mathrm{d}\beta\,, \tag{33a}$$

$$\partial_t \varpi(\alpha,t) = -\partial_\alpha\left[\frac{A}{2|\partial_\alpha z(\alpha,t)|^2}\mathcal{H}(\varpi(\alpha,t)\mathcal{H}\varpi(\alpha,t)) - 2Agz_2(\alpha,t)\right] + \mu\partial_\alpha^2\varpi(\alpha,t)\,. \tag{33b}$$

When $\delta = 0$, the Cauchy principal value of the integral in (33a) must be taken, while for $\delta > 0$, the integral is proper and the equations are then a regularized approximation to periodic vortex sheet evolution.

4.1.1. Other regularizations of the singular kernel (20)

For the purposes of comparison with the Krasny approximation, we consider two other desingularized kernels that approximate the singular kernels (20) or (18). These kernels were derived by Baker and Beale [7] (see also [11]), and are of the form

$$\mathcal{K}^\delta_i(x) = \mathcal{K}(x)(1 + g_i(\eta_\delta))\,, \tag{34}$$

where the subscript i is related to the number of moment conditions satisfied by the kernel (see [7] for the details). More precisely, we shall consider g_i for $i = 1, 3$, in which case we have

$$g_1(\eta_\delta) = -\exp(-\eta_\delta^2)\,,$$
$$g_3(\eta_\delta) = (-1 + 2\eta_\delta^2)\exp(-\eta_\delta^2)\,.$$

Here, $\mathcal{K}(x)$ refers to either the $\mathbb{R}^2$ kernel (18) or the periodic kernel (20). In the former case, the variable η_δ is given by $\eta_\delta = |x|/\delta$, while in the latter case, $\eta_\delta^2 = 2\left(\cosh(x_2) - \cos(x_1)\right)/\delta^2$.

Formula (34) is indeed a desingularization due to the fact that the functions g_i satisfy $1 + g_i(\eta_\delta) = \mathcal{O}(\eta_\delta^2)$ as $\eta_\delta \to 0$, whereas the singular kernel $\mathcal{K}(x)$ has an $\mathcal{O}(1/x)$ type singularity at the origin, so that the denominator is canceled and $\mathcal{K}^\delta_i(x)$ is a smooth function of x. We note that the exponential decay of (34) as $|\eta_\delta| \to \infty$ is in contrast to the slower algebraic decay of the Krasny desingularization (30).

The kernels $\mathcal{K}^\delta_i$ satisfy the sufficient conditions of the theorem of Liu and Xin [63], and consequently, solutions to the regularized system converge as $\delta \to 0$, when the initial vorticity amplitude ϖ is of distinguished sign.

4.2. Discretization of (33)

Suppose that a single wavelength of the periodic interface $\Gamma(t)$ is parametrized by the function $z(\alpha,t)$, and that the parameter α is discretized with $N + 1 = 2^r + 1$ nodes,

$$\alpha_k = -L/2 + (k-1)\triangle\alpha\,,$$

with $\triangle\alpha = L/N$.

We spatially discretize the equations of motion, then use a standard third-order explicit Runge-Kutta solver for time integration. A trapezoidal quadrature rule is used to approximate the right-hand side to the z-equation. Define for $k = 1, \ldots, N+1$ the functions $G_k(\alpha,t) : \mathbb{T}_L \times [0,T] \to \mathbb{R}^2$ on the discretized domain as

$$G_k(\alpha_l,t) = \begin{cases} \mathcal{K}^\delta_{\mathbb{T}_L}(z_k(t) - z_l(t))\,\varpi_l(t) & \text{, if } l \neq k\,, \\ 0 & \text{, if } l = k\,, \end{cases}$$

where we have used the notation $f_k(t) = f(\alpha_k, t)$. The right-hand side to the z-equation (33a) may then be approximated as

$$\frac{\mathrm{d}}{\mathrm{d}t} z_k(t) = \frac{\Delta\alpha}{2} G_k(\alpha_1, t) + \Delta\alpha \sum_{l=2}^{N} G_k(\alpha_l, t) + \frac{\Delta\alpha}{2} G_k(\alpha_{N+1}, t)\,. \tag{36}$$

The trapezoidal rule we employ, while in general is only second order accurate, achieves spectral accuracy when the integrand is smooth and the mesh is uniform [8].

For the ϖ equation, we follow [44] and convert the equation to Fourier space. Using the identities (1) and (5), we write the ϖ-equation (33b) in frequency space as

$$\begin{aligned} \partial_t \widehat{\varpi}(\xi, t) = &-\frac{A}{2}\left(\frac{2i\pi}{L}\xi\right) \mathcal{F}\left\{\frac{1}{|\partial_\alpha z|^2}\mathcal{F}^{-1}\left\{-i\,\mathrm{sgn}(\xi)\mathcal{F}\left\{\varpi\,\mathcal{F}^{-1}\left\{-i\,\mathrm{sgn}(\xi)\widehat{\varpi}\right\}\right\}\right\}\right\}(\xi, t) \\ &+ 2Ag\left(\frac{2i\pi}{L}\xi\right)\widehat{z}_2(\xi, t) - \mu\left|\frac{2\pi}{L}\xi\right|^2 \widehat{\varpi}(\xi, t)\,, \end{aligned} \tag{37}$$

where $\mathcal{F}^{-1}\{\cdot\}$ denotes the inverse Fourier transform operator.

The discretized version of (37) then becomes

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}(\widehat{\varpi}_k) = &-\frac{A}{2}\left(D_k\tilde{\mathcal{F}}\left\{\left(\frac{1}{|\partial_\alpha z_l|^2}\tilde{\mathcal{F}}^{-1}\left\{\left(H_m\tilde{\mathcal{F}}\left\{\left(\varpi_n\tilde{\mathcal{F}}^{-1}\left\{(H_r\widehat{\varpi}_r)\right\}_n\right)\right\}_m\right)\right\}_l\right)\right\}_k\right) \\ &+ 2Ag\left(D_k\widehat{z}_2^k\right) - \Delta\alpha\cdot\tilde{\mu}\left(D_k^2\widehat{\varpi}_k\right)\,, \end{aligned} \tag{38}$$

where we have used (6)-(8) to denote the discrete Hilbert and derivative operators in Fourier space. We remark that we are not using the usual summation convention in (38), and instead use the notation (f_k) to denote the vector with entries f_k for $k = 1, \ldots, N$, so that $(f_k g_k)$ denotes the vector with entries $f_k g_k$ for $k = 1, \ldots, N$.

Let us note that we have used the scaling $\mu = \Delta\alpha\cdot\tilde{\mu}$ for the artificial viscosity parameter for the ϖ-equation. In general, we shall keep $\tilde{\mu}$ fixed as the resolution $\Delta\alpha$ varies, but note that it is often necessary to vary $\tilde{\mu}$ with the resolution to stabilize small-scale noise that may occur in the variable ϖ.

Equations (36) and (38) form a nonlinear system of coupled ordinary differential equations, to which we can apply a standard third-order explicit Runge-Kutta time integration scheme. We supplement the equations with initial data $z_k(0)$ and $\varpi_k(0)$, as well as periodic boundary conditions $z_1^{N+1}(t) = L + z_1^1(t)$, $z_2^{N+1}(t) = z_2^1(t)$, and $\varpi_{N+1}(t) = \varpi_1(t)$ for all $t \geq 0$.

The direct summation method (36) employed for the integral calculation (33a) is $\mathcal{O}(N^2)$, and is thus inefficient for large values of N. Other methods have been proposed to reduce the computational complexity of the velocity calculation. For instance, one technique is the so-called "vortex-in-cell" method [23,6,98], in which the velocity of a point vortex is computed by solving a Poisson equation on an underlying mesh, and interpolation is used to compute values on the interface. This method reduces the computational cost by virtue of the use of fast Poisson solvers, and appears to accurately predict the large-scale behavior of the vortex sheet, but does not seem suitable for the study of small-scale behavior [99].

Other fast summation methods include the Fast Multipole Method of Greengard and Rokhlin [45] (see also [17]), the Barnes-Hut algorithm [10], and various other so-called "treecode" algorithms [4,100,2,48,37,84,61]. Such methods reduce the computational complexity of the summation to $\mathcal{O}(N)$ or $\mathcal{O}(N\log N)$ by combining large numbers of point vortices into single computational elements. However, they are often complicated to implement since the computations must be organized in a manner that leads to an efficient and accurate algorithm. Moreover, such algorithms often have significant computational overhead that make them efficient only for large values of N. In the numerical simulations considered in the current paper, we restrict our attention to problems requiring only relatively small values of N; for such problems, the direct summation method we employ is likely comparable (in terms of efficiency and CPU time) to the more sophisticated algorithms mentioned above. In future studies, we shall implement a fast summation method to study vortex sheet evolution for large values of N.

Following Krasny [57], we reduce the computational expense of calculating (36) in the following two ways: first, we use the relation $\mathcal{K}^{\delta}_{\mathbb{T}_L}(z_k - z_l) = -\mathcal{K}^{\delta}_{\mathbb{T}_L}(z_l - z_k)$ so that the calculation (36) is required for only half the points; second, for problems which are symmetric about $\alpha = 0$, we compute (36) for only half the points and use reflection to obtain the values for the rest.

4.3. Numerical studies and discussion

We next conduct several numerical studies to validate our regularized z-model system, as well as its numerical implementation. In particular, we shall compare the Krasny desingularization (30) with the two other desingularizations (34). We provide numerical evidence to show that all three numerical methods produce similar solutions, and that the computed numerical solutions appear to converge. The situation is somewhat complicated by the fact that there is a large gap in the theory of vortex sheet evolution when the vorticity does not have distinguished sign. In particular, the question of existence

of solutions to the incompressible Euler system when the vorticity is a measure but is not of distinguished sign is open; additionally, the question of uniqueness is open, even when the vorticity is of distinguished sign. Consequently, comparisons of the numerical methods as the mesh is refined is complicated due to the fact that the convergence may be towards different solutions. Nonetheless, in agreement with prior numerical studies [8,93], we find that the computed numerical solutions agree, in a sense to be made precise below.

Specifically, the quantities that we shall be interested in with regards to our convergence studies are (1) the bubble and spike tip locations, (2) the radius of the spiral roll-up region, and (3) the location of the center of the roll-up region. Quantity (1) provides some information about the convergence of solutions "at the large scales", while the quantities (2) and (3) provide information about the convergence of solutions "at the small scales". The bubble and spike tip locations are defined by $\max_i z_2(\alpha_i)$ and $\min_i z_2(\alpha_i)$, respectively, while the radius and center of spiral roll-up region are computed as follows. We first find the intersection points $\left\{(x_1^*, x_2^*)\right\}$ of the computed curve z with a fixed horizontal axis $x_2 = x_2^*$. These intersection points are computed by bilinear interpolation. The radius r_δ and location of the center σ_δ of the spiral region may then be approximated as $r_\delta \approx (\max x_1^* - \min x_1^*)/2$ and $\sigma_\delta \approx (\max x_1^* + \min x_1^*)/2$, respectively. The subscript δ indicates that these quantities depend on the regularization parameter δ (as well as the mesh resolution $\triangle\alpha$).

In the convergence studies presented, we will be interested in two different limits: the first is the limit $\delta \to 0$ with N held fixed, and the second is the limit $\delta \to 0$ and $N^{-1} \to 0$. In the latter case, it is important exactly how the limits are taken [56]. For the Krasny desingularization, we will use the scaling (31), and we will show that the resulting solutions are stable with increasing amounts of roll up as $N \to \infty$.

We are unaware of scaling laws similar to (31) for the kernels of (34), and will instead use the following empirical (though tedious and computationally expensive) procedure employed by Anderson [3] and Krasny [56]. This empirical method amounts to fixing a value of $\triangle\alpha$, say $\triangle\alpha = \triangle\alpha_1$, then choosing the smallest $\delta = \delta_1$ such that the computed numerical solution is stable for every $\delta > \delta_1$. This procedure is then repeated for $\triangle\alpha_2 < \triangle\alpha_1$, yielding $\delta_2 < \delta_1$. In this way, a sequence $(\triangle\alpha_1, \delta_1), (\triangle\alpha_2, \delta_2), \ldots$ is constructed, with $\delta_i, \triangle\alpha_i \to 0$, and we are able to discuss the limit $\delta \to 0$ and $N^{-1} \to 0$.

One of our goals in this section is to justify our use of the Krasny kernel in the numerical implementation of our multiscale algorithm in Section 5. We shall show that the Krasny kernel produces solutions with similar asymptotic behavior (i.e. as $\delta, N^{-1} \to 0$) as those produced using the kernels (34). On the other hand, calculations using the Krasny kernel require less computational expense than the corresponding calculations using the kernels of (34). Consequently, as our goal in Section 5 is to produce a fast-running algorithm for compressible flow simulations, we will use the Krasny approximation rather than the kernels (34).

We remark that, in general, we will be restricted to using relatively large values of the regularization parameter δ for the 3rd-order kernel $\mathcal{K}_3^\delta$, compared to those for the lower order Krasny and $\mathcal{K}_1^\delta$ kernels. This is due to the fact that a large amount of nodes N is required to fully resolve the small scale structure that is observed with the 3rd order kernel [7], which proves prohibitively computationally expensive for our purposes.

Standard MATLAB plotting routines have been employed to present interface evolution; in particular, we follow Krasny [57] and use trigonometric polynomials of degree $N/2$ to interpolate the discrete computed interface nodal positions $z_k(t)$.

4.3.1. KHI test on an ellipse: comparison with an exact solution

We begin with a numerical experiment for which there is a known exact solution; namely, we consider the KHI problem of Baker and Beale [7]. For this test, we compute the velocity induced by a vorticity measure concentrated on an ellipse. More precisely, we set the Atwood number to be zero, $A = 0$, so that the amplitude of vorticity ϖ remains constant over time, and choose the initial data as

$$\begin{aligned} z_1(\alpha, 0) &= \lambda \cosh(\sigma)\cos(\alpha)\,, \\ z_2(\alpha, 0) &= \lambda \sinh(\sigma)\sin(\alpha)\,, \\ \varpi(\alpha, 0) = \varpi(\alpha, t) &= \sin(\alpha)\,, \end{aligned}$$

with $\alpha \in [0, 2\pi]$. The parameter λ measures the eccentricity of the ellipse, with $\lambda \to 0$ yielding a circle, and $\lambda \to 1$ yielding a slit. The constant σ is determined from λ by the relation $\lambda \cosh(\sigma) = 1$.

The exact solution [7] is of the form

$$w_1(\alpha) = \frac{R}{4\lambda D}\,, \quad \text{and} \quad w_2(\alpha) = \frac{I}{4\lambda D}\,,$$

with

$$\begin{aligned} R &= 2e^{-\sigma}\left(2\sinh^2(\sigma)\cos^2(\alpha) + e^{-\sigma}\cosh(\sigma)\sin^2(\alpha)\right)\,, \\ I &= \sinh(\sigma)\sin(2\alpha)\,, \qquad D = \cosh^2(\sigma) - \cos^2(\alpha)\,. \end{aligned}$$

Since the curve z is a closed curve, we use the regularized versions of the $\mathbb{R}^2$ kernel (18) i.e. (32) and (34). We set $\lambda = 0.01$, consider $N = 16, \ldots, 512$ in increasing powers of 2, and consider two different values of $\delta/\triangle\alpha$.

Table 1
Error analysis and convergence results for the exact solution test of Baker and Beale [7]. Shown are the number of digits of accuracy, $-\log_{10} E_N$, for the three different numerical schemes employed.

N	$\delta = \triangle\alpha$			$\delta = \triangle\alpha/4$		
	$\mathcal{K}_1^\delta$	$\mathcal{K}_3^\delta$	Krasny	$\mathcal{K}_1^\delta$	$\mathcal{K}_3^\delta$	Krasny
16	0.960	2.440	0.820	1.204	1.204	1.175
32	1.258	3.320	1.057	1.505	1.505	1.490
64	1.558	4.031	1.318	1.806	1.806	1.798
128	1.859	4.484	1.599	2.107	2.107	2.103
256	2.160	4.833	1.891	2.408	2.408	2.406
512	2.461	5.147	2.187	2.709	2.709	2.708

We measure the L^∞ error $E_N = \max_{i=1,\dots,N} |w(\alpha_i) - w_i|$, where $w(\alpha_i)$ is the exact velocity and w_i is the computed velocity. Then, the quantity $-\log_{10} E_N$ denotes the number of digits of accuracy of the computed solution (see [7]). In Table 1, we list $-\log_{10} E_N$ for the three different methods of desingularizing the integral kernel.

For the larger value $\delta = \triangle\alpha$, we see that the higher order kernel $\mathcal{K}_3^\delta$ is the most accurate of the three methods, with the computed velocity accurate to 5 digits when $N = 512$; the lower order kernel $\mathcal{K}_1^\delta$ and the Krasny kernel perform similarly, with the computed velocity accurate to 2 digits when $N = 512$. For the smaller value $\delta = \triangle\alpha/4$, all three regularizations produce a velocity that is accurate to 2 digits. Thus, in the limit $\delta \to 0$ and $N \to \infty$, all three methods of regularization perform similarly; the main difference is then the fact that the kernels $\mathcal{K}_i^\delta$ are more expensive to compute than the Krasny kernel (32).

4.3.2. KHI problem on a periodic curve: test of the Krasny δ-regularization

The purpose of the following test is to demonstrate how the regularization parameter δ scales with the mesh resolution $\triangle\alpha$. The procedure we employ for determining the appropriate value of $\tilde{\delta}$ for use in the Krasny desingularization method is as follows: fix a relatively small value of N, say $N = 32$ or $N = 64$. Next, we find the smallest value of $\tilde{\delta}$ such that the computed interface demonstrates roll up, but without self-intersection. Finally, we fix this value of $\tilde{\delta}$, and use the scaling relation (31) for larger values of N.

Below, we provide an example of the above procedure applied to a periodic KHI problem. In this case, the Atwood number vanishes $A = 0$, and thus the amplitude of vorticity ϖ remains constant over time. The initial data is [7]

$$\begin{aligned} z_1(\alpha, 0) &= \alpha\,, \\ z_2(\alpha, 0) &= 0\,, \\ \varpi(\alpha, 0) = \varpi(\alpha, t) &= 1 - 0.5\cos(\alpha)\,, \end{aligned}$$

with $\alpha \in [0, 2\pi]$. Numerical studies indicate that a curvature singularity forms at $\alpha = \pi$ and $t \approx 1.61$, after which time the sheet rolls up in a tightly wound spiral.

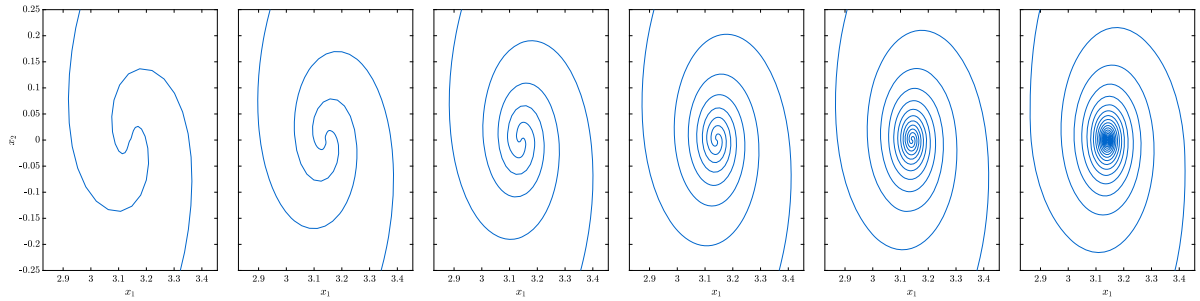
Experimentation with the value of $\tilde{\delta}$ with $N = 64$ shows that choosing $\tilde{\delta} = 0.15$ produces an interface which demonstrates roll-up but for which self-intersection does not occur (see Fig. 3(a)). A smaller value of $\tilde{\delta}$, say $\tilde{\delta} = 0.1$, produces an interface which self-intersects. The runtime for such a simulation is < 1 s, so that experimentation with the precise value of $\tilde{\delta}$ is not computationally expensive. Once the value of $\tilde{\delta}$ is set for $N = 64$, the same value is chosen for $N > 64$. The computed results are presented in Fig. 3. We observe that the scaling relation $\delta^2 = |\triangle\alpha \log \triangle\alpha| \cdot \tilde{\delta}^2$ results in more turns appearing in the core region as N increases, but that the solution remains stable and the curve does not self-intersect or suffer from irregular vortex motion.

Next, we consider the convergence of the numerical solution as $\delta\,, N^{-1} \to 0$. The convergence of solutions as $\delta \to 0$ with N fixed is considered in detail in [56,8], wherein it is shown that the computed numerical solution appears to converge.

In Fig. 4(a), we show how the amplitude $\max z_2$ of the curve varies as $\delta, N^{-1} \to 0$. The value for N^{-1} is obtained by cubic extrapolation. We see that the amplitude appears to converge to a finite value ≈ 0.81. Similarly, in Fig. 4(b), we estimate the radius of the spiral region by $r_\delta \approx \pi - \min x_1^*$, where x^* are the intersection points with the axis $x_2 = x_2^* = 0$. By symmetry, the center of the spiral is located at $(\pi\,, 0)$. Again, with the value for $N^{-1} = 0$ obtained by cubic extrapolation, we see that the r_δ appears to converge to a value close to 0.28. This demonstrates that the scaling (31) appears to be appropriate for recovering a meaningful solution as $\delta, N^{-1} \to 0$.

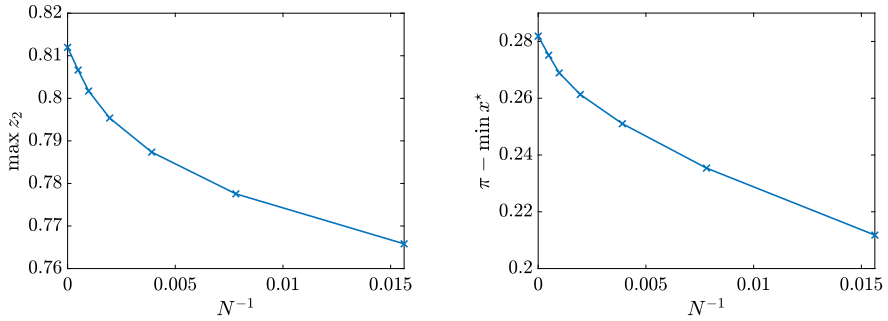
4.3.3. Single-mode RTI: comparison with experiments

We continue our numerical studies for the z-model by performing simulations for the low Atwood number single-mode RTI experiments of Waddell et al. [104]. The particular problem setup considered is a heavy fluid lying atop a lighter fluid, with the Atwood number given by $A = 0.155$ and the two fluids subject to an approximately constant gravitational acceleration $g = 0.74 \times 9.8\text{ms}^{-2}$. The z-model is employed for this problem on the domain $\alpha \in [-0.027, 0.027]$ with initial data



(a) $N = 64$ (b) $N = 128$ (c) $N = 256$ (d) $N = 512$ (e) $N = 1024$ (f) $N = 2048$

Fig. 3. Numerical simulation of a KHI test using the z-model with the Krasny desingularization. Shown is the interface position $z(\alpha, t)$ at time $t = 3.0$ for six simulations with resolution starting from $N = 64$ and doubling until $N = 2048$.



(a) Amplitude $\max z_2$. (b) Radius of the spiral region $r_\delta \approx \pi - \min x_1^*$.

Fig. 4. Convergence behavior for a KHI test using the z-model with the Krasny desingularization. Shown are (a) the amplitude of the curve $\max z_2$ and (b) the radius of the spiral region r_δ.

$$z_1(\alpha, 0) = \alpha\,,$$

$$z_2(\alpha, 0) = 0.0012\cos(2\pi\alpha/L)\,,$$

$$\varpi(\alpha, 0) = 0\,.$$

We perform six simulations with resolution starting from $N = 32$ and doubling until $N = 1024$. The strategy for parameter choice we adopt here is to keep the parameters $\tilde{\delta} = 2$ and $\tilde{\mu} = 0.02$ fixed as N varies, while allowing the time-steps δt to vary with N. Specifically, we first choose δt for $N = 64$ as the largest possible value that will allow the $N = 64$ simulation to run until the final time $t = 0.3795$. The values of δt for larger N are then determined by repeatedly halving this value until δt is sufficiently small so as to allow the simulation to complete.

The results of the six simulations are shown in Fig. 5, which should be compared with Fig. 4(l) of [104]. The bubble and spike shapes are roughly symmetric, in agreement with the observations in [104]. The scaling for the regularization parameters $\tilde{\delta}$ and $\tilde{\mu}$ results in more roll-up of the vortex sheet as the resolution is increased. This is demonstrated in Fig. 6, which shows that the interface for the $N = 1024$ simulation has a tightly packed spiral region with several complete revolutions of each branch of the spiral.

We next compare the growth rate of the z-model solution interface with growth rates obtained from small-time linear theory predictions and late-time experimental observations. Define the amplitude of the interface as $a(t) = \frac{1}{2}[\max_\alpha z_2(\alpha, t) - \min_\alpha z_2(\alpha, t)]$. Linear theory [67,96] predicts that for early times t before the non-linearity is activated, the amplitude satisfies $a(t) = a(0)\cosh(t\sqrt{2\pi Ag/L})$. We plot in Fig. 7(a) the linear theory amplitude and the computed amplitude $a(t)$ versus time for the simulations shown in Fig. 5. It is clear from the graph that the computed amplitude and linear prediction are in excellent agreement, as expected, for small times $t \leq 0.15$.

For large times, the nonlinearity is no longer negligible and the linear theory breaks down. Experimental observations indicate that the amplitude grows linearly at late times; a linear fit of the measured late time amplitude from experimental data is shown as the blue curve in Fig. 7(b). This curve is defined in [104] as $-0.007436 + 0.078177t$. It is clear from the graph that the measured amplitude differs considerably from the amplitude computed using the z-model. This difference

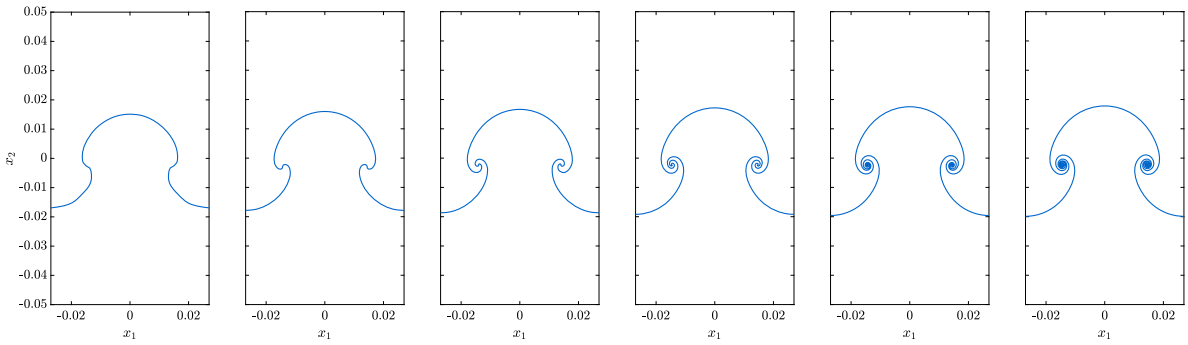


Fig. 5. Numerical simulation of the Waddell et al. [104] RTI using the z-model. The interface parameterizations $z(\alpha, t)$ for six simulations with resolution starting from $N = 32$ and doubling until $N = 1024$ are shown at time $t = 0.3795$.

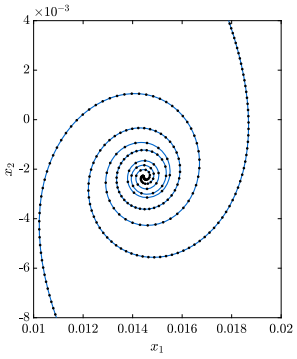


Fig. 6. Closeup view of the roll-up region for $N = 1024$.

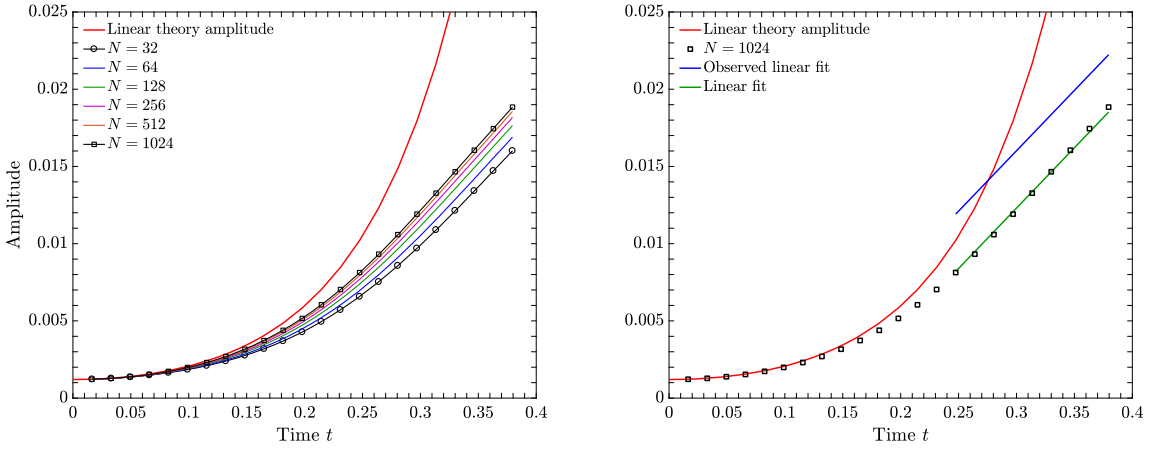
may be explained by the fact that the (effective) gravitational acceleration in the experiments is only approximately constant, and is in fact time-dependent, whereas we have used[3] a constant value of g.

While the amplitudes themselves differ, we nonetheless observe that the growth *rates* of the computed amplitude and measured amplitude are in excellent agreement. The green curve shown in Fig. 7(b) has the same slope as the blue curve (i.e. the measured amplitude) and is given explicitly as $-0.01115 + 0.078177t$. This curve matches almost exactly with the computed $a(t)$ for large times $t \geq 0.25$. In fact, the true gravitational acceleration in the experimental setup is initially time-dependent, but eventually reaches a constant value; in our numerical simulation, we have used this final constant value for g, which is why the amplitude of the numerical solution at large times $t \geq 0.25$ grows at the same rate as that observed in the experiment. The numerical solution and the experimental data are thus in good agreement (modulo the issue regarding the non-constant value of g), which consequently provides validating evidence for the z-model.

Next, we consider the convergence of the computed numerical solution in the limits $\delta \to 0$ with $N = 256$ fixed. Fig. 8(a) shows that the bubble and spike tip locations appear to converge linearly, which agrees with the results of [56] for the KHI test. Here, the value at $\delta = 0$ is obtained by linear extrapolation. In Fig. 8(b) and Fig. 8(c), we show the convergence of the computed spiral center σ_δ and radius r_δ. We choose the axis $x_2 = x_2^* = -0.0023$ to compute the intersection points x^*. Again, the computed values appear to converge, though the precise nature of this convergence is less clear. In this case, we obtain the value at $\delta = 0$ via cubic interpolation.

We repeat our convergence tests for $N = 256$ fixed and $\delta \to 0$ with the kernels (34), and compare the results with the Krasny kernel results in Fig. 9. We find excellent agreement between all three methods in the limiting behavior of the bubble and spike tip locations, as shown in Fig. 9(a); this is in agreement with previous numerical studies [8,93]. The small scale structure of the limiting solutions are slightly different, with the Krasny kernel and first order kernel $\mathcal{K}_1^\delta$ producing similar spiral center locations σ_δ and radii r_δ. Here, the value at $\delta = 0$ is obtained by cubic extrapolation. Let us note that for $\delta > 0$, all three methods predict similar values of σ_δ and r_δ.

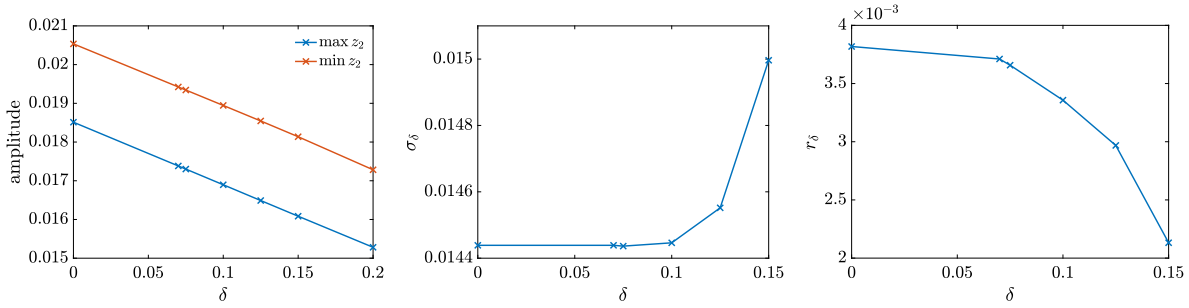
[3] We note that it is of course possible to have time-dependent g for z-model simulations. Our use of a constant value of g for this experiment is due to the fact that only an approximate constant value of g is provided in [104].



(a) Comparison of the computed amplitude and linear theory predictions for the simulations in Figure 5.

(b) Comparison of the computed amplitude for the $N = 1024$ simulation with large time linear fits.

Fig. 7. Plots of the amplitude $a(t)$ versus time t for the Waddell et al. [104] RTI problem. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

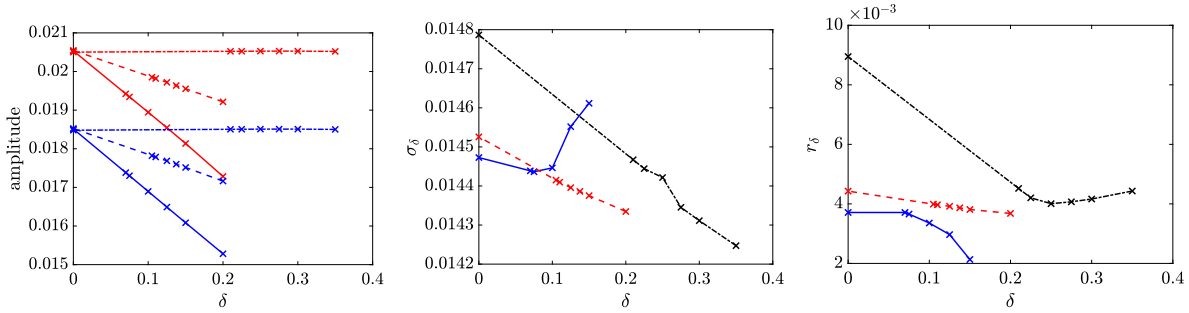


(a) Bubble and spike tip locations.

(b) Location of spiral center σ_δ.

(c) Radius of spiral r_δ.

Fig. 8. Convergence behavior for the RTI test of Waddell et al. [104] using the z-model with the Krasny desingularization. Shown are (a) the bubble and spike tip locations, $|\max z_2|$ and $|\min z_2|$, respectively, (b) the location of the center of the spiral region σ_δ, and (c) the radius of the spiral region r_δ.

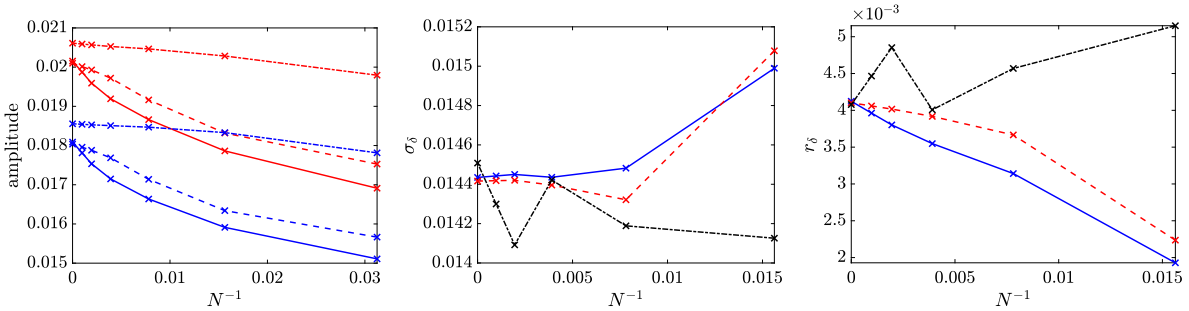


(a) Bubble (blue) and spike (red) tip locations.

(b) Location of spiral center σ_δ.

(c) Radius of spiral r_δ.

Fig. 9. Convergence behavior as $\delta \to 0$ with $N = 256$ fixed for the RTI test of Waddell et al. [104] using the z-model. Shown are (a) the bubble and spike tip locations, $|\max z_2|$ and $|\min z_2|$, respectively, (b) the location of the center of the spiral region σ_δ, and (c) the radius of the spiral region r_δ. The solid, dashed, and dotted curves in (a) refer to the Krasny, $\mathcal{K}_1^\delta$, and $\mathcal{K}_3^\delta$ kernels, respectively. The blue, red, and black curves in (b) and (c) refer to the Krasny, $\mathcal{K}_1^\delta$, and $\mathcal{K}_3^\delta$ kernels, respectively.



(a) Bubble (blue) and spike (red) tip locations.

(b) Location of spiral center σ_δ.

(c) Radius of spiral r_δ.

Fig. 10. Convergence behavior as $\delta, N^{-1} \to 0$ for the RTI test of Waddell et al. [104] using the z-model. Shown are (a) the bubble and spike tip locations, $|\max z_2|$ and $|\min z_2|$, respectively, (b) the location of the center of the spiral region σ_δ, and (c) the radius of the spiral region r_δ. The solid, dashed, and dotted curves in (a) refer to the Krasny, $\mathcal{K}_1^\delta$, and $\mathcal{K}_3^\delta$ kernels, respectively. The blue, red, and black curves in (b) and (c) refer to the Krasny, $\mathcal{K}_1^\delta$, and $\mathcal{K}_3^\delta$ kernels, respectively.

Next, we consider the limit $\delta, N^{-1} \to 0$, simultaneously; for the Krasny kernel, we use the scaling (31), while for the kernels $\mathcal{K}_i^\delta$ we use the empirical procedure discussed at the beginning of Section 4.3. We consider six simulations with resolution starting from $N = 32$ and doubling until $N = 1024$, and again compute the bubble/spike tip locations and quantities σ_δ and r_δ. We find excellent agreement between all three methods for the limiting values of each of the relevant quantities, see Fig. 10. Moreover, the Krasny scheme is the least computationally expensive method: for the $N = 1024$ simulation, the runtime for the Krasny scheme is $T_{\text{CPU}} \approx 333$ s, whereas the runtime for the 3rd order kernel $\mathcal{K}_3^\delta$ scheme is $T_{\text{CPU}} \approx 469$ s, and thus the simpler Krasny scheme is 40% faster. As such, we conclude that, as the simplest and least computationally expensive method, the Krasny desingularization is the most suitable method for our objective.

4.3.4. Single-mode RTI: comparison with numerical simulations

Next, we compare the numerical simulations of Sohn [92] that used the discretized equations (22), with our higher-order z-model solutions. A brief description of the numerical method in [92] is as follows. The z-equation (22a) is treated in an identical fashion as in our numerical framework; in particular, the same Krasny δ-desingularization and trapezoidal rule is used. The ϖ-equation (22b) is discretized in physical space, rather than in Fourier space as in our numerical method. The nonlinear term is treated using upwinding via the Godunov method, while an iterative procedure is used for the time-derivative term appearing on the right-hand side of (22b). Let us remark that, on average, 6 iterations per time-step are required for this numerical method.

Since we are interested in capturing vortex sheet roll-up, we consider the low Atwood number RTI test problem from [92]. The domain is $\alpha \in [-\pi, \pi]$, the Atwood number is $A = 0.05$, the gravitational constant is $g = 1$, and the initial data is

$$z_1(\alpha, 0) = \alpha\,,$$
$$z_2(\alpha, 0) = 0.5\cos(\alpha)\,,$$
$$\varpi(\alpha, 0) = 0\,.$$

The z-model is run for this problem with $N = 400$, which is the same value employed in [92], but whereas the value $\delta t = 0.002$ is required in [92], we are able to use the much larger $\delta t = 0.025$. The regularizing parameters chosen as $\tilde{\delta} = 0.6$ and $\tilde{\mu} = 0.005$. This value of $\tilde{\delta}$ was chosen to agree with the parameter choices in [92].

The computed z for the above numerical experiment is shown in Fig. 11 at various times t. This figure should be compared with Fig. 1(a) in [92], upon which it is clear that the two are essentially indistinguishable. In particular, we note that the two solutions are in excellent agreement in the roll-up region, with both branches of the spirals having almost four full rotations at the final time $t = 22$.

To quantify the agreement between our z-model solution and the reference solution from [92], we plot various computed quantities in Fig. 12. The bubble tip and spike tip speeds versus time are shown in Fig. 12(a) (compare with Fig. 3(a) in [92]). Shown also are theoretical predictions for the bubble speed from asymptotic potential flow models. Such models aim to describe analytically the evolution of the amplitude of the interface z, after the transition from exponential growth $z \sim z_0 e^{t\sqrt{Ag}}$ at small times (as predicted by the linear theory) to linear-in-time growth $z \sim w_\infty t$ (as observed in experiments, for instance), where w_∞ is the asymptotic bubble velocity. The Goncharov [42] and Sohn [91] models estimate w_∞ as

$$w_\infty = \sqrt{\frac{2A}{1+A}\frac{g}{3}} \qquad \text{and} \qquad w_\infty = \sqrt{\frac{Ag}{2+A}}\,,$$

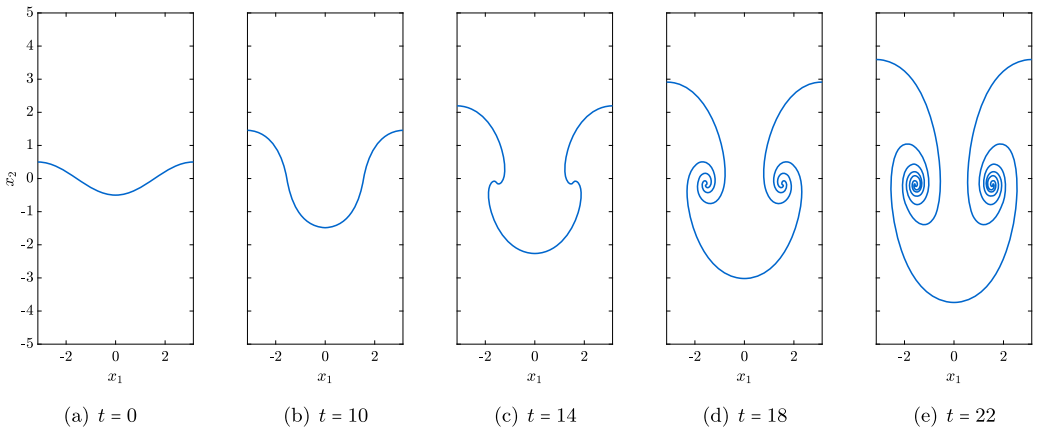


Fig. 11. Numerical simulation of the RTI using the z-model with the setup as in the numerical studies of Sohn [92]. Plots of the interface z for a simulation with Atwood number $A = 0.05$ are shown at various times t.

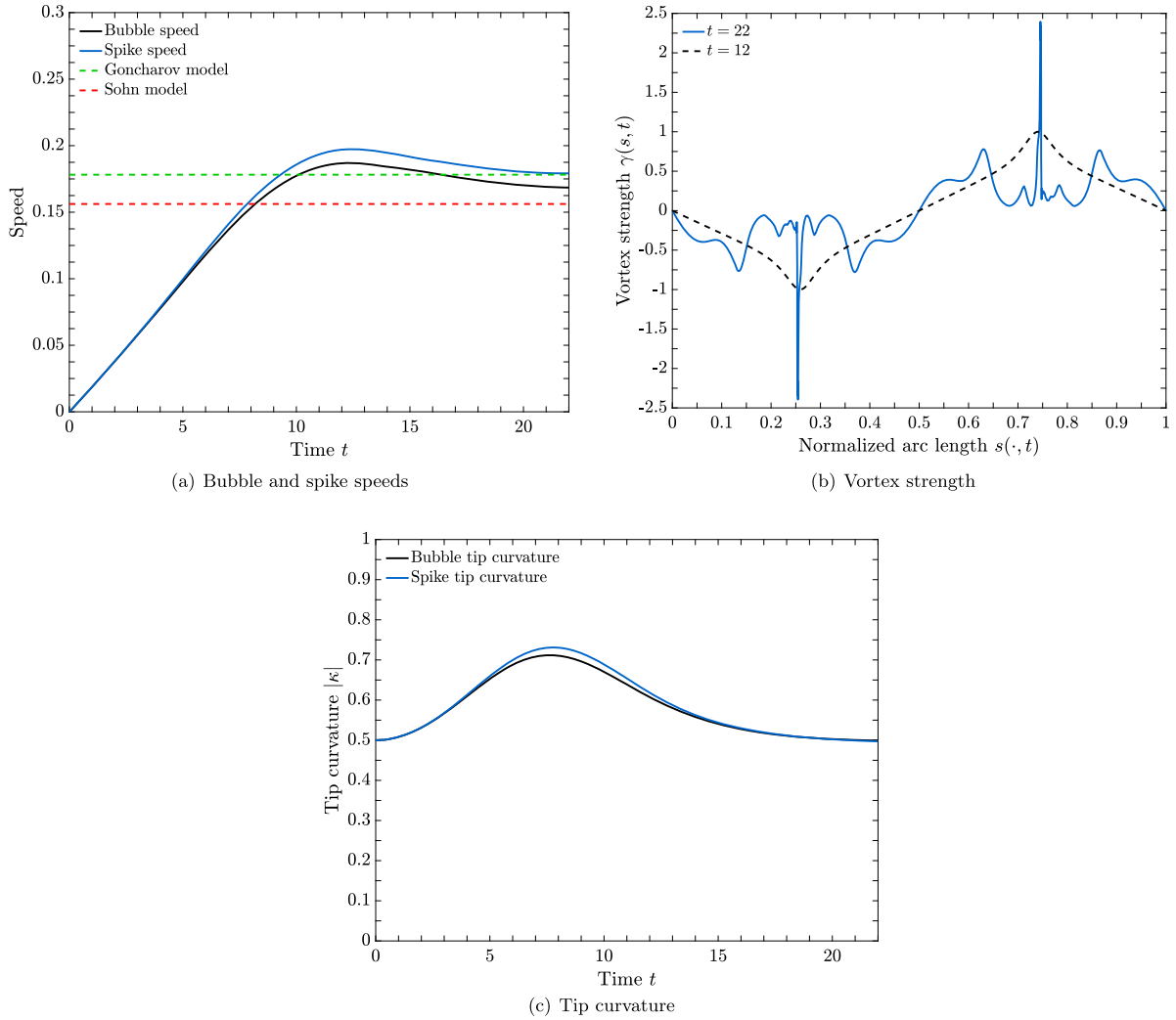


Fig. 12. Plots of various computed quantities for the Sohn [92] RTI problem.

Table 2
Time-step δt choices for the rocket rig test.

	N				
	128	256	512	1024	2048
δt	1×10^{-3}	1×10^{-3}	5×10^{-4}	1.25×10^{-4}	3.125×10^{-5}

respectively. As shown in Fig. 12(a), the computed speed and asymptotic predictions are in good agreement; let us note that the bubble and spike speeds computed using the z-model appear identical to those in [92].

Next, we compute the *vortex sheet strength* $\gamma(\cdot,t) = \varpi(\cdot,t)/|\partial_\alpha z(\cdot,t)|$. We plot $\gamma(s,t)$ versus s in Fig. 12(b), where $s(t)$ is the normalized arclength. Comparing Fig. 12(b) with Fig. 7(a) in [92], we see that $\gamma(s,t)$ computed using the z-model is slightly larger in magnitude at the shock, but is otherwise identical to the solution in [92]. Finally, we compute the magnitude of the bubble tip and spike tip curvatures $|\kappa|$ in Fig. 12(c), which is in excellent agreement with Fig. 6(a) in [92].

The above observations indicate that the z-model is able to accurately simulate interface turnover and roll-up, while minimizing the cost of the numerical computations. In particular, since six iterations are required, on average, for the algorithm in [92], the z-model computation is at least 6 times faster if the same time-step δt used. For the above simulation, we were able to use a much larger time-step, which shows that the z-model computation is a factor of at least $6 \times 0.025/0.002 = 75$ times faster. In fact, since the Fast Fourier Transform (FFT) is used to efficiently compute the ϖ-equation (38), whereas costly upwinding is required for the algorithm in [92], it is highly likely that the z-model computation is much greater than 75 times faster than the algorithm in [92].

4.3.5. Multi-mode RTI: the rocket rig experiment of Read and Youngs

We next consider the rocket rig experiment of Read [79] and Youngs [106], in which the initial interface separating the two fluids of densities ρ^+ and ρ^- is given by a small and random perturbation of the flat interface. Our aim is to compare the growth rates of the mixing layer computed using our z-model with the growth rates observed in experiments [79] and 2-D DNS simulations [106]. The experimental and numerical evidence indicate that the width of the mixing layer grows like

$$\Theta A g t^2 \,, \tag{43}$$

where Θ is some constant. The experimental evidence suggests that the constant Θ lies in the range $\Theta \in (0.05, 0.775)$, while the numerical simulations give $\Theta \in (0.04, 0.05)$. We shall follow Youngs [107], and employ the value $\Theta = 0.06$ for comparison purposes.

We set up the problem as follows: the domain is $\alpha \in [-\pi, +\pi]$, the densities of the two fluids are $\rho^+ = 0.66$ and $\rho^- = 1.89$, giving an Atwood number of $A \approx 0.482$, the gravitational acceleration is $g = -9.8 \times \frac{2\pi}{0.3}$, and the initial data is given by

$$\begin{aligned} z_1(\alpha,0) &= \alpha \,, \\ z_2(\alpha,0) &= \frac{1}{\sigma}\sum_{r=1}^{s} a_r \cos(r\alpha) + b_r \sin(r\alpha) \,, \\ \varpi(\alpha,0) &= 0 \,, \end{aligned}$$

where a_r and b_r are random number chosen from a standard Gaussian distribution, $s = 32$, and σ is a constant chosen such that $||z_2(\cdot,0)||_{L^2} = 0.01$.

We perform 5 simulations using the z-model and with resolution starting from $N = 128$ and doubling until $N = 2048$. The regularization parameters are fixed as $\tilde{\delta} = 0.065$ and $\mu = 0.06$ for all of the simulations. The time-step δt varies with N, and is listed in Table 2. We show plots of the computed interface parametrization $z(\alpha,t)$ at the final time $t = 0.15$ in Fig. 13. As expected, there is more-roll up of the interface as the resolution is increased. To quantify the amount of mixing, the width of the mixing region is approximated as $\max_\alpha z_2(\alpha,t) - \max_\alpha z_2(\alpha,0)$. A comparison of the computed mixing region width and the predicted quadratic growth rate (43) with $\Theta = 0.06$ is shown in Fig. 14, from which it is clear that the two are in very good agreement.

Since the initial conditions for the rocket rig test are random, it may be difficult to obtain a clear qualitative picture of the mixing region from a single simulation alone. Consequently, we repeat the test for $N = 512$ with six different random initial conditions. The interface position for the ensemble of runs is shown in Fig. 15(a), while the mean mixing region width is compared with the quadratic growth rate in Fig. 15(b). It is clear from Fig. 15(b) that the computed mixing region width is in excellent agreement with the predicted quadratic growth rate, thus providing strong evidence for the validity of the z-model. The average runtime for the $N = 512$ simulations is only $T_{\text{CPU}} \approx 20$s; thus, the use of the z-model permits the inference of large-scale qualitative and quantitive information with minimal computational expense.

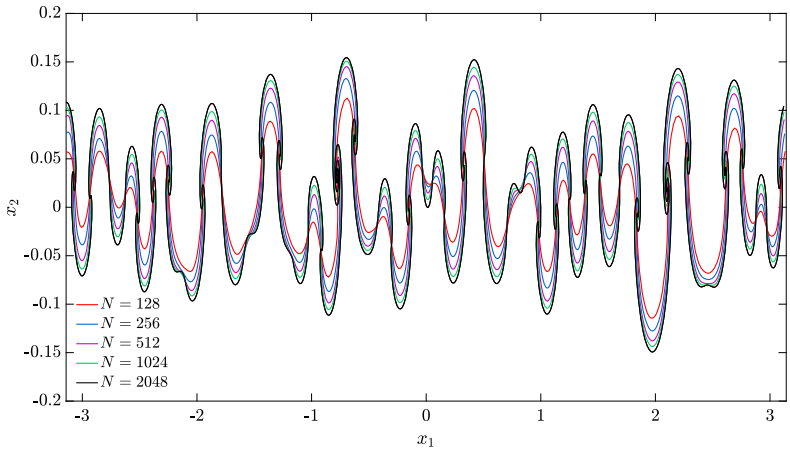


Fig. 13. Numerical simulation of the rocket rig test using the z-model. The results for five simulations with resolution starting from $N = 128$ and doubling until $N = 2048$ are shown at time $t = 0.15$.

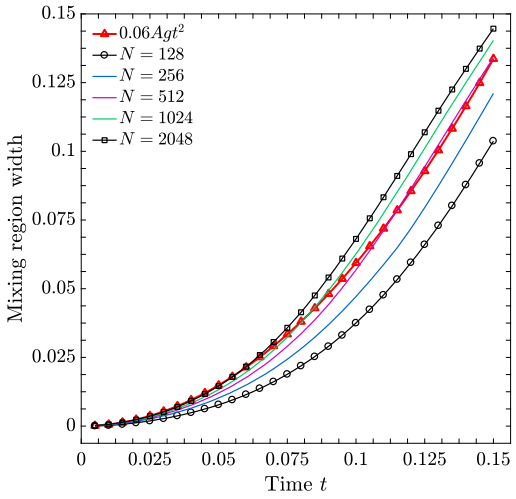
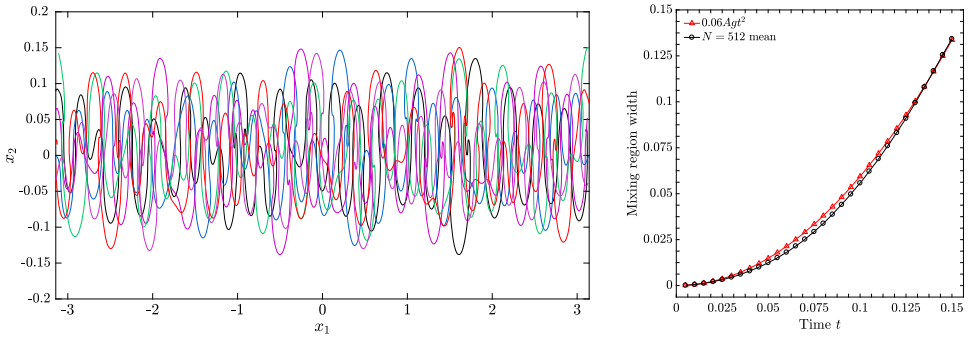


Fig. 14. Plot of the mixing region width $\max z_2(\cdot, t) - \min z_2(\cdot, 0)$ versus time t for the rocket rig test. Results are shown for five simulations using the z-model with increasing resolution N. The red curve is the predicted quadratic growth rate (43) with $\Theta = 0.06$.



(a) Interface position z (b) Mean mixing region width

Fig. 15. Ensemble of simulations for the rocket rig test with $N = 512$. Shown are (a) the interface position z at time $t = 0.15$, and (b) the mean mixing region width versus time.

5. A mutliscale model for interface evolution in compressible flow

We now derive a multiscale interface model, founded upon our incompressible-compressible decomposition of the Euler equations. The discontinuous incompressible velocity w solving (27) (and, in particular, obtained from (24)) can be used to compute small scale structures on the interface $\Gamma(t)$ and the Kelvin-Helmholtz instability (KHI). It is often the case that vortex sheet roll-up, caused by the KHI, occurs at spatial scales which are smaller than the scales along which bulk vorticity is transported and for which sound waves propagate. When this occurs, the continuous velocity v solving (28), which is only forced by bulk compression and vorticity, may be computed at larger spatial scales than the velocity w.

5.1. A multiscale model for the compressible RTI

In order to produce a fast-running model of the compressible RTI, we begin by using the higher-order z-model (24) to generate the velocity w. Our model is generated by coupling the equation for v (28) together with

$$\partial_t z(\alpha,t) = \mathrm{P}\int_{\mathbb{T}_L} \mathcal{K}_{\mathbb{T}_L}(z(\alpha,t)-z(\beta,t))\varpi(\beta,t)\,\mathrm{d}\beta + v(z(\alpha,t),t)\,, \tag{45a}$$

$$\partial_t \varpi(\alpha,t) = -\partial_\alpha\left[\frac{A}{2|\partial_\alpha z(\alpha,t)|^2}\mathcal{H}\left(\varpi(\alpha,t)\mathcal{H}\varpi(\alpha,t)\right) - 2Agz_2(\alpha,t)\right], \tag{45b}$$

and computing w using (19). As we explain below, the compressible equations (28) will be solved on a coarse grid, while (45) will be solved on a fine, but one-dimensional, grid.

5.2. A multiscale model for the compressible RMI

5.2.1. Vorticity production

For flows in which shock waves collide with contact discontinuities and initiate the RMI, we shall derive a modified form of the z-model which accounts for the vorticity production that is caused by the misalignment of the pressure gradient at the shock wave and the density gradient at the interface.

Computing the curl of (15b), we obtain the two-dimensional vorticity equation for compressible flow as

$$\frac{\mathrm{D}\omega}{\mathrm{D}t} + \omega\,\mathrm{div}\,u = -\frac{\nabla\rho\cdot\nabla^\perp p}{\rho^2}\,. \tag{46}$$

The term on the right-hand side of (46) is the *baroclinic* term, responsible for vorticity production on the interface when a shock-wave collides with a vortex sheet. The amplitude of vorticity ϖ is the weak (or distributional) form of the vorticity, and consequently it is important to include a weak form of the baroclinic term in the dynamics of ϖ. We thus introduce the following modification of the z-model

$$\partial_t z(\alpha,t) = \mathrm{P}\int_{\mathbb{T}_L} \mathcal{K}_{\mathbb{T}_L}(z(\alpha,t)-z(\beta,t))\varpi(\beta,t)\,\mathrm{d}\beta + v(z(\alpha,t),t)\,, \tag{47a}$$

$$\begin{aligned}\partial_t \varpi(\alpha,t) = &-\partial_\alpha\left[\frac{A}{2|\partial_\alpha z(\alpha,t)|^2}\mathcal{H}\left(\varpi(\alpha,t)\mathcal{H}\varpi(\alpha,t)\right) - 2Agz_2(\alpha,t)\right]\\ &- [\![\nabla p\cdot\partial_\alpha z(\alpha,t)/\rho]\!] + \mu\partial_\alpha^2\varpi(\alpha,t)\,,\end{aligned} \tag{47b}$$

which will be used for flows in which shocks collide with contact discontinuities.

Remark 1. Let us mention that for Richtmyer-Meshkov problems, at the time at which the planar shock collides with the perturbed contact, the pressure is discontinuous along points in the intersection of the shock and contact. As such the numerator in the baroclinic term, $[\![\nabla p\cdot\partial_\alpha z(\alpha,t)]\!]$ can be large at such points. Such a pressure profile does not occur in the Rayleigh-Taylor problems, for which the numerator $[\![\nabla p\cdot\partial_\alpha z(\alpha,t)]\!]$ vanishes along the contact. Thus, the use of this baroclinic term in the ϖ-equation is imperative for the simulation of the RMI problems.

5.2.2. Taylor's frozen turbulence hypothesis

Taylor's "frozen turbulence" hypothesis [95], roughly speaking, states that if the mean flow velocity is much larger than the velocity of the turbulent eddies, then the advection of the turbulent flow past a fixed point can be taken to be due entirely to the mean flow, or in other words, that the turbulent fluctuations are transported by the mean flow. For shock-contact collisions that initiate the RMI, the velocity v at the shock front is much larger in magnitude than the velocity w. That is to say, the relation $\max_x|v| \gg \max_x|w|$ holds true. For instance, for the RMI problem considered in Section 6.4, the quantity $\max_x|w|$ is two orders of magnitude smaller than the quantity $\max_x|v|$. For such flows, we view the velocity v

as the mean velocity and w as the fluctuation velocity, and impose the Taylor hypothesis that for very short time-intervals (i.e., about one time-step in an explicit numerical simulation), w is transported by v, so that

$$\partial_t w + (v \cdot \nabla) w = 0 \,. \tag{48}$$

As we described in Remark 1, at points along the interface at which the shock front intersects the contact discontinuity, there exists a large increase in the baroclinic term, which in turn, produces a large increase in the amplitude of vorticity and this then leads to a localized increase in the small-scale velocity field w via equation (47b) at each such intersection point at each time-step. As the shock passes through the contact discontinuity these points of intersection evolve, and this evolution causes large (and localized) space gradients ∇w and temporal gradients $\partial_t w$. The Taylor hypothesis (48) ensures that a proper balance is retained between the space and time gradient of w in a numerical implementation which approximates these two different types of derivatives in a very different manner.

Using (48) together with (28a), we find that

$$\partial_t(\rho w) + \operatorname{div}(\rho w \otimes v) + \operatorname{div}(\rho w \otimes w) = \rho (w \cdot \nabla) w \,,$$

and hence (28b) must be replaced by

$$\partial_t(\rho v) + \nabla \cdot (\rho v \otimes v) + \nabla p + \rho g e_2 = -\nabla \cdot (\rho v \otimes w) - \rho (w \cdot \nabla) w \,. \tag{28b'}$$

Therefore, our model for the RMI problem couples (47) with (28), but with (28b') replacing (28b). Using a Richtmyer-Meshkov test problem, we explain in detail in Section 6.4 the reasons for using the Taylor hypothesis and (28b') in our multiscale algorithm for RMI flows.

5.3. The multiscale algorithms for the RTI and RMI problems

We are now ready to give a precise description of the multiscale algorithm. Denote by $\mathbf{V}(x,t) = (\rho, \rho v_1, \rho v_2, E)^{\mathcal{T}}$, the solutions to the compressible equations (28). We shall use a standard 3rd order Runge-Kutta procedure for time-integration; in the following algorithms, we use the superscript notation to denote the Runge-Kutta stage.

We shall use two slightly different algorithms; the first is for our multiscale model for RTI problems, while the second is for the multiscale model for RMI problems.

We note that the ordering of Steps **4(b)**, **4(c)**, and **4(d)** in the RTI multiscale algorithm is important; our numerical experiments have shown that it is important that the velocity $w'(x)$ is computed *before* the auxiliary interface $\tilde{z}$ is updated.

For Richtmyer-Meshkov problems, we use a slightly different algorithm. In particular, we are no longer required to compute the time-derivative $\partial_t(\rho w)$ and so shall omit those steps from the algorithm. Notice that this means that we can omit in particular Step **4(b)** of the RTI algorithm, which removes at each time-step one of the costly integral computations. On the other hand, we must compute in Step **4(d)** of the RMI algorithm the baroclinic term in the ϖ-equation (47).

Remark 2. In this work, we apply our multiscale algorithm to 2-D flows that are symmetric across the line $x_1 = 0$. In particular, this means that the velocities w_1 and v_1 are odd functions of x_1, and the vorticity amplitude ϖ is an odd function of α.

RTI MULTISCALE ALGORITHM.

Step 0 Suppose that we are given the solution $z(\alpha,t)$, $\varpi(\alpha,t)$, and $\mathbf{V}(x,t)$ at time-step t, as well as a velocity $w'(x)$, and we wish to compute the solution at $t+\delta t$. Define $z^0(\alpha) := z(\alpha,t)$, $\varpi^0(\alpha) := \varpi(\alpha,t)$, and $\mathbf{V}^0(x) := \mathbf{V}(x,t)$, and let $\rho'(x) := \rho^0(x) = \rho(x,t)$.

Step 1

- **1(a)** Compute a velocity $\tilde{w}(x)$ from $z^0(\alpha)$ and $\varpi^0(\alpha)$ using the Biot-Savart law.
- **1(b)** Approximate $\partial_t(\rho\tilde{w})(x) = \left[\rho^0(x)\tilde{w}(x) - \rho'(x)w'(x)\right]/\delta t$.
- **1(c)** Solve the compressible equations (28) (with $w = \tilde{w}$) to obtain $\mathbf{V}^1(x)$.
- **1(d)** Compute an auxiliary $\tilde{z}(\alpha)$ and $\tilde{\varpi}(\alpha)$ by solving the system (45).
- **1(e)** Calculate the interfacial velocity $v(\tilde{z}(\alpha)) = v^1(x)|_{\tilde{z}(\alpha)}$ using interpolation.
- **1(f)** Update $z^1(\alpha) = \tilde{z}(\alpha) + \delta t \cdot v(\tilde{z}(\alpha))$, and set $\varpi^1(\alpha) = \tilde{\varpi}(\alpha)$.

Step 2 Repeat **Step 1** but with quantities evaluated at the next Runge-Kutta stage.
Step 3 Repeat **Step 2** but with quantities evaluated at the next Runge-Kutta stage.
Step 4

- **4(a)** Use the standard 3rd order Runge-Kutta formula to produce $\mathbf{V}(x,t+\delta t)$, and an auxiliary interface $\tilde{z}(\alpha)$ and vorticity amplitude $\tilde{\varpi}(\alpha)$.
- **4(b)** Compute a velocity $w'(x)$ from $\tilde{z}(\alpha)$ and $\tilde{\varpi}(\alpha)$ using the Biot-Savart law.
- **4(c)** Calculate the interfacial velocity $v(\tilde{z}(\alpha)) = v(x,t+\delta t)|_{\tilde{z}(\alpha)}$ using interpolation.
- **4(d)** Update $z(\alpha,t+\delta t) = \tilde{z}(\alpha) + \delta t \cdot v(\tilde{z}(\alpha))$, and set $\varpi(\alpha,t+\delta t) = \tilde{\varpi}(\alpha)$, then return to **Step 0**.

RMI MULTISCALE ALGORITHM.

Step 0 Suppose that we are given the solution $z(\alpha,t)$, $\varpi(\alpha,t)$, and $\mathbf{V}(x,t)$ at time-step t, as well as the baroclinic term $\phi(\alpha,t)=[\![\nabla p\cdot\partial_\alpha z/\rho]\!]$ and we wish to compute the solution at $t+\delta t$. Define $z^0(\alpha):=z(\alpha,t)$, $\varpi^0(\alpha):=\varpi(\alpha,t)$, and $\mathbf{V}^0(x):=\mathbf{V}(x,t)$.

Step 1

1(a) Compute a velocity $\tilde{w}(x)$ from $z^0(\alpha)$ and $\varpi^0(\alpha)$ using the Biot-Savart law.
1(b) Solve the modified compressible equations (28) (with $w=\tilde{w}$), using (28b') in place of (28b), to obtain $\mathbf{V}^1(x)$.
1(c) Compute an auxiliary $\tilde{z}(\alpha)$ and $\varpi(\alpha)$ by solving the system (47).
1(d) Calculate an auxiliary baroclinic term $\tilde{\phi}(\alpha)$ using $\tilde{z}(\alpha)$, $p^1(x)$, and $\rho^1(x)$.
1(e) Calculate the interfacial velocity $v(\tilde{z}(\alpha))=v^1(x)|_{\tilde{z}(\alpha)}$ using interpolation.
1(f) Update $z^1(\alpha)=\tilde{z}(\alpha)+\delta t\cdot v(\tilde{z}(\alpha))$, and set $\varpi^1(\alpha)=\tilde{\varpi}(\alpha)$.

Step 2 Repeat **Step 1** but with quantities evaluated at the next Runge-Kutta stage.
Step 3 Repeat **Step 2** but with quantities evaluated at the next Runge-Kutta stage.
Step 4

4(a) Use the standard 3rd order Runge-Kutta formula to produce $\mathbf{V}(x,t+\delta t)$, and an auxiliary interface $\tilde{z}(\alpha)$ and vorticity amplitude $\varpi(\alpha)$.
4(b) Calculate the interfacial velocity $v(\tilde{z}(\alpha))=v(x,t+\delta t)|_{\tilde{z}(\alpha)}$ using interpolation.
4(c) Update $z(\alpha,t+\delta t)=\tilde{z}(\alpha)+\delta t\cdot v(\tilde{z}(\alpha))$, and set $\varpi(\alpha,t+\delta t)=\tilde{\varpi}(\alpha)$.
4(d) Compute the baroclinic term $\phi(\alpha,t+\delta t)$ using $z(\alpha,t+\delta t)$, $p(x,t+\delta t)$, and $\rho(x,t+\delta t)$, then return to **Step 0**.

5.4. Numerical implementation of the multiscale algorithm

We now describe how we numerically implement the multiscale algorithm described in Section 5. Suppose that the conservative variables $\mathbf{V}$ are computed in the bounded domain $\Omega=\mathbb{T}_L\times[x_2^1,x_2^n]$, and that the flow is periodic in the horizontal variable x_1. Suppose also that a single wavelength of the periodic interface $\Gamma(t)$ is parametrized by the function $z(\alpha,t)$.

Discretize the domain Ω with $(2m-1)\times n$ cells with cell centers at

$$x_1^i=-L/2+(i-1)\delta x_1\,,$$
$$x_2^j=x_2^1+(j-1)\delta x_2\,,$$

with $\delta x_1=L/(2m-2)$ and $\delta x_2=(x_2^n-x_2^1)/(n-1)$, and suppose that the parameter α is discretized with $N=2^r+1$ nodes,

$$\alpha_k=-L/2+(k-1)\triangle\alpha\,,$$

with $\triangle\alpha=L/(N-1)$. We spatially discretize the equations of motion, then use a standard third-order explicit Runge-Kutta solver for time integration. We shall use a space-time smooth artificial viscosity method, which we call the C-method [78], for the compressible v-equations (28) to stabilize shock fronts and contact discontinuities, and thereby prevent the onset of Gibbs oscillations.

It remains to describe the following: first, the numerical implementation of the space-time smooth artificial viscosity C-method; second, the computation of the velocity w on the plane; third, the bilinear interpolation scheme; and finally, the calculation of the weak baroclinic term $[\![\nabla p\cdot\partial_\alpha z/\rho]\!]$.

5.4.1. Numerical implementation of the C-method

We implement a simple finite difference WENO-based scheme to spatially discretize the system (28). Our simplified WENO scheme is devoid of any exact or approximate Riemann solvers, and instead relies on the sign of the velocity to perform upwinding. A fourth-order central difference approximation is used to compute the pressure gradient ∇p, while a second-order central difference approximation is employed to compute the diffusion terms (60). For brevity, we omit further details of the numerical implementation of the C-method and refer the reader to Appendix B and [78] for further details.

5.4.2. Computing the w velocity

We first describe our method for calculating the discrete velocity $w_{i,j}=w(x_1^i,x_2^j)$ from a given discretized interface parametrization z_k and vorticity amplitude ϖ_k.

Since integral-kernel calculations can be computationally very expensive, we begin by proposing a simplification to speed up such calculations. Suppose first that we wish to compute the velocity w at a point $x_{i,j}$ such that $|x_2^j-z_2^k|\gg 1$ for every k. Then the following approximations are valid:

$$-\frac{\sinh(2\pi(x_2^j-z_2^k)/L)}{\cosh(2\pi(x_2^j-z_2^k)/L)-\cos(2\pi(x_1^i-z_1^k)/L)}\approx\pm 1\,,$$
$$\frac{\sin(2\pi(x_1^i-z_1^k)/L)}{\cosh(2\pi(x_2^j-z_2^k)/L)-\cos(2\pi(x_1^i-z_1^k)/L)}\approx 0\,.$$

Then, using the fact that ϖ is an odd function (cf. Remark 2), it follows that $w_{i,j} \approx 0$. Consequently, it is sufficient to compute w only for those $x_{i,j}$ that lie in the horizontal strip

$$\Omega_z = \left\{ x \in \mathbb{T}_L \times [x_2^1, x_2^n] \text{ s.t. } \min_k z_2^k - \lambda \le x_2 \le \max_k z_2^k + \lambda \right\}.$$

For the simulations considered here, we set $\lambda = 0.075$, but note that this parameter is problem dependent. Computing the velocity only in the strip Ω_z speeds up an otherwise time-consuming calculation.

We now define the scalar function $\mathcal{J}^\delta(x)$ by

$$\mathcal{J}^\delta(x) = \frac{1}{4\pi} \log \left\{ \delta^2 + \cosh(2\pi x_2/L) - \cos(2\pi x_1/L) \right\}.$$

The function $\mathcal{J}^\delta$ is a smoothed version of the singular kernel used in the integral representation of Δ^{-1}, so that $\mathcal{K}^\delta_{\mathbb{T}_L}(x) = \nabla^\perp \mathcal{J}^\delta(x)$.

The velocity w is determined by first calculating the *stream function* $\psi(x,t)$ using the formula

$$\psi(x,t) = \int_{\mathbb{T}_L} \mathcal{J}^\delta(x - z(\beta,t)) \varpi(\beta,t)\, \mathrm{d}\beta,$$

then using the relation $w = \nabla^\perp \psi$. Since integral-kernel calculations are computationally expensive, it is advantageous to perform a single such computation then take derivatives, rather than perform two such computations. Moreover, the function $\mathcal{J}^\delta(x)$ is (roughly speaking) one derivative smoother than the kernel $\mathcal{K}^\delta_{\mathbb{T}_L}(x)$, so that we may hope that the integral calculation is numerically more stable. Using the stream-function formulation also guarantees that the velocity field w produced is divergence free, whereas a direct singular integral calculation of w can produce inaccuracies so that the resultant velocity field does not satisfy $\nabla \cdot w = 0$.

For $x_{i,j} \in \Omega_z$, we can compute the stream function $\psi_{i,j}$ using the same trapezoidal method as described in Section 4. We define $\psi_{i,j} = 0$ for $x_{i,j} \notin \Omega_z$. We then use a standard second-order central difference approximation to determine the velocity $w_{i,j}$ from $\psi_{i,j}$.

5.4.3. Bilinear interpolation scheme

We shall employ a simple bilinear interpolation scheme as follows. Let z_k be the discretized interface parametrization, and suppose that we are given a scalar function $f_{i,j}$ defined at the cell centers $x_{i,j} \in \Omega$. We wish to determine an approximation f^k to the value of $f_{i,j}$ at the points α_k. For fixed k, we determine for which i and j the point z_k lies in the rectangle $[x_1^i, x_1^{i+1}] \times [x_2^j, x_2^{j+1}]$ by requiring that $(z_k^1 - x_1^i)(z_k^1 - x_1^{i+1}) \le 0$ and $(z_k^2 - x_2^j)(z_k^2 - x_2^{j+1}) \le 0$. The interpolated quantity f_k is then defined as

$$\begin{aligned} f^k = {} & f_{i,j} \left(1 - \frac{(z_1^k - x_1^i)}{\delta x_1}\right)\left(1 - \frac{(z_2^k - x_2^j)}{\delta x_2}\right) \\ & + f_{i+1,j} \left(1 - \frac{(x_1^{i+1} - z_1^k)}{\delta x_1}\right)\left(1 - \frac{(z_2^k - x_2^j)}{\delta x_2}\right) \\ & + f_{i,j+1} \left(1 - \frac{(z_1^k - x_1^i)}{\delta x_1}\right)\left(1 - \frac{(x_2^{j+1} - z_2^k)}{\delta x_2}\right) \\ & + f_{i+1,j+1} \left(1 - \frac{(x_1^{i+1} - z_1^k)}{\delta x_1}\right)\left(1 - \frac{(x_2^{j+1} - z_2^k)}{\delta x_2}\right). \end{aligned}$$

5.4.4. Calculation of the weak baroclinic term

Next, we discuss the calculation of the weak baroclinic term $[\![\nabla p \cdot \partial_\alpha z/\rho]\!]$ on the interface $\Gamma(t)$. Suppose that we are given the pressure $p_{i,j}$ and density $\rho_{i,j}$ defined on the plane, and the discretized interface parametrization z_k.

We begin by computing the unit normal n to the interface as

$$n = \frac{\partial_\alpha z^\perp}{|\partial_\alpha z^\perp|} = \frac{(-\partial_\alpha z_2\,, \partial_\alpha z_1)}{|(-\partial_\alpha z_2\,, \partial_\alpha z_1)|}.$$

The jump across the interface in a quantity f defined at grid points is approximated as

$$[\![f]\!] \approx -\delta n \cdot \nabla f \approx -\frac{|\delta x|}{2}\, n \cdot \nabla f|_{z(\alpha,t)}, \tag{49}$$

where $\nabla f|_{z(\alpha,t)}$ denotes the evaluation of the quantity $\nabla f(x_1, x_2)$ at the interface parametrization $z(\alpha, t)$; this is accomplished using the bilinear interpolation scheme described above. All derivatives are approximated using second-order accurate central difference approximations.

Thus, to evaluate the baroclinic term $[\![\nabla p \cdot \partial_\alpha z/\rho]\!]$, we compute $\nabla(\partial_i p/\rho)$ for $i = 1, 2$ on the fixed Eulerian grid, interpolate onto the interface, and use formula (49). More explicitly,

$$[\![\nabla p/\rho]\!] \cdot \partial_\alpha z \approx -\frac{|\delta x|}{2|\partial_\alpha z|} \partial_\alpha z^\perp \mathbf{M} \partial_\alpha z \,, \tag{50}$$

where $\mathbf{M}$ is the 2×2 matrix defined as

$$\mathbf{M} = \begin{bmatrix} \partial_1(\partial_1 p/\rho) & \partial_1(\partial_2 p/\rho) \\ \partial_2(\partial_1 p/\rho) & \partial_2(\partial_2 p/\rho) \end{bmatrix} \Big|_{z(\alpha,t)} ,$$

and $|_{z(\alpha,t)}$ once again denotes evaluation at the interface parametrization (using bilinear interpolation).

6. Numerical simulations of the RTI and RMI using the multiscale model

We next present results from four numerical experiments to demonstrate the efficacy of our multiscale model and its numerical implementation. The objective of this section is (1) to show that the multiscale model produces solutions with accurate interface motion that correctly captures the KH structures during RTI and RMI, and (2) to demonstrate that the multiscale algorithm is roughly two orders of magnitude times faster to run that a standard gas dynamics simulation.

6.1. The compressible RTI test of Almgren et al.

We first consider the compressible single-mode RTI test from the paper of Almgren et al. [1]. The domain is $(x_1, x_2) \in [-0.25, 0.25] \times [0, 1]$ and the gravitational constant is $g = 1$. Periodic conditions are applied in the horizontal x_1 direction and free-flow conditions are imposed at the boundaries in the x_2 direction. In particular, the pressure is extended linearly to satisfy the hydrostatic assumption at the top and bottom boundaries. The initial data is given as follows: the initial velocity is identically zero $u_0 = 0$, and the pressure p_0 is defined as

$$p_0 = \begin{cases} 5 - \rho^- g x_2 & , \text{ if } x_2 < 0.5 \\ 5 - 0.5\rho^- g - \rho^+ g(x_2 - 0.5) & , \text{ if } x_2 \geq 0.5 \end{cases} , \tag{51}$$

where $\rho^+ = 2$ and $\rho^- = 1$. The initial density ρ_0 is defined as

$$\rho_0(x_1, x_2) = \rho^- + \frac{\rho^+ - \rho^-}{2} \left[1 + \tanh\left(\frac{x_2 - \eta_0(x_1)}{h} \right) \right] , \tag{52}$$

with $\eta_0(x_1) = 0.5 - 0.01\cos(4\pi x_1)$. The tanh profile introduces a small length scale h over which the initial density is smeared.

We begin by computing a *benchmark* or *high-resolution reference* solution using the anisotropic C-method[4] on a fine mesh consisting of 128×512 cells, a CFL number of CFL ≈ 0.4, and $h = 0.005$ as in [1].

The computed density is shown in Fig. 16 at the final time $t = 2.5$. In [1], the authors compare solutions computed using the piecewise-parabolic method (PPM) and the piecewise-linear method. It is shown that the (dimensionally) operator-split versions of the methods result in spurious secondary instabilities, whereas the unsplit versions of the methods suppress these instabilities while keeping a sharp interface and well-defined roll-up regions (see Fig. 9 in [1]). While the C-method is dimensionally split, the use of the anisotropic artificial viscosity [78] prevents the onset of the spurious secondary instabilities, while ensuring that the KHI roll-up region and mixing zones are not smeared.[5]

6.1.1. The multiscale algorithm applied to the RTI

We next employ the RTI multiscale algorithm described in Section 5.3 with the following parameter values: the underlying coarse mesh used to solve for v contains 8×32 cells, while the interface is discretized with a fine mesh consisting of $N = 128$ nodes; the time-step is $\delta t = 2.5 \times 10^{-3}$, giving CFL ≈ 0.4. The Atwood number is $A = 1/3$, the initial velocity is $v_0 = 0$, the initial pressure is given by (51), the initial density is given by (52) with $h = 0.02$, and

[4] This is a spacetime smooth artificial viscosity method employed with a highly simplified WENO discretization of the compressible Euler equations (see [80,77,78]).

[5] We note that both the CPU time and memory usage are approximately a factor of 2 larger for unsplit methods than for split methods [1]. The C-method uses a highly simplified dimensionally-split WENO-type scheme and is thus relatively fast.

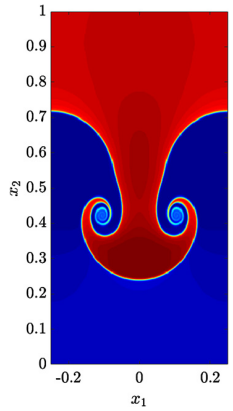
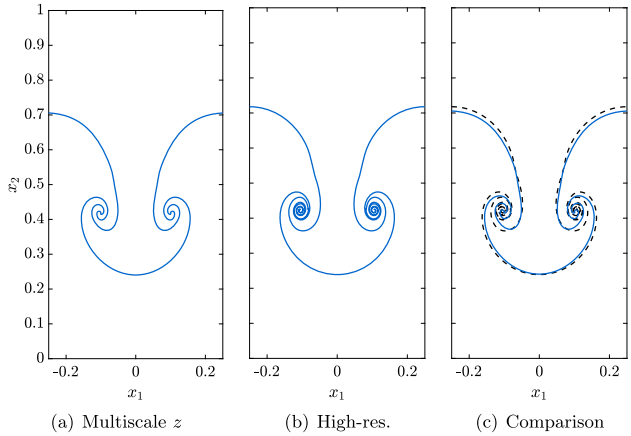


Fig. 16. The density profile at time $t = 2.5$ for the RTI test of Almgren et al. [1]. The solution is computed using the anisotropic C-method on a mesh with 128×512 cells.



(a) Multiscale z (b) High-res. (c) Comparison

Fig. 17. Results for the multiscale algorithm applied to the compressible single-mode RTI test of Almgren et al. [1], with the underlying grid containing 8×32 cells and the interface discretized with $N = 128$. Shown are (a) computed interface parametrization z, (b) benchmark interface position, and (c) the computed interface z (blue) overlaying the benchmark solution (dashed black) at the final time $t = 2.5$.

$$z_1(\alpha, 0) = \alpha\,,$$
$$z_2(\alpha, 0) = 0.5 - 0.01\cos(4\pi\alpha)\,,$$
$$\varpi(\alpha, 0) = 0\,.$$

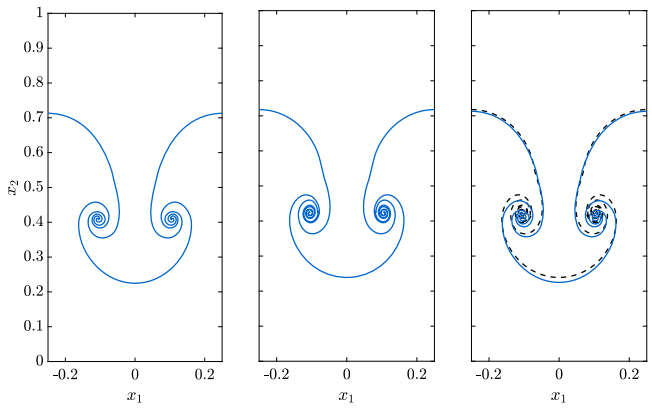
The artificial viscosity parameters are chosen as $\beta = 50$, $\tilde{\delta} = 1.0$, and $\mu = 1.5 \times 10^{-3}$.

We provide plots of the resulting solutions using our multiscale algorithm. The interface position z, at the final time $t = 2.5$, is shown in Fig. 17(a), and the high-resolution reference solution is shown in Fig. 17(b). In Fig. 17(c), we compare the interface positions computed using the multiscale algorithm and the high-resolution run; we see that the two solutions are in excellent agreement, with nearly identical spike tip and bubble tip positions; moreover, the multiscale algorithm successfully simulates the roll up of the vortex sheet. The reference solution computation had a runtime of $T_{\text{CPU}} \approx 5015$ s, whereas the multiscale algorithm runtime was only $T_{\text{CPU}} \approx 7$ s, giving a speed-up of approximately 683 times.

Increasing the resolution of both the coarse grid for v and the fine grid for w results in a solution with even more roll-up and accuracy. We show in Fig. 18(a) the multiscale solution computed using 16×64 cells for the coarse mesh and $N = 256$ nodes for the fine mesh. The time-step is $\delta t \approx 8.333 \times 10^{-4}$, giving CFL ≈ 0.27, and the artificial viscosity parameters are unchanged from the previous run. We observe that the computed interface shows more roll up, and is in even better agreement with the reference solution. The runtime of this simulation was $T_{\text{CPU}} \approx 105$ s, resulting in a speed-up factor of approximately 48.

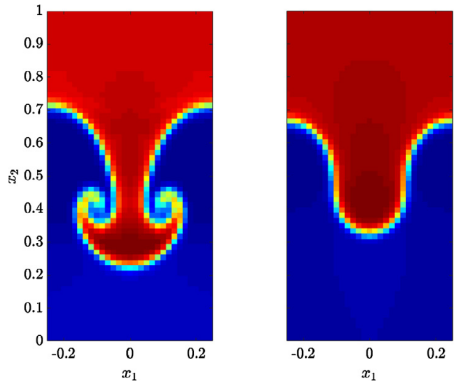
6.1.2. Comparison of the low resolution density with the multiscale density

The multiscale algorithm solves for the velocity v on a coarse mesh, using fine-scale information from the velocity w. In order to show how small-scale information improves the coarse-mesh simulation, we shall compare the solution obtained



(a) Multiscale z (b) High-res. (c) Comparison

Fig. 18. Results for the multiscale algorithm applied to the compressible single-mode RTI test of Almgren et al. [1], with the underlying grid containing 16×64 cells and the interface discretized with $N = 256$. Shown are (a) computed interface parametrization z, (b) benchmark interface position, and (c) the computed interface z (blue) overlaying the benchmark solution (dashed black) at the final time $t = 2.5$.



(a) Multiscale ρ (b) Low-res. ρ

Fig. 19. Comparison of the low resolution density with the multiscale density for the RTI test of Almgren et al. [1]. Fig. 19(a) is a plot of the density computed using the multiscale algorithm, and Fig. 19(b) is a plot of the low resolution Euler density, computed using the C-method on a coarse grid. Both solutions are computed using identical parameters.

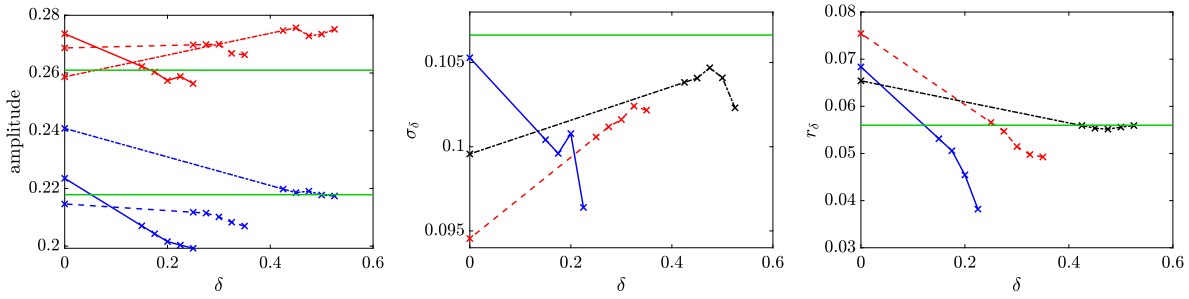
with our multiscale algorithm with the solution obtained by solving the Euler equations (using the same gas dynamics code as used for the high-resolution reference solution) on the same coarse meshes used to solve for v in the multiscale algorithm. We shall refer to this gas dynamics simulation as the *low resolution solution*.

To be more precise, we obtain the low resolution density profile (using the C-method) on a grid with 16×64 cells and with a time-step of $\delta t \approx 8.333 \times 10^{-4}$. In Fig. 19, this solution is compared against the multiscale density function obtained from the simulation shown in Fig. 18. It is clear that the low resolution density profile does not have any of the basic KHI structure of the actual solution, and is very different from the multiscale density which (although solved for on the same coarse mesh) shows the KH roll-up structure. The multiscale algorithm allows for the recovery of small-scale information on the coarse grid via the computation of the velocity w using a fine mesh for the interface. This small-scale information subsequently results in more structure in the roll-up region, and, therefore, a solution that is qualitatively more similar to high resolution simulations.

6.1.3. Comparison with other schemes and convergence studies

We next compare our multiscale algorithm with a modified version of the multiscale algorithm, in which the incompressible, irrotational velocity w is computed using the kernels (34), rather than the Krasny kernel (30). As in Section 4.3, we are interested in the convergence properties of (1) the bubble and spike tip locations, and (2) the radius r_δ and location σ_δ of the spiral roll up region. We choose the axis $x_2 = x_2^* = 0.4125$ to compute the intersection points x^*.

We first take $N = 128$ fixed and consider the limit $\delta \to 0$. The results are shown in Fig. 20. We observe that all three methods are in reasonable agreement with regards to the computed bubble and spike tip locations. The 3rd order kernel is in better agreement with the exact solution for the bubble position, whereas the Krasny kernel is in better agreement with the exact solution for the spike position. All three methods produce similar spiral radii r_δ, and the computed values are in

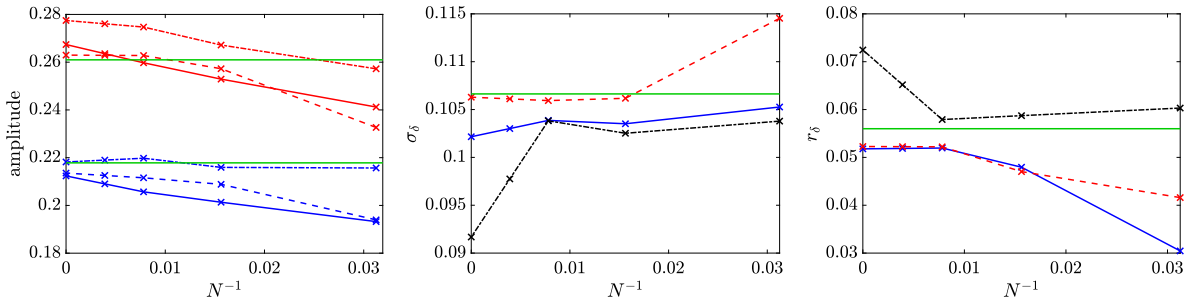


(a) Bubble (blue) and spike (red) tip locations.

(b) Location of spiral center σ_δ.

(c) Radius of spiral r_δ.

Fig. 20. Convergence behavior as $\delta \to 0$ with $N = 128$ fixed for the compressible RTI test of Almgren et al. [1] using the z-model. Shown are (a) the bubble and spike tip locations, $|\max z_2 - 0.5|$ and $|\min z_2 - 0.5|$, respectively, (b) the location of the center of the spiral region σ_δ, and (c) the radius of the spiral region r_δ. The solid, dashed, and dotted curves in (a) refer to the Krasny, $\mathcal{K}_1^\delta$, and $\mathcal{K}_3^\delta$ kernels, respectively. The blue, red, and black curves in (b) and (c) refer to the Krasny, $\mathcal{K}_1^\delta$, and $\mathcal{K}_3^\delta$ kernels, respectively. The green curves indicate the corresponding quantities for the exact solution.



(a) Bubble (blue) and spike (red) tip locations.

(b) Location of spiral center σ_δ.

(c) Radius of spiral r_δ.

Fig. 21. Convergence behavior as $\delta, N^{-1} \to 0$ for the compressible RTI test of Almgren et al. [1] using the z-model. Shown are (a) the bubble and spike tip locations, $|\max z_2 - 0.5|$ and $|\min z_2 - 0.5|$, respectively, (b) the location of the center of the spiral region σ_δ, and (c) the radius of the spiral region r_δ. The solid, dashed, and dotted curves in (a) refer to the Krasny, $\mathcal{K}_1^\delta$, and $\mathcal{K}_3^\delta$ kernels, respectively. The blue, red, and black curves in (b) and (c) refer to the Krasny, $\mathcal{K}_1^\delta$, and $\mathcal{K}_3^\delta$ kernels, respectively. The green curves indicate the corresponding quantities for the exact solution.

good agreement with the exact solution for $\delta > 0$. The 3rd order kernel, in particular, is in excellent agreement with the exact solution. The spiral center locations are similarly reasonably accurate for $\delta > 0$.

Next, we consider the limit $\delta, N^{-1} \to 0$. In particular, we consider $N = 32, 64, 128, 256$. We use the scaling (31) for the Krasny scheme, and the empirical procedure described at the beginning of Section 4.3 for the $\mathcal{K}_i^\delta$ kernel schemes. The results are shown in Fig. 21. All of the schemes perform similarly, with the Krasny and $\mathcal{K}_1^\delta$ schemes in better agreement with the exact solution with regards to spike tip position, spiral center location σ_δ, and spiral radius r_δ, while the 3rd order kernel more accurately predicts the bubble tip location. We note that the convergence behavior of the 3rd order kernel scheme with regards to the quantities σ_δ and r_δ is rather erratic; this is a result of the sensitivity of the scheme to the choice of parameter δ. The asymptotic behavior of σ_δ and r_δ in the Krasny scheme with the scaling (31), on the other hand, is more uniform.

6.2. The compressible RTI test of Liska & Wendroff

For our next numerical experiment, we consider the compressible single-mode RTI from the review paper of Liska and Wendroff [62]. The domain is $(x_1, x_2) \in [-1/6, +1/6] \times [0, 1]$, the gravitational constant is $g = 0.1$, and the initial data is

$$\begin{bmatrix} \rho_0 \\ (\rho u)_0 \\ E_0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ p_0/(\gamma - 1) \end{bmatrix} \mathbb{1}_{[0,\eta_0)}(x_2) + \begin{bmatrix} 2 \\ 0 \\ p_0/(\gamma - 1) \end{bmatrix} \mathbb{1}_{[\eta_0,1]}(x_2)\,, \tag{53}$$

where the initial interface Γ_0 is parameterized by $(x_1, \eta_0(x_1))$ with $\eta_0(x_1) = 0.5 + 0.01\cos(6\pi x_1)$, and p_0 is the initial pressure, defined as

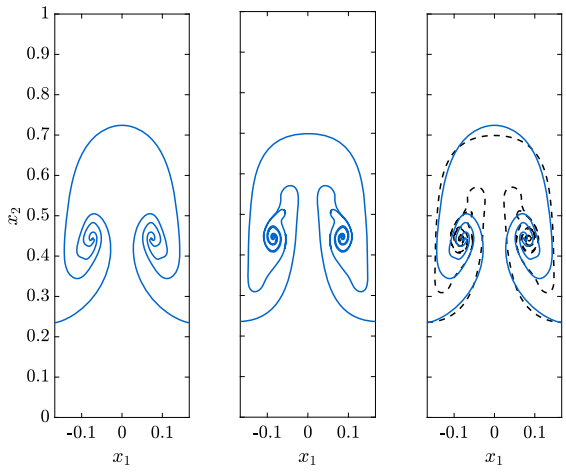


Fig. 22. Results for the multiscale algorithm applied to the compressible single-mode RTI test of Liska and Wendroff [62], with the underlying grid containing 8×32 cells and the interface discretized with $N = 64$. Shown are (a) computed interface parametrization z, (b) benchmark interface position, and (c) the computed interface z (blue) overlaying the benchmark solution (dashed black) at the final time $t = 8.5$.

$$p_0 = \begin{cases} 2.4 + g\,(\eta_0(x_1) - x_2) + 2g\,(1 - \eta_0(x_1)) & , \text{ if } x_2 < \eta_0(x_1) \\ 2.4 + 2g\,(1 - x_2) & , \text{ if } x_2 \geq \eta_0(x_1) \end{cases} .$$

Periodic conditions are applied in the horizontal direction x_1, while free-flow conditions are applied at the boundaries in the x_2 direction.

In [78], we used the anisotropic C-method to solve this problem; the resulting solutions have the classic mushroom-shaped interface profile without overly diffused KHI roll-up regions and mixing zones. We shall use this solution, computed on a fine mesh of 50×200 cells with CFL ≈ 0.1 as our *high resolution reference* solution.[6]

6.2.1. The multiscale algorithm applied to the RTI

We begin by employing the multiscale algorithm to the RTI problem (53) with the following parameter values: the coarse mesh for v contains 8×32 cells, the fine mesh for w uses $N = 64$, and the time-step is $\delta t = 2.5 \times 10^{-3}$, giving CFL ≈ 0.35. The Atwood number is set as $A = 1/3$, and the initial data is given by (53) (with u replaced by v) with the following modification: as in the numerical experiment performed in Section 6.1, we smooth the initial density field ρ_0 over a length scale. The initial density is thus given by formula (52) with $\eta_0(x_1) = 0.5 + 0.01\cos(6\pi x_1)$, and the smoothing length scale is chosen as $h = 0.02$. We also set

$$z_1(\alpha, 0) = \alpha\,,$$
$$z_2(\alpha, 0) = 0.5 + 0.01\cos(6\pi\alpha)\,,$$
$$\varpi\,(\alpha, 0) = 0\,.$$

The artificial viscosity parameters are chosen as $\beta = 5$, $\tilde{\delta} = 1.5$, and $\mu = 0.001$.

The multiscale interface position z at time $t = 8.5$ is shown in Fig. 22; the high resolution reference solution computation is presented in Fig. 22(b), and the comparison of the two solution is made in Fig. 22(c). We see that the spike tip position of the computed solution matches almost exactly with the spike tip position of the reference solution, and the bubble tip positions are also in good agreement. Moreover, the multiscale solution successfully simulates the roll-up of the contact discontinuity and, by comparing with Fig. 22(b), we observe that this roll-up occurs in the correct region of the flow and matches well with the *scale* of the high resolution solution.

In Fig. 23, we compare the interface $z(\alpha, t)$ from the multiscale run and the high-resolution run at various times t. As can be seen, the two solutions are in very good agreement throughout the time interval of the simulation.

To compare the bubble and spike tip positions, we plot in Fig. 24 the quantities $\sigma_b(t) = |0.5 - \max_\alpha z(\alpha, t)|$ and $\sigma_s(t) = |0.5 - \min_\alpha z(\alpha, t)|$. The spike tip position is in excellent agreement with the reference solution for all times t, while the bubble tip position is also in excellent agreement for times $t < 6$. For $t > 6$, the bubble tip position diverges slightly from

[6] We have found that this smaller CFL number is required for this low Mach number flow calculation to prevent high frequency noise from corrupting the solution.

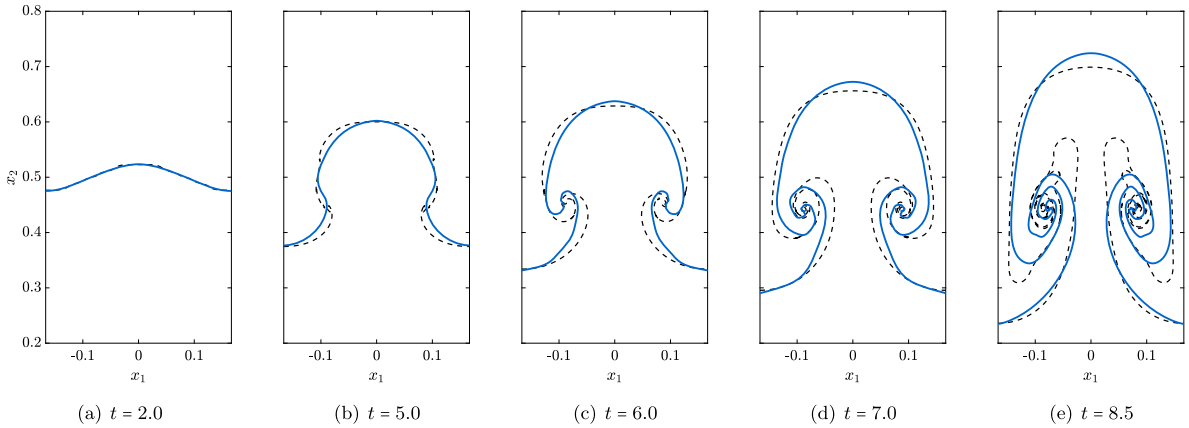


Fig. 23. Evolution over time t of the interface for the compressible single-mode RTI. Here z is computed using the multiscale algorithm on a mesh with 8×32 cells and an interface discretized with $N = 64$. The blue curve is the computed $z(\alpha, t)$, and the dashed black curve is the reference solution.

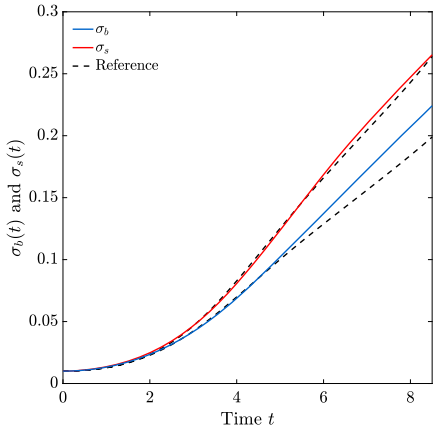


Fig. 24. Plots of the quantities $\sigma_b(t)$ and $\sigma_s(t)$ for the solution computed using the multiscale algorithm applied to the compressible single-mode RTI on a mesh with 8×32 cells and an interface with $N = 64$.

the reference solution, but the error is still relatively small; at the final time $t = 8.5$, the error is $\approx 2.5\%$ of the height of the computational domain.

Finally, we turn to the issue of the runtimes of the computations. The reference solution computation had a runtime of $T_{\text{CPU}} \approx 2906$ s, whereas the multiscale algorithm runtime was only $T_{\text{CPU}} \approx 14$ s, which gives a speed-up of approximately 203 times. We are thus able to infer both large-scale (amplitude growth rates) as well as small-scale (roll-up region structure) information by use of the multiscale model and algorithm, while drastically reducing the computational burden and runtime when compared with the reference solution computation.

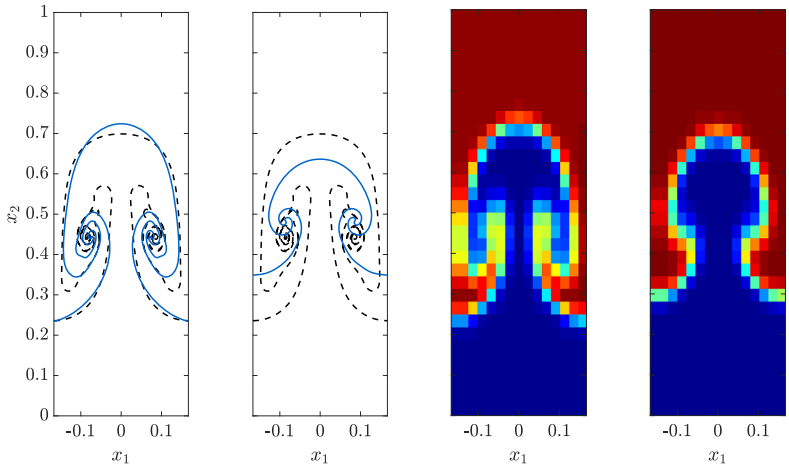
6.2.2. Comparing the mutliscale solution with both the low resolution simulation and the incompressible z-model

To demonstrate the efficacy of our multiscale model, we compare the multiscale solution with the low resolution solution as well as with the incompressible and irrotational z-model.

In Fig. 25, the results from these simulations are compared with the simulations performed using the multiscale algorithm. We see that the solutions are drastically different; the low resolution density ρ has no roll-up of the contact discontinuity, while the interface z computed using the incompressible z-model has completely incorrect bubble and spike positions. On the other hand, the use of the multiscale model allows the coarse grid calculation to "correct" the interface position z, while the fine resolution computation on the interface z similarly "corrects" for the lack of roll-up of the coarse grid solution.

6.2.3. Basic mesh refinement study

Next, we perform a basic mesh refinement study to analyze the behavior of our multiscale algorithm as both the underlying mesh, as well as the interface discretization, is refined. More precisely, we shall consider grids with 8×32, or 10×40 or 12×48 cells, and the interface discretized with either $N = 64$ or $N = 128$ or $N = 256$. We shall keep the artificial vis-



(a) Multiscale z (b) z-model (c) Multiscale ρ (d) Low res. ρ

Fig. 25. Comparison of the solutions to the compressible single-mode RTI test of Liska and Wendroff [62]. Figs. 25(a) and 25(c) are plots of the interface z and density ρ, respectively, computed using the multiscale algorithm. Fig. 25(b) is the interface z computed using the incompressible and z-model. Fig. 25(d) is the low resolution density ρ computed using the C-method. All the relevant parameters are fixed across the simulations.

cosity parameters β and $\tilde{\delta}$ fixed, with $\beta = 5$ and $\tilde{\delta} = 1.5$. The time-step δt and artificial viscosity parameter μ are allowed to vary as both the underlying mesh resolution, as well as the interface resolution, are varied. The exact choices for these parameters, as well as the corresponding runtimes T_{CPU} and speed-up factors Λ compared to the high resolution reference solution calculation, are presented in Table 3 in Appendix A, but we note here that the values of μ are almost the same ($\mu \approx 0.001$) for all of the runs.

The results for the mesh refinement study are shown in Fig. 34 in Appendix A. As N increases with the number of cells fixed, the computed interface positions are roughly the same, except in the KHI roll-up regions, in which the interfaces computed with larger N show significantly more roll-up (compare, for instance, Fig. 34(a) with Fig. 34(c)). This is in line with the observations in Section 4.3 for numerical simulations using the incompressible and irrotational z-model, and is due to the scaling of the parameter $\tilde{\delta}$.

On the other hand, if the number of cells contained in the underlying coarse mesh is increased with N held fixed, the resulting solutions for the interface do not have more roll-up, but instead appear to converge to the reference solution away from the roll-up region (compare, for instance, Fig. 34(b) with Fig. 34(h)). This "convergence" is particularly noticeable in the "pits" of the mushroom shape.

6.3. A single-mode RMI problem

We next consider the single-mode RMI with the following problem setup. The domain is $(x_1, x_2) \in [-1/6, +1/6] \times [-1, 1]$, the gravitational constant is $g = 0.5$, and the initial data is given by

$$\begin{bmatrix} \rho_0 \\ (\rho u)_0 \\ E_0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ f/(\gamma-1) \end{bmatrix} \mathbb{1}_{[-1,\eta_0)}(x_2) + \begin{bmatrix} 2 \\ 0 \\ f/(\gamma-1) \end{bmatrix} \mathbb{1}_{[\eta_0,0.8)}(x_2) + \begin{bmatrix} 4.857143 \\ 0 \\ 36.6666 + f/(\gamma-1) \end{bmatrix} \mathbb{1}_{[0.8,1]}(x_2), \quad (54)$$

where the initial interface Γ_0 is parameterized by $(x_1, \eta_0(x_1))$ with $\eta_0(x_1) = 0.5 + 0.1\cos(6\pi x_1)$, and f is the function defined as

$$f(x_1, x_2) = \begin{cases} 2.4 + g\,(\eta_0(x_1) - x_2) + 2g\,(1 - \eta_0(x_1)) & , \text{ if } x_2 < \eta_0(x_1) \\ 2.4 + 2g\,(1 - x_2) & , \text{ if } x_2 \geq \eta_0(x_1) \end{cases}.$$

Periodic conditions are applied in the horizontal direction x_1, while free-flow and solid-wall conditions are applied at the bottom and top boundaries, respectively.

A high resolution reference solution is computed using the C-method on a fine grid with 50×400 cells. The time-step is fixed as $\delta t = 0.0001$, which results in CFL $\in (0.14, 0.2)$. Again, we have found that this relatively small CFL number is required to prevent the occurrence of high-frequency noise in the computed solution.

6.3.1. The multiscale algorithm applied to the RMI

We apply the multiscale algorithm to the RMI problem (54) with the following parameter choices: the coarse mesh for v contains 10×80 cells, the fine mesh for w uses $N = 64$, and the time-step is $\delta t = 0.00125$, which yields CFL $\in (0.30, 0.45)$.

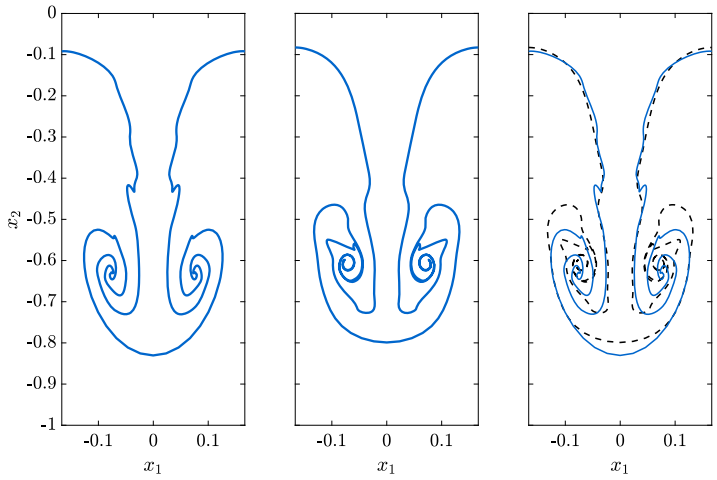


Fig. 26. Results for the multiscale algorithm applied to single-mode RMI test, with the underlying grid containing 10×80 cells and the interface discretized with $N = 64$. Shown are (a) computed interface parametrization z, (b) benchmark interface position, and (c) the computed interface z (blue) overlaying the benchmark solution (dashed black) at the final time $t = 1.6$.

The artificial viscosity parameters in (59) are set as $\beta = 1$, $\beta_s = 1$, while $\tilde{\delta} = 1.25$ and $\mu = 0.005$. The Atwood number is set as $A = 1/3$, and the initial data is given by (54) (with u replaced by v), together with

$$z_1(\alpha, 0) = \alpha\,,$$
$$z_2(\alpha, 0) = 0.5 + 0.1\cos(6\pi\alpha)\,,$$
$$\varpi(\alpha, 0) = 0\,.$$

We present results for the computed interface parametrization at the final time $t = 1.6$ in Fig. 26(a). Plots showing the evolution of the interface over time are shown in Fig. 27. The computed interface position z agrees well with the reference solution for the duration of simulation, and both interfaces display similar amounts of roll-up at the final time.The positions of the bubbles coincide for the two solution, while the spike tip positions are also in good agreement. We also note that in the high resolution reference solution there is slight "kink" in the "stem" of the mushroom; this "kink" is also displayed in the interface computed using the multiscale algorithm.

The approximate runtime of the reference solution calculation is $T_{\text{CPU}} \approx 1653$ s, whereas our multiscale algorithm computation had a runtime of only $T_{\text{CPU}} \approx 15$ s, yielding a speed-up factor of almost 110 times. As with the multiscale algorithm applied to the RTI, we are able to accurately predict large scale quantities and small scale structure with minimal computational expense.

6.3.2. The effects of the baroclinic term

In this section, we briefly discuss the importance of including the baroclinic term in the modified ϖ-equation (47b). This term is crucial in ensuring that the interface z is advected by the correct velocity. Without this term, information about the baroclinic deposition of vorticity by the shock on the interface is not transmitted to the incompressible portion of the multiscale algorithm. We show in Fig. 28 the evolution of a multiscale solution, computed without the inclusion of the baroclinic term, but otherwise identical to the RMI multiscale algorithm. This solution (displayed as a red curve) is compared against the solution obtained using the actual RMI multiscale algorithm (displayed as a blue curve), as well as the high resolution reference solution (displayed as a dashed black curve). It is clear that the omission of the baroclinic term leads to a solution with an incorrect interface position, and with significantly less KH roll-up.

In the absence of this baroclinic term, while the coarse scale shock velocity v still advects the interface upon shock-contact collision, the computed amplitude of vorticity ϖ (shown in Fig. 29 as a red curve) has the wrong sign after the shock-contact collision at $t \approx 0.12$, which yields an incorrect calculation of the fine scale velocity w, and subsequently an incorrect interface position z.

We compare this incorrect ϖ with the amplitude of vorticity ϖ computed using the complete RMI multiscale algorithm (shown in Fig. 29 as a blue curve); without the baroclinic term, ϖ has the wrong sign until time $t = 0.70$, at which point the advection of the interface by the coarse scale velocity v *forces* ϖ to have the correct sign. This is in contrast to our RMI multiscale algorithm, in which the baroclinic term forces ϖ to switch sign after the shock-contact interaction, which results in the correct computation of the fine grid velocity w and, consequently, the correct interface position.

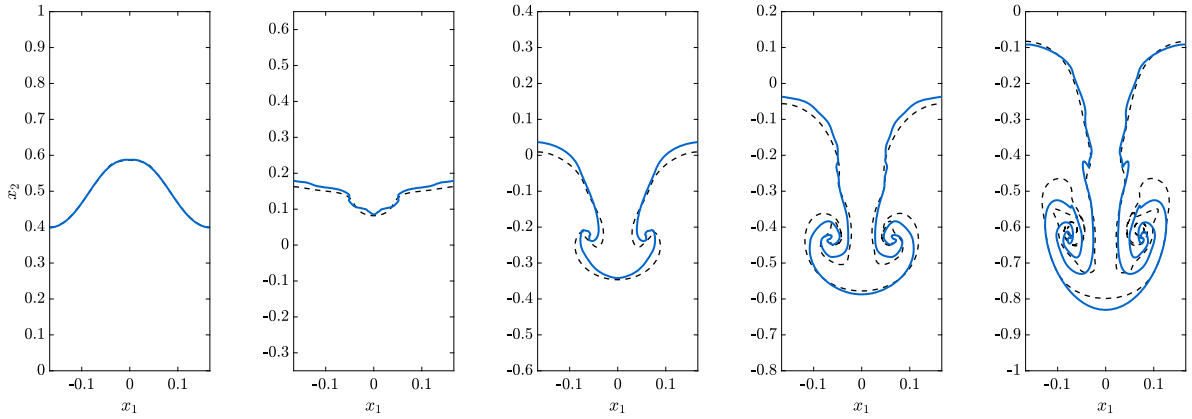


Fig. 27. Evolution over time t of the interface for the single-mode RMI. Here z is computed using the multiscale algorithm on a mesh with 10×80 cells and an interface discretized with $N = 64$. The blue curve is the computed $z(\alpha, t)$, and the dashed black curve is the reference solution.

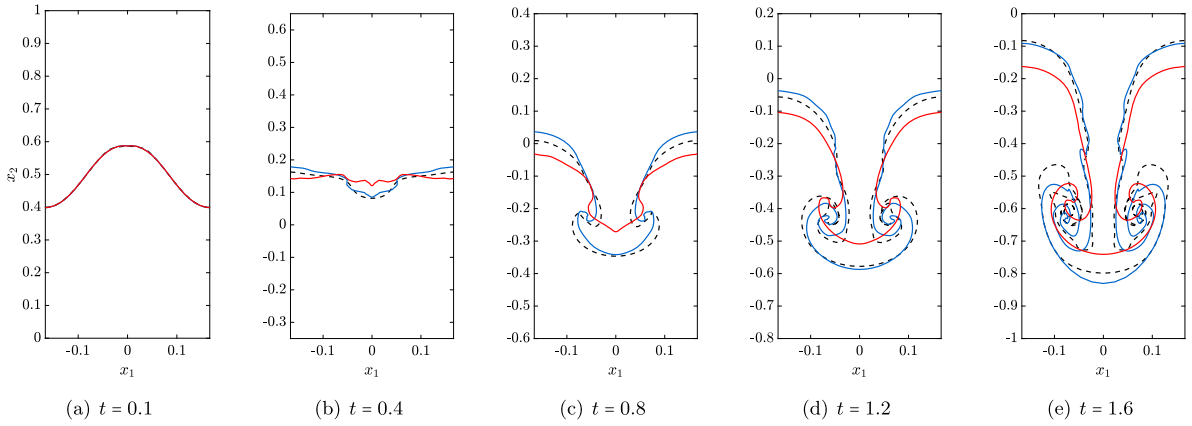


Fig. 28. Evolution over time t of the interface for the single-mode RMI. Here, the red curve is z computed using the multiscale algorithm, but without the use of the baroclinic term in the ϖ-equation, on a mesh with 10×80 cells and an interface discretized with $N = 64$. The blue curve is z computed using the complete multiscale algorithm, while dashed black curve is the reference solution.

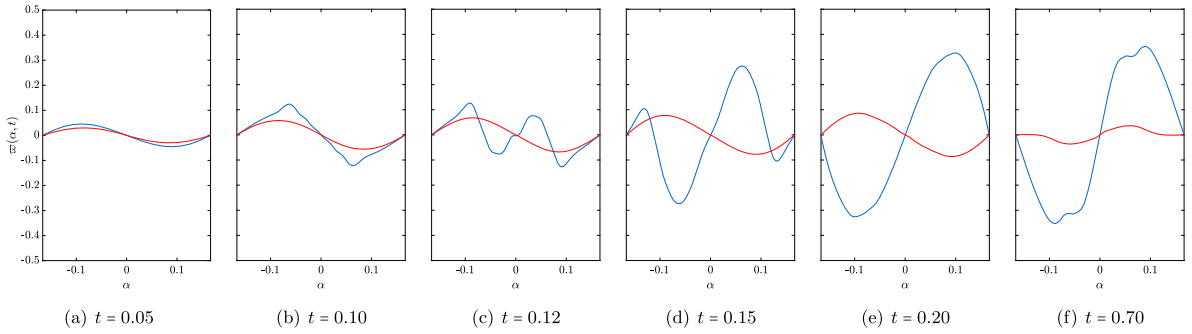


Fig. 29. Evolution over time t of the amplitude of vorticity $\varpi(\alpha, t)$ versus α for the single-mode RMI. Here, the red curve is ϖ computed using the multiscale algorithm, but without the use of the baroclinic term in the ϖ-equation, on a mesh with 10×80 cells and an interface discretized with $N = 64$. The blue curve is ϖ computed using the complete multiscale algorithm.

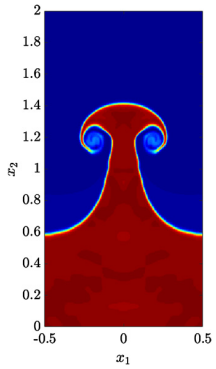


Fig. 30. The density profile at time $t = 8.0$ for the RMI test of Nourgaliev et al. [75]. This high-resolution solution is computed using the C-method on a mesh with 100×800 cells.

6.4. The RMI test of Nourgaliev et al.

We next consider the RMI problem introduced in [75], and later considered in [97,74,105]. A heavy fluid of density $\rho^+ = 5.04$ lies below a light fluid of density $\rho^- = 1$, and a planar Mach 1.24 shock travels vertically downwards through the light fluid and eventually collides with the interface separating the two fluids, resulting in a transmitted shock and a reflected shock. The instability is generated by the subsequent acceleration of the light fluid into the heavy fluid. The domain is $(x_1, x_2) \in [-0.5, 0.5] \times [0, 4]$, the adiabatic constant is $\gamma = 1.276$, gravity is assumed to be negligible (i.e. $g = 0$), and the initial data is defined as follows. The initial interface Γ_0 is parametrized by $(x_1, \eta_0(x_1))$, with $\eta_0(x_1) = 2.9 + 0.1\cos(2\pi x_1)$, and the initial shock position is at $x_2 = 3.2$. The initial horizontal velocity vanishes $u_1(x, 0) = 0$, and the initial vertical velocity satisfies $u_2(x, 0) = -0.550368 \cdot \mathbb{1}_{x_2 \geq 3.2}(x)$. The initial pressure is $p_0(x) = \mathbb{1}_{x_2 < 3.2}(x) + 1.628 \cdot \mathbb{1}_{x_2 \geq 3.2}(x)$. As in the numerical experiments in Section 6.1, the initial density is smoothed over a length scale h using a tanh profile:

$$\rho_0(x_1, x_2) = \rho^- + \frac{\rho^+ - \rho^-}{2}\left[1 + \tanh\left(\frac{\psi(x_1) - x_2}{h}\right)\right]. \tag{55}$$

For our high resolution reference solution computed using the C-method on a fine mesh, we use $h = 0.005$. Periodic boundary conditions are applied in the x_1-direction, and free-flow conditions are imposed in the x_2-direction.

Our high resolution reference solution is computed on a mesh with 100×800 cells and a time-step of $\delta t = 6.25 \times 10^{-4}$, which gives CFL ≈ 0.4. The computed density is shown in Fig. 30 at the final time $t = 8$, and can be compared with Figs. 10, 11, 23, and 23 in [75], [97], [74], and [105], respectively.

6.4.1. The multiscale algorithm applied to the RMI

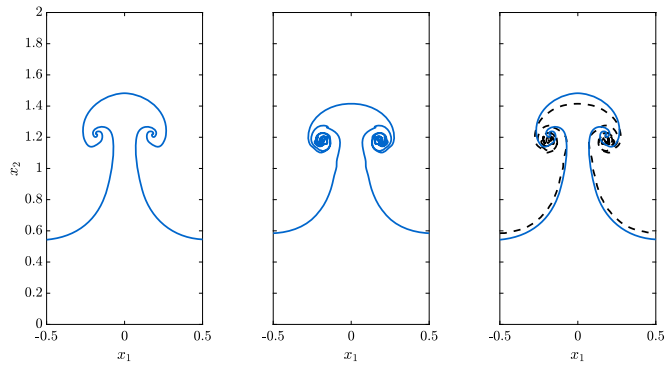
Next, we apply the multiscale RMI algorithm to the problem described above. The relevant parameters are chosen as follows. The coarse mesh for v contains 10×80 cells, the fine mesh for w uses $N = 128$, and the time-step is set as $\delta t = 6.25 \times 10^{-3}$, giving CFL ≈ 0.4. The initial velocities are $v_1(x, 0) = 0$ and $v_2(x, 0) = -0.550368 \cdot \mathbb{1}_{x_2 \geq 3.2}(x)$, and the initial pressure is $p_0(x) = \mathbb{1}_{x_2 < 3.2}(x) + 1.628 \cdot \mathbb{1}_{x_2 \geq 3.2}(x)$. The initial density is smoothed using (55) with $h = 0.02$. The Atwood number is $A \approx 0.67$, and the initial data for the interface calculation is

$$z_1(\alpha, 0) = \alpha\,,$$
$$z_2(\alpha, 0) = 2.9 + 0.1\cos(2\pi\alpha)\,,$$
$$\varpi(\alpha, 0) = 0\,.$$

The artificial viscosity parameters are set as $\beta = 1$, $\beta_s = 0$, $\tilde{\delta} = 1$, and $\varpi = 1 \times 10^{-4}$.

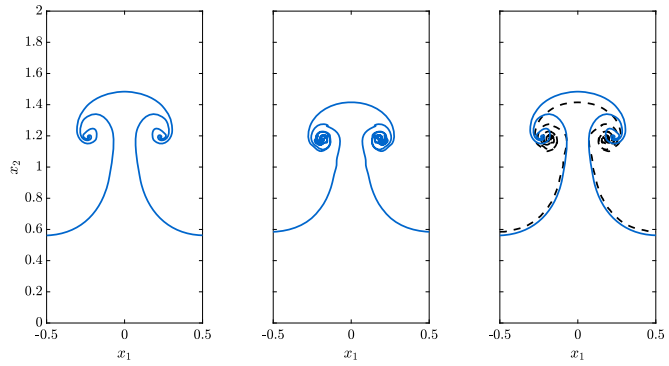
The computed interface is shown in Fig. 31(a), the high resolution reference solution is shown in Fig. 31(b), and a comparison of the two solutions is made in Fig. 31(c). The multiscale algorithm is able to simulate both the transport of the contact as well as the KHI roll up; moreover, Fig. 31(c) shows that the bubble tip and spike tip positions compare well with the high resolution reference solution. The runtime for the high resolution simulation was $T_{\text{CPU}} \approx 6298$ s, whereas the runtime for the multiscale simulation was only $T_{\text{CPU}} \approx 24$ s, giving a speed up factor of approximately 260 times.

Next, we double the resolution of both the coarse mesh for v as well as the fine mesh for w i.e. the grid now contains 20×160 cells and $N = 256$. The time-step is halved, $\delta t = 3.125 \times 10^{-3}$, so that the CFL number is still approximately 0.4. All the other parameters from the previous run are kept fixed, and we employ the multiscale algorithm; the resulting solution is shown in Fig. 32(a). We see that there is more roll-up of the vortex sheet, and that the "cap" of the mushroom is "flattened", which is in qualitative agreement with the reference solution. The runtime for this simulation is $T_{\text{CPU}} \approx 265$ s, giving a speed up factor of approximately 24 times over the high resolution simulation.



(a) Multiscale z (b) High-res. (c) Comparison

Fig. 31. Results for the multiscale algorithm applied to single-mode RMI test of Nourgaliev et al. [75], with the underlying grid containing 10×80 cells and the interface discretized with $N = 128$. Shown are (a) computed interface parametrization z, (b) benchmark interface position, and (c) the computed interface z (blue) overlaying the benchmark solution (dashed black) at the final time $t = 8.0$.



(a) Multiscale z (b) High-res. (c) Comparison

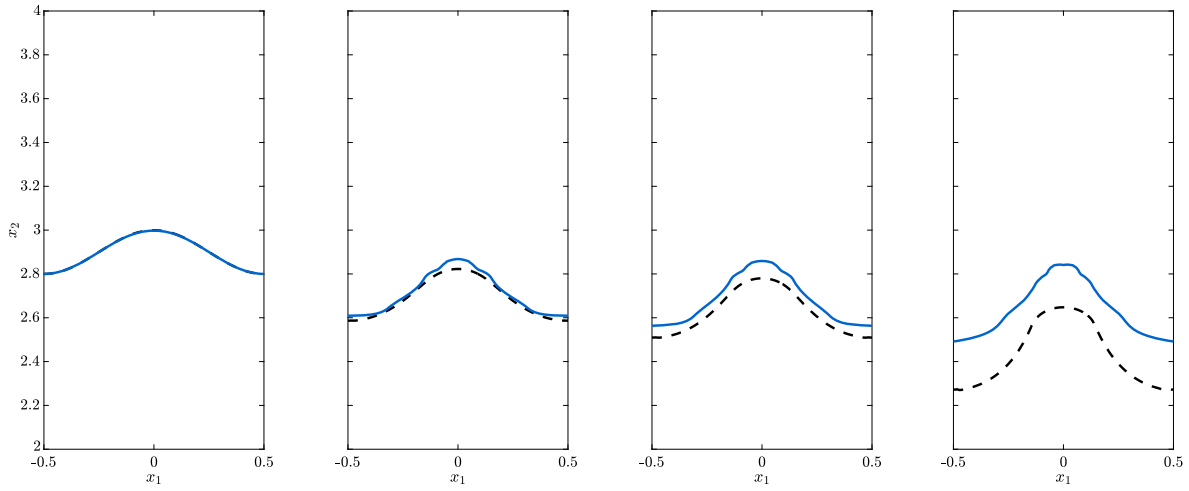
Fig. 32. Results for the multiscale algorithm applied to single-mode RMI test of Nourgaliev et al. [75], with the underlying grid containing 20×160 cells and the interface discretized with $N = 256$. Shown are (a) computed interface parametrization z, (b) benchmark interface position, and (c) the computed interface z (blue) overlaying the benchmark solution (dashed black) at the final time $t = 8.0$.

6.4.2. The effects of the Taylor hypothesis

We now briefly discuss the effects of the Taylor "frozen turbulence" hypothesis in our multiscale RMI algorithm. We recall that the hypothesis asserts that the fine grid velocity w is transported by the coarse grid velocity v (over small time intervals). The primary effect of the Taylor hypothesis is that the dynamics of the coarse grid compressible velocity v are evolved using (28b') in place of (28b). Without the use of the hypothesis, the additional terms in (28b) lead to an incorrect calculation of the velocity field v, which in turn leads to an incorrect update of the interface position z (i.e. Steps **1(e)**, **1(f)** and **4(b)**, **4(c)** of the RMI algorithm).

We demonstrate this in Fig. 33, in which we show results of a simulation performed using a modified version of the RMI multiscale algorithm in which the Taylor hypothesis is not employed, but is otherwise identical to the RMI multiscale algorithm in Section 5.3. The solution is computed using the same parameters as those used for the simulation presented in Fig. 31(a).

The shock collides with the interface at time $t = 0.15$, shown in Fig. 33(a); at this time, vorticity is deposited on the interface by the shock. The interface is transported with reasonable accuracy for $t < 1.0$ (as demonstrated in Fig. 33(b)), because the velocity w is very small at early times, and thus does not have a noticeable effect on the interface evolution; for early times, the evolution of the interface is mainly due to the "transport" coarse grid velocity v. However, as the magnitude of the vorticity on the interface grows with time, w is no longer negligible, and affects the evolution of the interface. The extra terms appearing on the right-hand side of (28b) (which do not appear in (28b')) lead to an incorrect calculation of the coarse scale velocity v. Since the fine grid velocity w is not being transported by v, the coarse scale compression and vorticity information is not correctly conveyed to the interfacial algorithm, which leads to a deceleration of the interface (shown in Fig. 33(c)). The solution then quickly degrades, with the interface position completely incorrect by time $t = 2.0$, as shown in Fig. 33(d). For problems in which an unstable interface is transported as the instability develops,



(a) $t = 0.15$ (b) $t = 1.0$ (c) $t = 1.25$ (d) $t = 2.0$

Fig. 33. Results for the modified multiscale algorithm without the Taylor "frozen turbulence" hypothesis applied to single-mode RMI test of Nourgaliev et al. [75], with all relevant parameters unchanged from the simulation presented in Fig. 31. Shown are the computed interface z (blue) and reference solution (dashed black) at various times t.

the use of the Taylor hypothesis ensures that coarse scale information is accurately conveyed to the small scale calculations, thereby producing accurate solutions with correct interface positions.

7. Concluding remarks

This paper introduces a novel multiscale model describing the evolution of contact discontinuities in compressible fluid flow. The multiscale model is based on a decomposition of the velocity field $u = v + w$. While the velocity w is discontinuous, it is also incompressible and irrotational, and can be solved efficiently on fine meshes using a new asymptotic (high-order) z-model to approximate the full Birkhoff-Rott system of singular integral equations. The velocity v is compressible and rotational, but is smooth near the contact discontinuity and can thus be computed efficiently on relatively coarse meshes.

We have proposed an extremely simple numerical implementation of the incompressible z-model, and have presented numerical results for the z-model which simulate classical RTI experiments. These results show excellent qualitative and quantitative agreement of our computed solutions with both experimental predictions, as well as "reference" solution calculations performed using the complete Birkhoff-Rott system. In the latter case, we have demonstrated that our z-model algorithm is at least 75 times faster than a standard numerical algorithm for the Birkhoff-Rott system. We have additionally compared our simple numerical implementation with more sophisticated methods using higher order regularizations, and found good agreement (as in previous numerical studies [8,93]) between all three methods in predicting both the bubble and spike tip locations, as well as the radii and location of the spiral roll up structures. While our simple numerical method leads to a fast-running algorithm, in the future, more sophisticated numerical implementations of the z-model will be considered, including implementations of a fast summation method [45], a point-insertion procedure [57], non-oscillatory shock-capturing, and space-time smooth artificial viscosity [77].

We have also developed multiscale models and algorithms for compressible flows with vorticity undergoing RTI and RMI. Our algorithms couple the fine scale velocity w, which controls the small scale structure of the interface, with the coarse grid velocity v, which controls the bulk compression and vorticity of the fluid. The RMI multiscale algorithm includes the effects of vorticity deposition on the interface during shock-contact interaction, and also enforces a version of Taylor's frozen turbulence hypothesis, which asserts that small scale velocity fluctuations are transported by the mean flow. We have presented numerical results for both the RTI and RMI, and demonstrated that the computed solutions are in good agreement, both qualitatively and quantitatively, with high order gas dynamics simulations, while having computational run times that are at least two orders of magnitude smaller. In particular, the multiscale solutions exhibit KHI roll up regions of the contact discontinuity that are in good agreement with the roll up regions observed in solutions obtained from high resolution calculations. Such roll up of the contact is in general not observed in low resolution Euler simulations; however, the coupling of the fine scale velocity w to the coarse scale velocity v through our multiscale model leads to simulations (on coarse meshes) which exhibit roll-up regions similar to those in high-resolution simulations.

In future work, we shall generalize our models to three space dimensions, and consider its applications to the numerical simulation of other physical problems, such as RTI and RMI with multimode initial perturbations, rising bubbles, and shock-bubble interaction. The mathematical analysis of the z-model and the multiscale models will also be considered in

Table 3
Time-step δt and artificial viscosity parameter μ choices for the compressible single-mode RTI mesh refinement study described in Section 6.2.3. Shown also are the runtimes T_{CPU} and speed-up factors Λ.

Cells \ N	64	128	256
8×32	$\delta t = 2.5 \times 10^{-3}$ $\mu = 1 \times 10^{-3}$ $T_{\text{CPU}} = 14$ s $\Lambda = 203$	$\delta t = 2.5 \times 10^{-3}$ $\mu = 9 \times 10^{-4}$ $T_{\text{CPU}} = 28$ s $\Lambda = 105$	$\delta t = 6.25 \times 10^{-4}$ $\mu = 8.75 \times 10^{-4}$ $T_{\text{CPU}} = 244$ s $\Lambda = 12$
10×40	$\delta t = 2.5 \times 10^{-3}$ $\mu = 1 \times 10^{-3}$ $T_{\text{CPU}} = 19$ s $\Lambda = 149$	$\delta t = 2.5 \times 10^{-3}$ $\mu = 8.75 \times 10^{-4}$ $T_{\text{CPU}} = 37$ s $\Lambda = 78$	$\delta t = 6.25 \times 10^{-4}$ $\mu = 9 \times 10^{-4}$ $T_{\text{CPU}} = 304$ s $\Lambda = 10$
12×48	$\delta t = 1.67 \times 10^{-3}$ $\mu = 1.5 \times 10^{-3}$ $T_{\text{CPU}} = 38$ s $\Lambda = 76$	$\delta t = 1.67 \times 10^{-3}$ $\mu = 1.25 \times 10^{-3}$ $T_{\text{CPU}} = 69$ s $\Lambda = 42$	$\delta t = 4.55 \times 10^{-4}$ $\mu = 1.1 \times 10^{-3}$ $T_{\text{CPU}} = 578$ s $\Lambda = 5$

future work. In particular, we shall consider a detailed theoretical and numerical study of the convergence behavior of our multiscale solutions, as the interfacial and planar meshes are refined, and artificial viscosity parameters converge to zero.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

SS was partially supported by DTRA HDTRA11810022. We would like to express our gratitude to the anonymous referee for their numerous suggestions that have greatly improved the manuscript.

Appendix A. Mesh refinement for the multiscale algorithm applied to the RTI

In this section, we present the results for the mesh refinement study of the multiscale algorithm applied to the single-mode compressible RTI test of Liska & Wendroff (see Section 6.2.3). Table 3 contains a list of the choices for the parameters δt and μ, as well as the runtimes T_{CPU} and relative speed up factors Λ for the RTI mesh refinement study. The results from this study are presented in Fig. 34, which show the reference solution (dashed black curve) overlaid by the computed interface parametrization z (red curve).

Appendix B. The C-method for space-time smooth artificial viscosity

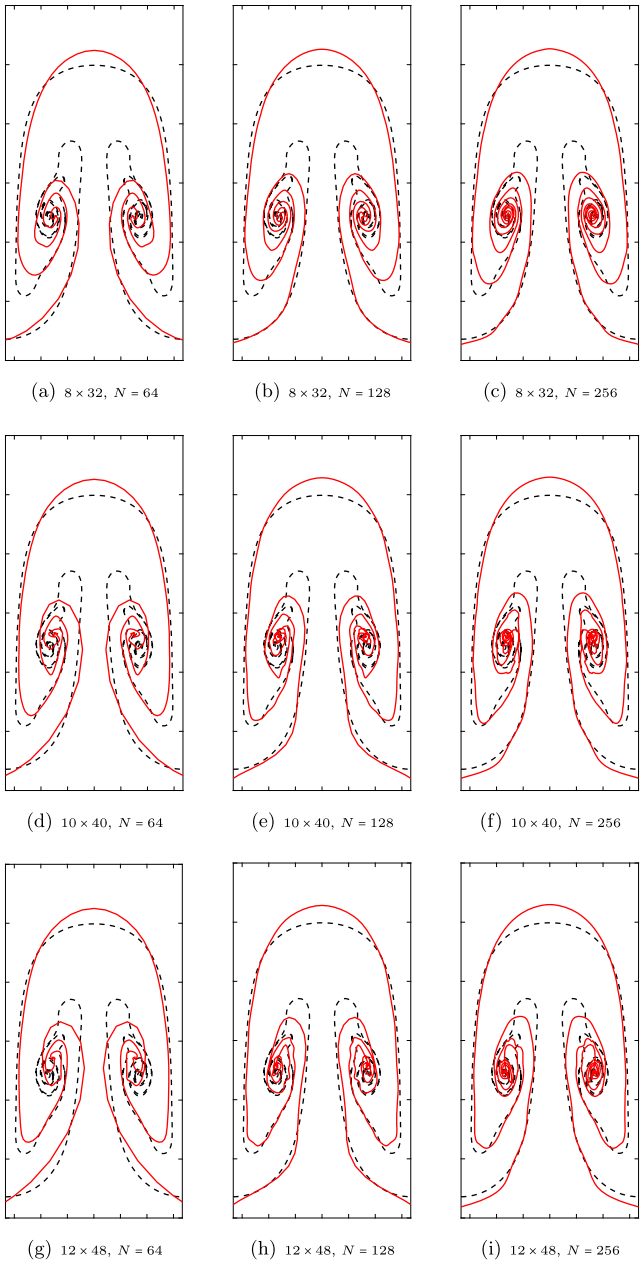
The presence of jump discontinuities in the solution $\mathbf{U}$ to (10) poses a significant challenge for numerical schemes attempting to approximate such solutions, due to the occurrence of small-scale oscillations (or Gibbs phenomenon). A variety of high-order discretization techniques have been developed to combat this issue, such as MUSCL [101,24,52], PPM [25], WENO [64,53,86], and its predecessor, ENO [49,87,88]. These methods rely on a careful reconstruction of the numerical flux; centered numerical fluxes, such as the Lax-Friedrichs flux [60], add dissipation implicitly to preserve stability and monotonicity, while upwinding methods based upon exact or approximate Riemann solvers tend to be complex and computationally costly. We refer the reader to [77] and the references therein for further details.

Explicit artificial viscosity methods provide a simple way to stabilize shock fronts and contact discontinuities. These methods regularize solutions by introducing diffusion terms to the equations of motion; discontinuities are *smeared* over a small region in space, which stabilize the solution and prevent the occurrence of spurious oscillations, while high-order accuracy is maintained in smooth regions of the flow. We next describe a method for adding localized, space-time smooth artificial viscosity to the system (10), which we call the C-method [80,77,78].

The C-method is a variant of the original classical artificial viscosity method of Von Neumann and Richtmyer [103], and couples the Euler equations (10) to a set of scalar reaction-diffusion equations, whose solutions act as space-time smooth artificial viscosity indicators. The C-method tracks the geometry of the evolving fronts, which allows for the implementation of both directionally isotropic and anisotropic artificial viscosity schemes. The latter is important for the capture of the roll-up of vortex sheets subject to the Kelvin-Helmholtz instability.

We introduce the following Euler-C-$\hat{C}$ system:

$$\partial_t \rho + \nabla \cdot (\rho u) = 0\,, \tag{56a}$$



(a) 8×32, $N = 64$ (b) 8×32, $N = 128$ (c) 8×32, $N = 256$

(d) 10×40, $N = 64$ (e) 10×40, $N = 128$ (f) 10×40, $N = 256$

(g) 12×48, $N = 64$ (h) 12×48, $N = 128$ (i) 12×48, $N = 256$

Fig. 34. Plots of the interface computed using the multiscale algorithm applied to the single-mode compressible RTI test of Liska and Wendroff [62] for different grid and interface resolutions. The red curve is the computed interface, and the dashed black curve is the reference solution.

$$\partial_t(\rho u) + \nabla \cdot (\rho u \otimes u) + \nabla p + \rho g e_2 = \partial_{x_i}\left(\tilde{\beta}\rho\hat{C}C^{\tau_i}C^{\tau_j}\partial_{x_j}u\right) + \nabla \cdot (\tilde{\beta}_s \rho C \nabla u)\,, \tag{56b}$$

$$\partial_t E + \nabla \cdot (u(E+p)) + \rho g u_2 = 0\,, \tag{56c}$$

$$\partial_t C - \mathcal{L}[C\,;\varepsilon,\kappa] = \frac{S(u)}{\varepsilon|\delta x|}G_\rho\,, \tag{56d}$$

$$\partial_t \hat{C} - \mathcal{L}[\hat{C}\,;\varepsilon,\kappa] = \frac{S(u)}{\varepsilon|\delta x|}\hat{G}_\rho\,, \tag{56e}$$

$$\partial_t C^{\tau_i} - \mathcal{L}[C^{\tau_i}\,;\varepsilon,\kappa] = \frac{S(u)}{\varepsilon|\delta x|}\hat{\tau}_i\,,\ \text{for } i = 1,2\,. \tag{56f}$$

Here, the operator $\mathcal{L}$ is defined by

$$\mathcal{L}[C\,;\varepsilon,\kappa] = -\frac{S(u)}{\varepsilon|\delta x|}C + \kappa\, S(u)\Delta C\,, \tag{57}$$

where ε and κ are parameters controlling the support and smoothness of the function C, respectively, $\delta x = (\delta x_1\,, \delta x_2)$ is the grid spacing, and $\Delta = \partial_{x_1}^2 + \partial_{x_2}^2$ is the Laplacian operator. The forcing functions to the C-equations are

$$G_\rho = \mathbb{1}_{(-\infty,0)}(\nabla\cdot u)[1 - \mathbb{1}_{(\infty,0)}(\partial_n\rho\partial_n e)]|\nabla\rho|\,, \tag{58a}$$

$$\hat{G}_\rho = \mathbb{1}_{(-\infty,0)}(\partial_n\rho\partial_n e)|\nabla\rho|\,, \tag{58b}$$

$$\hat{\tau}_1 = -\mathbb{1}_{(-\infty,0)}(\partial_n\rho\partial_n e)\partial_y\rho\,, \tag{58c}$$

$$\hat{\tau}_2 = \mathbb{1}_{(-\infty,0)}(\partial_n\rho\partial_n e)\partial_x\rho\,, \tag{58d}$$

where $\partial_n = n\cdot\nabla$ is the normal derivative operator and the function $\mathbb{1}_{(-\infty,0)}(\xi)$ is a compression or expansion switch defined by

$$\mathbb{1}_{(-\infty,0)}(\xi) = \begin{cases} 1\,, & \text{if } \xi < 0\,, \\ 0\,, & \text{if } \xi \geq 0\,. \end{cases}$$

The artificial viscosity parameters $\tilde{\beta}$ and $\tilde{\alpha}$ are defined by

$$\tilde{\beta} = |\delta x|^2\cdot\frac{\max_x|\nabla u|}{\mu^2\max_x\hat{C}}\beta \quad \text{and} \quad \tilde{\beta_s} = |\delta x|^2\cdot\frac{\max_x|\nabla u|}{\max_x C}\beta_s\,, \tag{59}$$

with $\mu = \max_x\left\{|C^{\tau_1}|\,, |C^{\tau_2}|\right\}$ a normalization constant, and β and α constant positive numbers.

In writing the system (56), we have utilized the summation convention which specifies that a repeated free index in the same term implies summation over all values of that index.

The two artificial viscosity terms in (56) are

$$\partial_{x_i}\left(\tilde{\beta}\rho\hat{C}C^{\tau_i}C^{\tau_j}\partial_{x_j}u\right)\,, \tag{60a}$$

$$\nabla\cdot(\tilde{\beta_s}\rho C\nabla u)\,. \tag{60b}$$

(60a) is an *anisotropic* artificial viscosity term that adds dissipation only in directions tangential to the front. This artificial viscosity term is localized to the vortex sheet through the use of the C-functions; the expansion switch $\mathbb{1}_{(-\infty,0)}(\partial_n e\partial_n\rho)$ in the forcing functions $\hat{G}_\rho$ and $\hat{\tau}_i$ vanishes at shock fronts, but is active at contact discontinuities. The *isotropic* artificial viscosity operator (60b), on the other hand, adds dissipation in all directions; this artificial viscosity term is localized to the shock fronts through the compression switch $\mathbb{1}_{(-\infty,0)}(\nabla\cdot u)$ in the forcing function G_ρ. For the vortex sheet, it is important that dissipation is added only in directions tangential to the sheet; therefore, the switch $1 - \mathbb{1}_{(-\infty,0)}(\partial_n e\partial_n\rho)$ "turns off" the isotropic dissipation in the regions where a shock front intersects with a vortex sheet.

Remark 3. If there are no shock fronts present in a solution, as is the case for the classical Rayleigh-Taylor problems, then the isotropic diffusion term (60b) in (56) is omitted, and consequently so is equation (56d) for the C-function localized to shock waves.

Remark 4. For problems which are symmetric about $x_1 = 0$, we compute the solution only for $x_1 \geq 0$ and then use reflection to obtain the solution for $x_1 < 0$. A similar reflection procedure is used in the numerical implementation of the z-model.

For the purposes of brevity, we have omitted some of the details and refer the reader to [78] for further discussion of the C-method in 2-D.

References

[1] A.S. Almgren, V.E. Beckner, J.B. Bell, M.S. Day, L.H. Howell, C.C. Joggerst, M.J. Lijewski, A. Nonaka, M. Singer, M. Zingale, CASTRO: a new compressible astrophysical solver. I. Hydrodynamics and self-gravity, Astrophys. J. 715 (2) (may 2010) 1221–1238, https://doi.org/10.1088/0004-637x/715/2/1221.

[2] C. Anderson, An implementation of the fast multipole method without multipoles, SIAM J. Sci. Stat. Comput. 13 (4) (1992) 923–947, https://doi.org/10.1137/0913055.

[3] C.R. Anderson, A vortex method for flows with slight density variations, J. Comput. Phys. (ISSN 0021-9991) 61 (3) (1985) 417–444, https://doi.org/10.1016/0021-9991(85)90073-7, http://www.sciencedirect.com/science/article/pii/0021999185900737.

[4] A. Appel, An efficient program for many-body simulation, SIAM J. Sci. Stat. Comput. 6 (1) (1985) 85–103, https://doi.org/10.1137/0906008.

[5] C.H. Aurther, R. Granero-Belinchón, S. Shkoller, J. Wilkening, Rigorous asymptotic models of water waves, Water Waves (ISSN 2523-3688) (Mar 2019), https://doi.org/10.1007/s42286-019-00005-w.

[6] G.R. Baker, The "cloud in cell" technique applied to the roll up of vortex sheets, J. Comput. Phys. (ISSN 0021-9991) 31 (1) (1979) 76–95, https://doi.org/10.1016/0021-9991(79)90063-9.
[7] G.R. Baker, J.T. Beale, Vortex blob methods applied to interfacial motion, J. Comput. Phys. (ISSN 0021-9991) 196 (1) (2004) 233–258, https://doi.org/10.1016/j.jcp.2003.10.023.
[8] G.R. Baker, L.D. Pham, A comparison of blob methods for vortex sheet roll-up, J. Fluid Mech. 547 (2006) 297–316, https://doi.org/10.1017/S0022112005007305.
[9] G.R. Baker, D.I. Meiron, S.A. Orszag, Vortex simulations of the Rayleigh-Taylor instability, Phys. Fluids 23 (8) (1980) 1485–1490, https://doi.org/10.1063/1.863173.
[10] J. Barnes, P. Hut, A hierarchical O(N log N) force-calculation algorithm, Nature 324 (Dec. 1986) 446–449, https://doi.org/10.1038/324446a0.
[11] J. Beale, A. Majda, High order accurate vortex methods with explicit velocity kernels, J. Comput. Phys. (ISSN 0021-9991) 58 (2) (1985) 188–208, https://doi.org/10.1016/0021-9991(85)90176-7, http://www.sciencedirect.com/science/article/pii/0021999185901767.
[12] J.T. Beale, A. Majda, Vortex methods. II. Higher order accuracy in two and three dimensions, Math. Comput. (ISSN 0025-5718) 39 (159) (1982) 29–52, https://doi.org/10.2307/2007618.
[13] R. Betti, V.N. Goncharov, R.L. McCrory, C.P. Verdon, Growth rates of the ablative Rayleigh-Taylor instability in inertial confinement fusion, Phys. Plasmas 5 (5) (1998) 1446–1454, https://doi.org/10.1063/1.872802.
[14] G. Birkhoff, Helmholtz and Taylor instability, in: Proc. Sympos. Appl. Math., vol. XIII, American Mathematical Society, Providence, R.I., 1962, pp. 55–76.
[15] M. Brouillette, The Richtmyer-Meshkov instability, Annu. Rev. Fluid Mech. 34 (1) (2002) 445–468, https://doi.org/10.1146/annurev.fluid.34.090101.162238.
[16] R. Caflisch, J. Lowengrub, Convergence of the vortex method for vortex sheets, SIAM J. Numer. Anal. 26 (5) (1989) 1060–1080, https://doi.org/10.1137/0726059.
[17] J. Carrier, L. Greengard, V. Rokhlin, A fast adaptive multipole algorithm for particle simulations, SIAM J. Sci. Stat. Comput. (ISSN 0196-5204) 9 (4) (1988) 669–686, https://doi.org/10.1137/0909044.
[18] C.-H. Cheng, S. Shkoller, D. Coutand, On the motion of vortex sheets with surface tension in three-dimensional Euler equations with vorticity, Commun. Pure Appl. Math. 61 (12) (2008) 1715–1752, https://doi.org/10.1002/cpa.20240.
[19] C.H.A. Cheng, S. Shkoller, Solvability and regularity for an elliptic system prescribing the curl, divergence, and partial trace of a vector field on Sobolev-class domains, J. Math. Fluid Mech. (ISSN 1422-6928) 19 (3) (2017) 375–422, https://doi.org/10.1007/s00021-016-0289-y.
[20] A.J. Chorin, Numerical study of slightly viscous flow, J. Fluid Mech. 57 (4) (1973) 785–796, https://doi.org/10.1017/S0022112073002016.
[21] A.J. Chorin, Vortex models and boundary layer instability, SIAM J. Sci. Stat. Comput. 1 (1) (1980) 1–21, https://doi.org/10.1137/0901001.
[22] A.J. Chorin, P.S. Bernard, Discretization of a vortex sheet, with an example of roll-up, J. Comput. Phys. (ISSN 0021-9991) 13 (3) (1973) 423–429, https://doi.org/10.1016/0021-9991(73)90045-4, http://www.sciencedirect.com/science/article/pii/0021999173900454.
[23] J. Christiansen, Numerical simulation of hydrodynamics by the method of point vortices, J. Comput. Phys. (ISSN 0021-9991) 13 (3) (1973) 363–379, https://doi.org/10.1016/0021-9991(73)90042-9, http://www.sciencedirect.com/science/article/pii/0021999173900429.
[24] P. Colella, A direct Eulerian MUSCL scheme for gas dynamics, SIAM J. Sci. Stat. Comput. (ISSN 0196-5204) 6 (1) (1985) 104–117, https://doi.org/10.1137/0906009.
[25] P. Colella, P.R. Woodward, The Piecewise Parabolic Method (PPM) for gas-dynamical simulations, J. Comput. Phys. 54 (Sept. 1984) 174–201, https://doi.org/10.1016/0021-9991(84)90143-8.
[26] A. Córdoba, D. Córdoba, F. Gancedo, Interface evolution: water waves in 2-D, Adv. Math. (ISSN 0001-8708) 223 (1) (2010) 120–173, https://doi.org/10.1016/j.aim.2009.07.016.
[27] G.-H. Cottet, P.D. Koumoutsakos, Vortex Methods: Theory and Practice, Cambridge University Press, 2000.
[28] J.-F. Coulombel, A. Morando, Stability of contact discontinuities for the nonisentropic Euler equations, Ann. Univ. Ferrara 50 (2004) 79, https://doi.org/10.1007/BF02825344.
[29] J.-F. Coulombel, P. Secchi, Nonlinear compressible vortex sheets in two space dimensions, Ann. Sci. Éc. Norm. Supér. (4) 41 (1) (2008) 85–139, https://doi.org/10.24033/asens.2064, http://www.numdam.org/item/ASENS_2008_4_41_1_85_0.
[30] W. Craig, C. Sulem, Numerical simulation of gravity waves, J. Comput. Phys. (ISSN 0021-9991) 108 (1) (1993) 73–83, https://doi.org/10.1006/jcph.1993.1164.
[31] C.M. Dafermos, Hyperbolic Conservation Laws in Continuum Physics, fourth edition, Grundlehren der Mathematischen Wissenschaften (Fundamental Principles of Mathematical Sciences), vol. 325, Springer-Verlag, Berlin, 2016, ISBN 978-3-662-49449-3; 978-3-662-49451-6.
[32] B.J. Daly, Numerical study of two fluid Rayleigh-Taylor instability, Phys. Fluids 10 (Feb. 1967) 297–307, https://doi.org/10.1063/1.1762109.
[33] P. Degond, S. Mas-Gallic, The weighted particle method for convection-diffusion equations. Part 1: The case of an isotropic viscosity, Math. Comput. 53 (188) (1989) 485–507, ISSN 00255718, 10886842, http://www.jstor.org/stable/2008716.
[34] J.-M. Delort, Existence de nappes de tourbillon en dimension deux, J. Am. Math. Soc. (ISSN 0894-0347) 4 (3) (1991) 553–586, https://doi.org/10.2307/2939269.
[35] I.V. Denisova, Global solvability of a problem on two fluid motion without surface tension, Zap. Nauchn. Sem. S.-Peterburg. Otdel. Mat. Inst. Steklov. (POMI) 348 (Kraevye Zadachi Matematicheskoĭ Fiziki i Smezhnye Voprosy Teorii Funktsiĭ 38) (2007) 19–39, https://doi.org/10.1007/s10958-008-9096-1, 303, ISSN 0373-2703.
[36] G. Dimonte, D.L. Youngs, A. Dimits, S. Weber, M. Marinak, S. Wunsch, C. Garasi, A. Robinson, M.J. Andrews, P. Ramaprabhu, A.C. Calder, B. Fryxell, J. Biello, L. Dursi, P. MacNeice, K. Olson, P. Ricker, R. Rosner, F. Timmes, H. Tufo, Y.-N. Young, M. Zingale, A comparative study of the turbulent Rayleigh-Taylor instability using high-resolution three-dimensional numerical simulations: the Alpha-Group collaboration, Phys. Fluids 16 (5) (2004) 1668–1693, https://doi.org/10.1063/1.1688328.
[37] C.I. Draghicescu, M. Draghicescu, A fast algorithm for vortex blob interactions, J. Comput. Phys. (ISSN 0021-9991) 116 (1) (1995) 69–78, https://doi.org/10.1006/jcph.1995.1006.
[38] D.G. Ebin, Ill-posedness of the Rayleigh-Taylor and Helmholtz problems for incompressible fluids, Commun. Partial Differ. Equ. (ISSN 0360-5302) 13 (10) (1988) 1265–1295, https://doi.org/10.1080/03605308808820576.
[39] L.C. Evans, S. Müller, Hardy spaces and the two-dimensional Euler equations with nonnegative vorticity, J. Am. Math. Soc. (ISSN 0894-0347) 7 (1) (1994) 199–219, https://doi.org/10.2307/2152727.
[40] J. Glimm, O. McBryan, R. Menikoff, D.H. Sharp, Front tracking applied to Rayleigh Taylor instability, SIAM J. Sci. Stat. Comput. (ISSN 0196-5204) 7 (1) (Jan. 1986) 230–251, https://doi.org/10.1137/0907016.
[41] J. Glimm, X.L. Li, R. Menikoff, D.H. Sharp, Q. Zhang, A numerical study of bubble interactions in Rayleigh-Taylor instability for compressible fluids, Phys. Fluids A, Fluid Dyn. 2 (11) (1990) 2046–2054, https://doi.org/10.1063/1.857679.
[42] V.N. Goncharov, Analytical model of nonlinear, single-mode, classical Rayleigh-Taylor instability at arbitrary Atwood numbers, Phys. Rev. Lett. 88 (Mar 2002) 134502, https://doi.org/10.1103/PhysRevLett.88.134502.
[43] J. Goodman, T.Y. Hou, J. Lowengrub, Convergence of the point vortex method for the 2-D Euler equations, Commun. Pure Appl. Math. (ISSN 0010-3640) 43 (3) (1990) 415–430, https://doi.org/10.1002/cpa.3160430305.
[44] R. Granero-Belinchón, S. Shkoller, A model for Rayleigh-Taylor mixing and interface turnover, Multiscale Model. Simul. (ISSN 1540-3459) 15 (1) (2017) 274–308, https://doi.org/10.1137/16M1083463.

[45] L. Greengard, V. Rokhlin, A fast algorithm for particle simulations, J. Comput. Phys. (ISSN 0021-9991) 73 (2) (1987) 325–348, https://doi.org/10.1016/0021-9991(87)90140-9.
[46] S.W. Haan, Weakly nonlinear hydrodynamic instabilities in inertial fusion, Phys. Fluids, B Plasma Phys. 3 (8) (1991) 2349–2355, https://doi.org/10.1063/1.859603.
[47] O.H. Hald, Convergence of vortex methods for Euler's equations. II, SIAM J. Numer. Anal. (ISSN 0036-1429) 16 (5) (1979) 726–755, https://doi.org/10.1137/0716055.
[48] J.T. Hamilton, G. Majda, On the Rokhlin-Greengard method with vortex blobs for problems posed in all space or periodic in one direction, J. Comput. Phys. (ISSN 0021-9991) 121 (1) (1995) 29–50, https://doi.org/10.1006/jcph.1995.1177.
[49] A. Harten, B. Engquist, S. Osher, S.R. Chakravarthy, Uniformly high order accurate essentially non-oscillatory schemes, III, J. Comput. Phys. 131 (Feb. 1997) 3–47, https://doi.org/10.1006/jcph.1996.5632.
[50] H. Helmholtz, XLIII. On discontinuous movements of fluids, Philos. Mag. 36 (244) (1868) 337–346, https://doi.org/10.1080/14786446808640073.
[51] J.J. Hester, J.M. Stone, P.A. Scowen, B.-I. Jun, J.S. Gallagher III, M.L. Norman, G.E. Ballester, C.J. Burrows, S. Casertano, J.T. Clarke, D. Crisp, R.E. Griffiths, J.G. Hoessel, J.A. Holtzman, J. Krist, J.R. Mould, R. Sankrit, K.R. Stapelfeldt, J.T. Trauger, A. Watson, J.A. Westphal, WFPC2 studies of the Crab Nebula. III. Magnetic Rayleigh-Taylor instabilities and the origin of the filaments, Astrophys. J. 456 (Jan. 1996) 225, https://doi.org/10.1086/176643.
[52] H.T. Huynh, Accurate upwind methods for the Euler equations, SIAM J. Numer. Anal. (ISSN 0036-1429) 32 (5) (1995) 1565–1619, https://doi.org/10.1137/0732071.
[53] G.-S. Jiang, C.-W. Shu, Efficient implementation of weighted ENO schemes, J. Comput. Phys. (ISSN 0021-9991) 126 (1) (1996) 202–228, https://doi.org/10.1006/jcph.1996.0130.
[54] V. Kamotski, G. Lebeau, On 2D Rayleigh-Taylor instabilities, Asymptot. Anal. (ISSN 0921-7134) 42 (1–2) (2005) 1–27.
[55] R. Krasny, A study of singularity formation in a vortex sheet by the point-vortex approximation, J. Fluid Mech. (ISSN 0022-1120) 167 (1986) 65–93, https://doi.org/10.1017/S0022112086002732.
[56] R. Krasny, Desingularization of periodic vortex sheet roll-up, J. Comput. Phys. 65 (Aug. 1986) 292–313, https://doi.org/10.1016/0021-9991(86)90210-X.
[57] R. Krasny, Computation of vortex sheet roll-up in the Trefftz plane, J. Fluid Mech. 184 (1987) 123–155, https://doi.org/10.1017/S0022112087002830.
[58] H. Kull, Theory of the Rayleigh-Taylor instability, Phys. Rep. (ISSN 0370-1573) 206 (5) (1991) 197–325, https://doi.org/10.1016/0370-1573(91)90153-D, http://www.sciencedirect.com/science/article/pii/037015739190153D.
[59] M. Latini, O. Schilling, W.S. Don, High-resolution simulations and modeling of reshocked single-mode Richtmyer-Meshkov instability: comparison to experimental data and to amplitude growth model predictions, Phys. Fluids 19 (2) (2007) 024104, https://doi.org/10.1063/1.2472508.
[60] P.D. Lax, Weak solutions of nonlinear hyperbolic equations and their numerical computation, Commun. Pure Appl. Math. (ISSN 0010-3640) 7 (1954) 159–193, https://doi.org/10.1002/cpa.3160070112.
[61] K. Lindsay, R. Krasny, A particle method and adaptive treecode for vortex sheet motion in three-dimensional flow, J. Comput. Phys. (ISSN 0021-9991) 172 (2) (2001) 879–907, https://doi.org/10.1006/jcph.2001.6862.
[62] R. Liska, B. Wendroff, Comparison of several difference schemes on 1d and 2d test problems for the Euler equations, SIAM J. Sci. Comput. 25 (3) (2003) 995–1017, https://doi.org/10.1137/S1064827502402120.
[63] J.-G. Liu, Z.P. Xin, Convergence of vortex methods for weak solutions to the 2-D Euler equations with vortex sheet data, Commun. Pure Appl. Math. (ISSN 0010-3640) 48 (6) (1995) 611–628, https://doi.org/10.1002/cpa.3160480603.
[64] X.-D. Liu, S. Osher, T. Chan, Weighted essentially non-oscillatory schemes, J. Comput. Phys. (ISSN 0021-9991) 115 (1) (1994) 200–212, https://doi.org/10.1006/jcph.1994.1187.
[65] M.C. Lopes Filho, H.J. Nussenzveig Lopes, Z. Xin, Existence of vortex sheets with reflection symmetry in two space dimensions, Arch. Ration. Mech. Anal. (ISSN 0003-9527) 158 (3) (2001) 235–257, https://doi.org/10.1007/s002050100145.
[66] M.C. Lopes Filho, J. Lowengrub, H.J. Nussenzveig Lopes, Y. Zheng, Numerical evidence of nonuniqueness in the evolution of vortex sheets, ESAIM: Math. Model. Numer. Anal. 40 (2) (2006) 225–237, https://doi.org/10.1051/m2an:2006012.
[67] Rayleigh Lord, Investigation of the character of the equilibrium of an incompressible heavy fluid of variable density, Proc. Lond. Math. Soc. s1–14 (1) (1882) 170–177, https://doi.org/10.1112/plms/s1-14.1.170.
[68] A.J. Majda, Remarks on weak solutions for vortex sheets with a distinguished sign, Indiana Univ. Math. J. (ISSN 0022-2518) 42 (3) (1993) 921–939, https://doi.org/10.1512/iumj.1993.42.42043.
[69] A.J. Majda, G. Majda, Y. Zheng, Concentrations in the one-dimensional Vlasov-Poisson equations I: Temporal development and non-unique weak solutions in the single component case, Physica D (ISSN 0167-2789) 74 (3) (1994) 268–300, https://doi.org/10.1016/0167-2789(94)90198-8, http://www.sciencedirect.com/science/article/pii/0167278994901988.
[70] C. Meneveau, P. Sagaut, Large Eddy Simulation for Incompressible Flows: An Introduction. Scientific Computation, Springer, Berlin, Heidelberg, ISBN 9783540264033, 2006, https://books.google.com/books?id=SH90vyraAT0C.
[71] E.E. Meshkov, Instability of the interface of two gases accelerated by a shock wave, Fluid Dyn. (ISSN 1573-8507) 4 (5) (Sep 1969) 101–104, https://doi.org/10.1007/BF01015969.
[72] K. Mohseni, B. Kosović, S. Shkoller, J.E. Marsden, Numerical simulations of the Lagrangian averaged Navier-Stokes equations for homogeneous isotropic turbulence, Phys. Fluids 15 (2) (2003) 524–544, https://doi.org/10.1063/1.1533069.
[73] P. Moin, K. Mahesh, Direct numerical simulation: a tool in turbulence research, Annu. Rev. Fluid Mech. 30 (1) (1998) 539–578, https://doi.org/10.1146/annurev.fluid.30.1.539.
[74] T. Nonomura, S. Morizawa, H. Terashima, S. Obayashi, K. Fujii, Numerical (error) issues on compressible multicomponent flows using a high-order differencing scheme: weighted compact nonlinear scheme, J. Comput. Phys. (ISSN 0021-9991) 231 (8) (2012) 3181–3210, https://doi.org/10.1016/j.jcp.2011.12.035, http://www.sciencedirect.com/science/article/pii/S0021999112000022.
[75] R. Nourgaliev, T. Dinh, T. Theofanous, Adaptive characteristics-based matching for compressible multifluid dynamics, J. Comput. Phys. (ISSN 0021-9991) 213 (2) (2006) 500–529, https://doi.org/10.1016/j.jcp.2005.08.028, http://www.sciencedirect.com/science/article/pii/S0021999105003906.
[76] D.I. Pullin, On similarity flows containing two-branched vortex sheets, in: R.E. Caflisch (Ed.), Mathematical Aspects of Vortex Dynamics, 1989, pp. 97–106.
[77] R. Ramani, J. Reisner, S. Shkoller, A space-time smooth artificial viscosity method with wavelet noise indicator and shock collision scheme, Part 1: The 1-D case, J. Comput. Phys. (ISSN 0021-9991) 387 (2019) 81–116, https://doi.org/10.1016/j.jcp.2019.02.049, http://www.sciencedirect.com/science/article/pii/S0021999119301664.
[78] R. Ramani, J. Reisner, S. Shkoller, A space-time smooth artificial viscosity method with wavelet noise indicator and shock collision scheme, Part 2: The 2-D case, J. Comput. Phys. (ISSN 0021-9991) 387 (2019) 45–80, https://doi.org/10.1016/j.jcp.2019.02.048, http://www.sciencedirect.com/science/article/pii/S0021999119301652.
[79] K. Read, Experimental investigation of turbulent mixing by Rayleigh-Taylor instability, Physica D (ISSN 0167-2789) 12 (1) (1984) 45–58, https://doi.org/10.1016/0167-2789(84)90513-X, http://www.sciencedirect.com/science/article/pii/016727898490513X.
[80] J. Reisner, J. Serencsa, S. Shkoller, A space-time smooth artificial viscosity method for nonlinear conservation laws, J. Comput. Phys. (ISSN 0021-9991) 235 (2013) 912–933, https://doi.org/10.1016/j.jcp.2012.08.027.
[81] R.D. Richtmyer, Taylor instability in shock acceleration of compressible fluids, Commun. Pure Appl. Math. (ISSN 0010-3640) 13 (1960) 297–319, https://doi.org/10.1002/cpa.3160130207.

[82] L. Rosenhead, The formation of vortices from a surface of discontinuity, Proc. R. Soc. Lond. Ser. A, Math. Phys. Sci. 134 (Nov. 1931) 170–192, https://doi.org/10.1098/rspa.1931.0189.
[83] N. Rott, Diffraction of a weak shock with vortex generation, J. Fluid Mech. (ISSN 0022-1120) 1 (1956) 111, https://doi.org/10.1017/S0022112056000081.
[84] T. Sakajo, H. Okamoto, An application of Draghicescu's fast summation method to vortex sheet motion, J. Phys. Soc. Jpn. 67 (2) (1998) 462–470, https://doi.org/10.1143/JPSJ.67.462.
[85] D. Sharp, An overview of Rayleigh-Taylor instability, Physica D (ISSN 0167-2789) 12 (1) (1984) 3–18, https://doi.org/10.1016/0167-2789(84)90510-4, http://www.sciencedirect.com/science/article/pii/0167278984905104.
[86] C.-W. Shu, High-order finite difference and finite volume WENO schemes and discontinuous Galerkin methods for CFD, Int. J. Comput. Fluid Dyn. (ISSN 1061-8562) 17 (2) (2003) 107–118, https://doi.org/10.1080/1061856031000104851.
[87] C.-W. Shu, S. Osher, Efficient implementation of essentially nonoscillatory shock-capturing schemes, J. Comput. Phys. (ISSN 0021-9991) 77 (2) (1988) 439–471, https://doi.org/10.1016/0021-9991(88)90177-5.
[88] C.-W. Shu, S. Osher, Efficient implementation of essentially nonoscillatory shock-capturing schemes. II, J. Comput. Phys. (ISSN 0021-9991) 83 (1) (1989) 32–78, https://doi.org/10.1016/0021-9991(89)90222-2.
[89] William F.R.S. Thomson, XLVI. Hydrokinetic solutions and observations, Philos. Mag. 42 (281) (1871) 362–377, https://doi.org/10.1080/14786447108640585.
[90] W.D. Smyth, Ocean mixing by Kelvin-Helmholtz instability, Oceanography 25 (June 2012), https://doi.org/10.5670/oceanog.2012.49.
[91] S.-I. Sohn, Simple potential-flow model of Rayleigh-Taylor and Richtmyer-Meshkov instabilities for all density ratios, Phys. Rev. E 67 (Feb 2003) 026301, https://doi.org/10.1103/PhysRevE.67.026301.
[92] S.-I. Sohn, Vortex model and simulations for Rayleigh-Taylor and Richtmyer-Meshkov instabilities, Phys. Rev. E 69 (Mar 2004) 036703, https://doi.org/10.1103/PhysRevE.69.036703.
[93] S.-I. Sohn, Two vortex-blob regularization models for vortex sheet motion, Phys. Fluids 26 (4) (2014) 044105, https://doi.org/10.1063/1.4872027.
[94] L. Székelyhidi, Weak solutions to the incompressible Euler equations with vortex sheet initial data, C. R. Math. Acad. Sci. Paris (ISSN 1631-073X) 349 (19–20) (2011) 1063–1066, https://doi.org/10.1016/j.crma.2011.09.009.
[95] G.I. Taylor, The spectrum of turbulence, Proc. R. Soc. Lond. Ser. A, Math. Phys. Sci. 164 (Feb. 1938) 476–490, https://doi.org/10.1098/rspa.1938.0032.
[96] G.I. Taylor, The instability of liquid surfaces when accelerated in a direction perpendicular to their planes. I, Proc. R. Soc. Lond. Ser. A, Math. Phys. Sci. 201 (1065) (1950) 192–196, https://doi.org/10.1098/rspa.1950.0052.
[97] H. Terashima, G. Tryggvason, A front-tracking/ghost-fluid method for fluid interfaces in compressible flows, J. Comput. Phys. (ISSN 0021-9991) 228 (11) (2009) 4012–4037, https://doi.org/10.1016/j.jcp.2009.02.023, http://www.sciencedirect.com/science/article/pii/S0021999109000898.
[98] G. Tryggvason, Numerical simulations of the Rayleigh-Taylor instability, J. Comput. Phys. (ISSN 0021-9991) 75 (2) (1988) 253–282, https://doi.org/10.1016/0021-9991(88)90112-X, http://www.sciencedirect.com/science/article/pii/002199918890112X.
[99] G. Tryggvason, Simulation of vortex sheet roll-up by vortex methods, J. Comput. Phys. (ISSN 0021-9991) 80 (1) (1989) 1–16, https://doi.org/10.1016/0021-9991(89)90087-9, http://www.sciencedirect.com/science/article/pii/0021999189900879.
[100] L. van Dommelen, E.A. Rundensteiner, Fast, adaptive summation of point forces in the two-dimensional Poisson equation, J. Comput. Phys. (ISSN 0021-9991) 83 (1) (1989) 126–147, https://doi.org/10.1016/0021-9991(89)90225-8, http://www.sciencedirect.com/science/article/pii/0021999189902258.
[101] B. van Leer, Towards the ultimate conservative difference scheme. V. A second-order sequel to Godunov's method, J. Comput. Phys. (ISSN 0021-9991) 32 (1) (1979) 101–136, https://doi.org/10.1016/0021-9991(79)90145-1, http://www.sciencedirect.com/science/article/pii/0021999179901451.
[102] H. Versteeg, W. Malalasekera, An Introduction to Computational Fluid Dynamics: The Finite Volume Method, Pearson Education Limited, ISBN 9780131274983, 2007, https://books.google.com/books?id=RvBZ-UMpGzIC.
[103] J. Von Neumann, R.D. Richtmyer, A method for the numerical calculation of hydrodynamic shocks, J. Appl. Phys. 21 (1950) 232–237.
[104] J.T. Waddell, C.E. Niederhaus, J.W. Jacobs, Experimental study of Rayleigh-Taylor instability: low Atwood number liquid systems with single-mode initial perturbations, Phys. Fluids 13 (5) (2001) 1263–1273, https://doi.org/10.1063/1.1359762.
[105] M.L. Wong, S.K. Lele, High-order localized dissipation weighted compact nonlinear scheme for shock- and interface-capturing in compressible flows, J. Comput. Phys. (ISSN 0021-9991) 339 (2017) 179–209, https://doi.org/10.1016/j.jcp.2017.03.008, http://www.sciencedirect.com/science/article/pii/S002199911730195X.
[106] D.L. Youngs, Numerical simulation of turbulent mixing by Rayleigh-Taylor instability, Physica D (ISSN 0167-2789) 12 (1) (1984) 32–44, https://doi.org/10.1016/0167-2789(84)90512-8, http://www.sciencedirect.com/science/article/pii/0167278984905128.
[107] D.L. Youngs, Modelling turbulent mixing by Rayleigh-Taylor instability, Physica D (ISSN 0167-2789) 37 (1) (1989) 270–287, https://doi.org/10.1016/0167-2789(89)90135-8, http://www.sciencedirect.com/science/article/pii/0167278989901358.
[108] Y. Zhou, Rayleigh-Taylor and Richtmyer-Meshkov instability induced flow, turbulence, and mixing. I, Phys. Rep. (ISSN 0370-1573) 720–722 (2017) 1–136, https://doi.org/10.1016/j.physrep.2017.07.005.

CHAPTER 3

The SAM-ALE scheme

This chapter was published in Journal of Computational Physics, Vol 490, R. Ramani & S. Shkoller, A fast dynamic smooth adaptive meshing scheme with applications to compressible flow, 112280, Copyright Elsevier (2023).

Journal of Computational Physics 490 (2023) 112280

Contents lists available at ScienceDirect

Journal of Computational Physics

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

A fast dynamic smooth adaptive meshing scheme with applications to compressible flow

Raaghav Ramani *, Steve Shkoller

Department of Mathematics, University of California, Davis, CA 95616, USA

ARTICLE INFO

Article history:
Received 13 June 2022
Received in revised form 24 March 2023
Accepted 5 June 2023
Available online 14 June 2023

Keywords:
Mesh adaptation
Moving meshes
Monge-Ampère
Compressible flows
WENO
Rayleigh-Taylor

ABSTRACT

We develop a fast-running smooth adaptive meshing (SAM) algorithm for dynamic curvilinear mesh generation, which is based on a fast solution strategy of the time-dependent Monge-Ampère (MA) equation, $\det \nabla \psi(x,t) = \mathsf{G} \circ \psi(x,t)$. The novelty of our approach is a new so-called *perturbation formulation* of MA, which constructs the solution map ψ via composition of a sequence of near-identity deformations of a reference mesh. Then, we formulate a new version of the deformation method [21] that results in a simple, fast, and high-order accurate numerical scheme and a dynamic SAM algorithm that is of optimal complexity when applied to time-dependent mesh generation for solutions to hyperbolic systems such as the Euler equations of gas dynamics. We perform a series of challenging $2D$ and $3D$ mesh generation experiments for grids with large deformations, and demonstrate that SAM is able to produce smooth meshes comparable to state-of-the-art solvers [22,18], while running approximately 200 times faster. The SAM algorithm is then coupled to a simple Arbitrary Lagrangian Eulerian (ALE) scheme for $2D$ gas dynamics. Specifically, we implement the C-method [64,65] and develop a new ALE interface tracking algorithm for contact discontinuities. We perform numerical experiments for both the Noh implosion problem as well as a classical Rayleigh-Taylor instability problem. Results confirm that low-resolution simulations using our SAM-ALE algorithm compare favorably with high-resolution uniform mesh runs.

Contents

* Corresponding author.
E-mail addresses: rramani@math.ucdavis.edu (R. Ramani), shkoller@math.ucdavis.edu (S. Shkoller).

https://doi.org/10.1016/j.jcp.2023.112280

1. Introduction

The efficiency of smooth moving-mesh methods for numerical simulations of gas dynamics[1] and related systems has been investigated in recent years [3,85,35,36,62,58,25,49,26]; however, to the best of our knowledge, compelling evidence of the gain in efficiency relative to fixed uniform-mesh simulations in multiple space dimensions has rarely been provided. In a recent result [49], the authors demonstrate that low-resolution adaptive simulations are roughly 2-6 times faster than high-resolution uniform simulations of comparable quality; most results in this area focus on novel solution methodologies but not on the ultimate speed-up that may be gained by the algorithms that they produce. The papers cited above focus on one half of the moving-mesh methodology, namely, the numerical discretization of the physical PDEs. They develop state-of-the-art high-resolution shock-capturing techniques, but use well-established and somewhat standard meshing algorithms. Our point-of-view is that it is essential to simultaneously develop both numerical methods for hyperbolic systems (for discontinuous solutions) as well as novel meshing strategies.[2]

Herein, we propose a novel and fast[3] Smooth Adaptive Meshing (SAM) algorithm for multi-D simulations requiring mesh adaptivity. We present adaptive-simulation speed-up results for two classical but extremely challenging gas dynamic problems: the Noh shock implosion, and the (highly unstable) Rayleigh-Taylor (RT) test. For the Noh problem, our adaptive simulations are free of the numerical anomalies that are present in almost all reported results, while running approximately 6 times faster than a comparable uniform-mesh simulation. The ten-fold speed-up provided by SAM for the RT problem is, to the best of our knowledge, the first of its kind.[4]

[1] Moving-mesh simulations are often referred to as *adaptive simulations* and we shall use this terminology herein.

[2] This philosophy is in agreement with [62], in which the authors state that the main obstacle in their moving-mesh simulations is the lack of a simple, robust, and efficient algorithm for dynamic and smooth adaptive mesh generation, particularly in 3D geometries, and for multi-phase flows with unstable interfaces.

[3] We will demonstrate that our SAM algorithm is the first to be able to solve classical Rayleigh-Taylor problems on coarse, but adaptive, grids faster than simulations on uniform grids.

[4] Most attempts at using moving-mesh adaptivity to numerically simulate the RT instability result in runs that prematurely blow-up due to mesh tangling, meaning that those algorithms are not sufficiently stable to provide a competitive speed-up factor. Recent papers [57,4] instead focus on novel and sophisticated meshing techniques with the goal of simply simulating the RT instability until the final simulation time without the code crashing; however, these meshing algorithms are currently too expensive to provide speed-up over uniform-mesh simulations.

1.1. Mesh refinement for multi-D gas dynamics

It is by now well-known that static uniform meshes are both inaccurate and inefficient at representing the dynamically evolving and interacting small-scale structures that appear in solutions to nonlinear conservation laws in multiple space dimensions. Adaptive mesh refinement (AMR) via h-adaptivity is the most well-developed refinement technique and is used in many commercial codes [70,10,29,30]. However, the dyadic refinement at the heart of AMR schemes results in an artificially discontinuous transition from coarse-scale to fine-scale representation of numerical solutions on AMR meshes. Several theoretical and numerical studies [5,78,56] have demonstrated the spurious reflection, refraction, and scattering of waves that propagate across discontinuously refined grids. Many problems in gas dynamics, such as strong blast waves, self-similar implosions, and unstable contact discontinuities are extremely sensitive to small perturbations; spurious wave reflections produce corrupted numerical solutions, with the anomalies persisting, or even worsening, as the AMR mesh is globally refined [29,76].

On the other hand, Lagrangian-type schemes are well-known to produce highly distorted or tangled meshes i.e. some cells in the grid are non-convex or have folded over, at which point the simulation breaks down. Arbitrary Lagrangian Eulerian (ALE) methods aim to mitigate the problem of mesh tangling. *Indirect* ALE methods are somewhat *ad hoc*, and current rezoning strategies are heuristic in nature [46,57]. In this work, we consider the *direct* ALE approach, in which an adaptive mesh is generated directly without any initial Lagrangian phase or subsequent mesh rezoning.

1.1.1. Adaptive mesh redistribution

Our SAM scheme falls under the category of r-refinement schemes, or adaptive mesh redistribution methods. In contrast to Lagrangian-rezone methods, a grid is generated via a user-prescribed monitor function which determines the grid size and orientation. High-resolution representation of numerical solutions is obtained by defining the monitor function appropriately, e.g., using solution derivatives. Moreover, the adaptive grids can be generated to align with the geometry of evolving fronts [39], and to naturally capture self-similar dynamics or scale-invariant structures [13,11].

Historically, the first r-refinement methods were based on the variational approach, examples of which include the equipotential [82], variable diffusion [83], cost function [6], and harmonic mapping [27] methods. The variational approach also currently appears to be the method of choice for use in direct ALE schemes, several of which employ the popular MMPDE framework [40,47]. These variational methods, however, require the accurate numerical solution of a coupled set of d complicated nonlinear auxiliary PDEs in $\mathbb{R}^d$, for which simple, fast, and accurate algorithms are in general not available. For these reasons, among others, r-refinement methods have yet to become incorporated into large scale established hydrodynamics codes. See, for example, [41,14,22,18] and the references therein for thorough reviews of r-adaptive methods and their associated difficulties.

1.1.2. Prescribing the Jacobian determinant

The fundamental guiding principle for smooth adaptive mesh generation is control of the local cell volume of the adaptive grid. In the time-dependent multi-D setting, we assume that we have a given smooth positive *target Jacobian function* $\mathsf{G}(y,t)$ describing the size of the cells in the moving target adaptive mesh. We then seek to construct a diffeomorphism $\psi(x,t)$ mapping a fixed reference mesh to the target mesh by requiring that $\det\nabla\psi(x,t) = \mathsf{G}(\psi(x,t),t)$. A semi-discretization in time $t = t_k$, where k is the time-index, yields a sequence of nonlinear elliptic equations of Monge-Ampère (MA) type

$$\det\nabla\psi_k(x) = \mathsf{G}_k(\psi_k(x))\,, \tag{1}$$

where each G_k is again a given positive target Jacobian function.

Solutions to the MA equation are unique in $1D$. For dimension $d \geq 2$, however, the single scalar MA equation is insufficient to uniquely determine ψ. The question then becomes how to choose a particular solution ψ that is in some sense optimal. One such choice that has received a great deal of attention in recent years is the Monge-Kantorovich (MK) formulation based on optimal transport, in which a map ψ is (uniquely [8,16]) constructed to minimize the L^2 displacement $||\psi(x)-x||_{L^2}$. This is attractive from a numerical perspective, since smaller grid velocities can reduce interpolation and other numerical errors [48].

On the other hand, the MK formulation results in a fully nonlinear second order elliptic equation, whose numerical solution is difficult to obtain. One approach is to consider a parabolized formulation by introducing an artificial time variable τ then iterating until a steady state is reached [72,9,59,81]. In this case, the Jacobian constraint is only satisfied in the asymptotic limit $\tau \to \infty$, and many iterations may be required to obtain a sufficiently accurate solution, particularly for target meshes with large deformations. An alternative, fully nonlinear approach using preconditioned Newton-Krylov solvers is designed in [22,18], leading to a robust, scalable algorithm that is, to the best of our knowledge, the state-of-the-art in the field (see also the recent papers [12,15]). However, the Newton-Krylov iterative approach is still relatively slow for our ultimate goal of efficient adaptive gas dynamics simulations; specifically, its implementation in our ALE scheme (to be described below) leads to adaptive mesh simulations with computational runtimes greater than would otherwise be

obtained with a uniformly high-resolution mesh, thereby defeating the purpose of using an adaptive meshing scheme in the first place.

1.2. Fast smooth adaptive meshing

In contrast to the MK approach, we construct a map ψ_k satisfying (1) with the aim of optimizing for the efficiency of the resulting numerical algorithm, which we refer to as SAM. The key to our fast SAM algorithm is a new *perturbation formulation* of (1) along with a new formulation and implementation of the deformation method [21].

Specifically, the perturbation formulation constructs each map ψ_{k+1} as the image of the map ψ_k acting on a *near identity deformation* $\delta\psi_{k+1} \approx \mathrm{id}$ of a fixed reference mesh Ω_{ref}. The formulation on Ω_{ref} is crucial, since it enables the use of, at each time-step t_k, the same numerical solvers for the mesh PDEs.[5] This, in turn, produces a code with a simple modular structure so that the basic mesh redistribution procedure is developed entirely in the static setting on Ω_{ref}, then "bootstrapped" to form a dynamic scheme. The same principle also yields an algorithm for efficiently generating smooth meshes with very large zoom-in factors, which allows us to obtain high-resolution representation of small-scale structures with few total number of mesh points.

The mesh redistribution algorithm we propose is a new version of the deformation method [50,54,51], which constructs a solution to the nonlinear Monge-Ampère equation via a single elliptic solve for a linear Poisson problem, along with the solution of a system of transport equations for a flowmap $\eta(x,\tau)$ between pseudo-time $\tau = 0$ and $\tau = 1$. There are at least two advantages of this new deformation method: the first is that the algorithm can be made fully automated with no user-prescribed parameters; the second is that costly and often complicated interpolation procedures are not required. We design a simple, fast, stable, and high-order accurate method using an efficient spectral solver with boundary smoothing for the Poisson equation, and standard RK4 time integration with high-order linear upwind differencing for the transport equations. A key implication of our numerical design choices is a consistency between the stability conditions for the transport problem in SAM and the physical time-step in an ALE gas dynamics simulation. As we shall demonstrate, this consistency results in a dynamic SAM algorithm with optimal complexity for hyperbolic systems.

Our SAM algorithm is approximately 200 times faster than the MK nonlinear solvers [22,18], and the computed numerical solutions exhibit both higher accuracy as well as better convergence rates under global mesh refinement. We perform a number of challenging mesh generation experiments designed to replicate flows with high vorticity and large deformations, and demonstrate that the meshes produced with our dynamic SAM scheme are smooth and accurate. For example, we are able to generate smooth moving meshes that resolve around a complex $3D$ swirling helical-type curve at 256^3 resolution with only a serial implementation on a laptop computer and without any specific and sophisticated algorithmic optimizations (see Section 5.5).

1.3. Application to ALE gas dynamics

To demonstrate the efficacy of our SAM scheme in practical applications, we formulate a simple coupled SAM-ALE method for $2D$ gas dynamics. Several moving-mesh methods for the $2D$ Euler system have been developed based on the MMPDE approach and finite volume (FV) and finite element (FE) methods [73,74]. A formulation on smooth tensor product meshes enables the use of finite difference (FD) methods, which are both simpler and more efficient than FV and FE methods,[6] and have been investigated in several recent papers [60,45,49]. In this work, we further develop the C-method [64,65], a simplified WENO-based solver with space-time smooth nonlinear artificial viscosity and explicit tracking of material interfaces.

Special care is given to the so-called *geometric conservation law* (GCL), and we show that our nonlinear WENO reconstruction procedure respects the free-stream preservation property on adaptive meshes. The C-method dynamically tracks the location and geometry of evolving fronts, and is used to add both directionally isotropic and anisotropic artificial viscosity to shocks and contacts. Herein, we implement the C-method in the ALE context and introduce a new ALE front-tracking algorithm for contact discontinuities, which we subsequently use to construct suitable target Jacobian functions for SAM. Previous studies have mainly investigated target Jacobian functions constructed based on interpolation errors [42,39], or weighted combinations of solution gradient estimates [77], which sometimes fail to capture small scale vortical structures [73]. Our simple ALE front-tracking algorithm allows us to generate smooth adaptive meshes that capture small scale Kelvin-Helmholtz roll-up zones in unstable RT problems. We apply our coupled SAM-ALE scheme to two challenging test problems, namely the Noh implosion and RT instability. For the Noh problem, we find that the 50×50 SAM-ALE solution is more accurate than the 200×200 uniform solution, while running approximately 6 times faster. Moreover, the SAM-ALE solution is completely free of spurious numerical anomalies, such as lack of symmetry, unphysical oscillations, and wall-heating. For the RT problem, we find that the 64×128 SAM-ALE solution is comparable to the 256×512 uniform solution, while running 10 times faster.

[5] This is in contrast with other methods [67,32] which require finite-element solvers with costly recalculation (at each time-step of a dynamic simulation) of the mass and stiffness matrices, as well as complicated interpolation procedures.

[6] FV schemes are 4 times more expensive than FD schemes in $2D$, and 9 times more expensive in $3D$ [80].

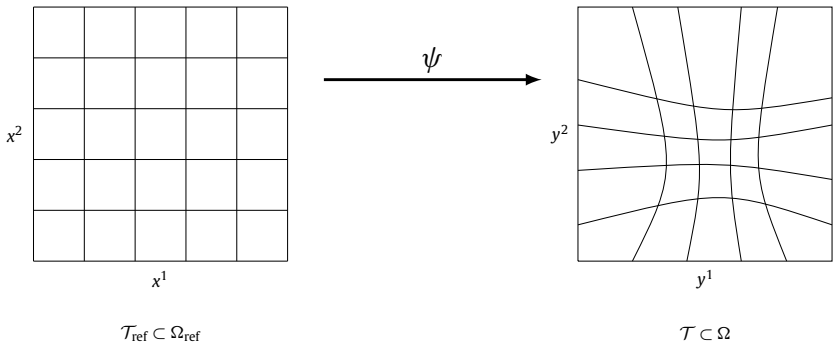


Fig. 1. The uniform $m \times n$ mesh $\mathcal{T}_{\text{ref}}$ and the adaptive $m \times n$ mesh $\mathcal{T} = \psi(\mathcal{T}_{\text{ref}})$.

1.4. Outline

Section 2 introduces notation and definitions that will be used throughout the paper. In Section 3, we develop the basic SAM algorithm for static mesh generation, upon which we shall build our dynamic scheme. We show that our scheme is high-order accurate and benchmark the algorithm against the MK scheme. In Section 4, we consider dynamic mesh generation and introduce the perturbation formulation of the MA system. We then perform, in Section 5, a series of challenging mesh generation experiments to demonstrate the capabilities of the scheme. In Section 6, we formulate a simple coupled SAM-ALE scheme for the $2D$ compressible Euler system, and describe some aspects of our numerical method. In Section 7, we apply SAM-ALE to the Noh and RT test problems and compare the results with low-resolution and high-resolution uniform solutions. Finally, in Section 8, we provide some brief concluding remarks. Three sections are included in the Appendices: the first concerns the C-method regularization for the $2D$ ALE-Euler system, the second describes a simple boundary smoothing technique, and the third provides a machine comparison test for the purposes of benchmarking our SAM algorithm.

2. Preliminaries

2.1. Domains, meshes, and mappings

The focus of this work is mesh adaptation on $2D$ rectangles and we provide the mathematical formulation and numerical implementation details of our mesh adaptation strategy in this setting. However, all of our meshing algorithms can be extended to $3D$ cuboids,[7] and we show in Section 5.5 results from a mesh generation experiment modeling three-dimensional swirling flow.

Let $\Omega_{\text{ref}} \subset \mathbb{R}^2$ be a *reference* domain with coordinates $x = (x^1, x^2) \in \Omega_{\text{ref}}$, and given explicitly by the rectangle $\Omega_{\text{ref}} = (x^1_{\min}, x^1_{\max}) \times (x^2_{\min}, x^2_{\max})$. The outward pointing unit normal vector to the boundary $\partial\Omega_{\text{ref}}$ is defined everywhere on $\partial\Omega_{\text{ref}}$, except at the four corners, and is denoted by ν. The domain Ω_{ref} is also sometimes referred to in the literature as the *logical* or *computational* domain, and in the context of ALE gas dynamics, the *ALE* domain.

We denote by $\Omega \subset \mathbb{R}^2$ the *physical* or *Eulerian* domain, with coordinates $y = (y^1, y^2) \in \Omega$ and boundary $\partial\Omega$. We assume that Ω_{ref} and Ω represent the same mathematical domain i.e. $\Omega_{\text{ref}} = \Omega$. The purpose of using the different notations Ω_{ref} and Ω is to clearly distinguish between functions defined on each of these domains, as we shall explain in the next subsection. We let $\text{id} : \Omega_{\text{ref}} \to \Omega$ denote the identity map, i.e. $\text{id}(x) = x$.

We discretize Ω_{ref} and Ω with $m+1$ nodes in the horizontal direction, and $n+1$ nodes in the vertical direction, and denote by $\mathcal{T}_{\text{ref}}$ and $\mathcal{T}$ the grids (or meshes) on each of these domains. Each of these meshes contains $N = m \times n$ cells. The domain Ω_{ref} is discretized uniformly, and we refer to $\mathcal{T}_{\text{ref}}$ as the *reference* or *uniform* mesh. The *physical* or *adaptive* mesh $\mathcal{T}$ is *a priori* unknown and will be generated through a meshing scheme. The mesh $\mathcal{T}$ is not assumed to be uniform, but contains the same number of cells and retains the same mesh connectivity structure as the uniform mesh $\mathcal{T}_{\text{ref}}$ cf. Fig. 1. The fixed uniform mesh spacing is denoted by $\Delta x = (\Delta x^1, \Delta x^2)$.

The physical domain Ω can also be discretized uniformly with a uniform mesh $\mathcal{U}$. Since $\Omega_{\text{ref}} = \Omega$, the meshes $\mathcal{U}$ and $\mathcal{T}_{\text{ref}}$ are identical. We stress, however, that functions defined on each of these meshes are very different.

[7] In fact, our algorithms can also be applied in arbitrary complex geometry (see Fig. 19 for a preliminary result), though their numerical implementations are more involved.

The mesh $\mathcal{T}$ will be the image of $\mathcal{T}_{\text{ref}}$ under the action of a suitable map $\psi : \Omega_{\text{ref}} \to \Omega$. The map ψ is bijective, continuously differentiable, and has a continuously differentiable inverse $\psi^{-1} : \Omega \to \Omega_{\text{ref}}$ i.e. ψ is a smooth diffeomorphism. Our SAM scheme solves for the map ψ by prescribing its Jacobian determinant, as we shall explain in Sections 3 and 4. Nodes in $\mathcal{T}$ on the boundary $\partial\Omega$ will be allowed to move tangential to the boundary, with the exception of the four nodes at the corners of Ω, which must remain fixed.

In the dynamic setting, we consider ψ to be a time-dependent map $\psi : \Omega_{\text{ref}} \times [0, T] \to \Omega$ where, for each $t \in [0, T]$, the map $\psi(\cdot, t) : \Omega_{\text{ref}} \to \Omega$ is a smooth diffeomorphism with prescribed Jacobian cf. Fig. 3.

2.2. Eulerian and ALE variables

A *physical* or *Eulerian* function (scalar, vector-valued, or tensor) is defined on Ω and denoted with the upright mathematical font $\mathsf{f} : \Omega \to \mathbb{R}^k$. For a time-dependent function $\mathsf{f} : \Omega \times [0, T] \to \mathbb{R}^k$ we shall write $\mathsf{f}(y, t)$.

Since ψ maps Ω_{ref} to Ω, we write $y = \psi(x, t)$ for $(x, t) \in \Omega_{\text{ref}} \times [0, T]$. Given an Eulerian variable $\mathsf{f} : \Omega \to \mathbb{R}^k$, we define its *computational* or *ALE* counterpart $f : \Omega_{\text{ref}} \times [0, T] \to \mathbb{R}^k$ by

$$f(x, t) = [\mathsf{f} \circ \psi](x, t) = \mathsf{f}(\psi(x, t), t), \quad \forall (x, t) \in \Omega_{\text{ref}} \times [0, T]. \tag{2}$$

We shall also denote the function composition in (2) by $\mathsf{f} \circ \psi$. When there is no confusion, we omit the function arguments and write f or f.

In the discrete setting, computational variables are defined at the nodal points of the uniform reference mesh $\mathcal{T}_{\text{ref}}$. Physical/Eulerian variables, on the other hand, can be defined on either the adaptive mesh $\mathcal{T}(t)$ or the uniform mesh $\mathcal{U}$ on Ω.

2.3. Derivatives and important geometric quantities

We denote spatial derivatives on Ω_{ref} and Ω by

$$\partial_i = \frac{\partial}{\partial x_i} \quad \text{and} \quad D_i = \frac{\partial}{\partial y_i},$$

respectively. Higher order derivatives are then denoted in the standard fashion, e.g. $\partial_{ij} = \partial_i \partial_j$. We use the notation $\nabla = (\partial_1, \partial_2)^{\mathcal{T}}$ and $D = (D_1, D_2)^{\mathcal{T}}$ for the gradient operators with respect to x and y coordinates, respectively. The Laplacian operator on Ω_{ref} is $\Delta = (\partial_1^2 + \partial_2^2)$. The operator Δ should not be confused with the discrete uniform mesh spacing Δx^i.

The time derivative of a function f is written as $\partial_t f$, or sometimes with the subscript notation f_t. Throughout, we shall use Einstein's summation convention wherein a repeated index in the same term indicates summation over all values of that index. We shall also use the standard Kronecker delta symbol δ^i_j.

We now introduce the following important geometric quantities, all defined on $\Omega_{\text{ref}} \times [0, T]$:

$$\mathcal{A} = [\nabla \psi]^{-1} \quad \text{(inverse of the deformation tensor)}, \tag{3a}$$

$$\mathcal{J} = \det \nabla \psi \quad \text{(Jacobian determinant)}, \tag{3b}$$

$$a = \mathcal{J}\mathcal{A} \quad \text{(cofactor matrix of the deformation tensor)}. \tag{3c}$$

We assume that there exists $\varepsilon > 0$ such that

$$\mathcal{J}(x, t) \geq \varepsilon > 0, \text{ for every } (x, t) \in \Omega_{\text{ref}} \times [0, T].$$

Thus, the Jacobian determinant in $2D$ reads

$$\mathcal{J}(x, t) = \partial_1 \psi^1 \, \partial_2 \psi^2 - \partial_1 \psi^2 \, \partial_2 \psi^1.$$

For a matrix $M = (M^j_i)$, the subscript i indexes the columns of M, while the superscript j indexes the rows.

By explicit computation, we can verify the so-called Piola identity, which states that the columns of the cofactor matrix are divergence-free:

$$\partial_j a^j_i = 0, \text{ for } i = 1, 2. \tag{4}$$

Given an Eulerian variable $\mathsf{f}(y, t)$ and its ALE counterpart $f(x, t)$, we use the chain rule to compute

$$D_i \mathsf{f}(y, t) = \frac{1}{\mathcal{J}(x, t)} a^j_i(x, t) \, \partial_j f(x, t) = \frac{1}{\mathcal{J}} \partial_j (a^j_i f), \tag{5}$$

where we have used the Piola identity (4) in the second equality. Using (5) and the chain rule again, we have that

$$\partial_t \mathsf{f}(y,t) = \partial_t f - \frac{1}{\mathcal{J}} a_i^j \psi_t^i \, \partial_j f \,, \tag{6}$$

where $\psi_t(x,t) \equiv \partial_t \psi(x,t)$ is the *mesh velocity*.

2.4. Computational platform and code optimization

All of the algorithms in this work were coded in Fortran90, and all of the numerical simulations performed were run on a Macbook Pro laptop with an Apple M1 pro processor and 32 GB of RAM. The operating system is macOS Ventura 13.1, and the gfortran compiler is used. The codes for the numerical methods described in the paper are implemented in the same programming framework, but are not otherwise specially optimized, apart from specific calculations described in the paper. The same input, output, and timing routines are used in all of the codes. This consistency allows for a reliable comparison of the different algorithms and their associated imposed computational burdens.

3. Fast static adaptive meshing

3.1. Mathematical formulation of static mesh generation

We construct an adaptive mesh $\mathcal{T}$ as the image of the uniform mesh $\mathcal{T}_{\text{ref}}$ under the action of a suitable smooth diffeomorphism $\psi : \Omega_{\text{ref}} \to \Omega$ cf. Fig. 1. Our objective is to compute the map ψ by prescribing its Jacobian determinant $\mathcal{J}(x) = \det \nabla \psi(x)$. Specifically, given a strictly positive *target Jacobian function* $\mathsf{G} : \Omega \to \mathbb{R}^+$, the map ψ is found as a solution to the following nonlinear nonlocal Monge-Ampère (MA) equation

$$\begin{cases} \det \nabla \psi(x) = \mathsf{G} \circ \psi(x) \,, & x \in \Omega_{\text{ref}} \quad \text{(a)} \\ \psi(x) \cdot \nu = x \cdot \nu \,, & x \in \partial\Omega_{\text{ref}} \quad \text{(b)} \end{cases} \tag{7}$$

with ν the unit outward normal to the boundary $\partial\Omega_{\text{ref}}$.

The function G is a user prescribed or constructed function that compresses the mesh in regions where G is small, and expands the mesh in regions where G is large. Note that G is a physical target Jacobian function defined on the physical domain Ω. Assuming that a map ψ satisfying (7) is found, the function G then describes the size of the cells in $\mathcal{T}$. Let $\mathcal{V}$ denote a cell in $\mathcal{T}$, and $\mathcal{V}_{\text{ref}} = \psi^{-1}(\mathcal{V})$ the uniform cell in $\mathcal{T}_{\text{ref}}$ mapped to $\mathcal{V}$ by ψ. If G is sufficiently smooth, a Taylor series argument shows that

$$|\mathcal{V}| := \int_{\mathcal{V}} \mathrm{d}y = \int_{\mathcal{V}_{\text{ref}}} \det \nabla \psi(x) \, \mathrm{d}x = |\mathcal{V}_{\text{ref}}| \cdot \mathsf{G}(\psi(x_c)) + \mathcal{O}(|\Delta x|^2) \,,$$

where x_c denotes the cell center of $\mathcal{V}_{\text{ref}}$. Thus, the value of G in $\mathcal{V}$ is a scaling factor that scales the uniform cell volume $|\mathcal{V}_{\text{ref}}| = \Delta x^1 \Delta x^2$ to the volume $|\mathcal{V}|$, up to some spatially fixed constant of order $\mathcal{O}(|\Delta x|)$.

It is convenient to formulate the problem for the inverse map $\phi = \psi^{-1}$, which is found as a solution to

$$\begin{cases} \det D\phi(y) = \frac{1}{\mathsf{G}(y)} \,, & y \in \Omega \quad \text{(a)} \\ \phi(y) \cdot \nu = y \cdot \nu \,, & y \in \partial\Omega \,. \quad \text{(b)} \end{cases} \tag{8}$$

For a solution to exist for (7), the function G is required to satisfy the *solvability condition*

$$\int_{\Omega} \frac{1}{\mathsf{G}(y)} \, \mathrm{d}y = \int_{\Omega_{\text{ref}}} \frac{\det \nabla \psi(x)}{\mathsf{G} \circ \psi(x)} \, \mathrm{d}x = |\Omega_{\text{ref}}| = |\Omega| \,. \tag{9}$$

If (9) holds, then the system (7) admits an infinitude of solutions. The question then becomes how to construct a solution ψ that is in some sense optimal. Our primary concern in this work is the development of a fast-running algorithm that can be easily implemented within an ALE framework for hydrodynamics simulations. We next describe a simple and efficient procedure for constructing a solution to (7).

3.2. The basic mesh generation procedure

The key to our fast-running algorithm is the reduction of the nonlinear equation (7) to a simple linear Poisson solve and transport equation solve. Our approach is motivated, as in [50,54,31], by the deformation method of Dacorogna and Moser [21]. Specifically, a solution to (7) is obtained by the five step construction provided in Algorithm 1. We refer to this algorithm as SAM or, in the context of time-dependent meshing, *static* SAM.

Algorithm 1 STATIC SAM.

Step 1: Assume that the physical target Jacobian function $\mathsf{G}:\Omega\to\mathbb{R}^+$ is given and satisfies the solvability condition

$$\int_\Omega \frac{1}{\mathsf{G}(y)}\,\mathrm{d}y = |\Omega|\,, \tag{10}$$

and let $\mathsf{F}(y)=1/\mathsf{G}(y)$. In practice, we are usually given an auxiliary target Jacobian function $\bar{\mathsf{G}}:\Omega\to\mathbb{R}^+$ that *does not* satisfy (10), and we define G and F by the following normalization procedure:

$$\bar{\mathsf{F}}(y)=\frac{1}{\bar{\mathsf{G}}(y)} \quad\longrightarrow\quad \mathsf{F}(y)=|\Omega|\frac{\bar{\mathsf{F}}(y)}{\int_\Omega \bar{\mathsf{F}}(y)\,\mathrm{d}y} \quad\longrightarrow\quad \mathsf{G}(y)=\frac{1}{\mathsf{F}(y)}\,.$$

Step 2: Solve the following linear Poisson equation with homogeneous Neumann boundary conditions for the *potential* $\Phi:\Omega_{\mathrm{ref}}\to\mathbb{R}$

$$\begin{cases} \Delta\Phi(x)=\mathsf{F}\circ\mathrm{id}(x)-1\,, & x\in\Omega_{\mathrm{ref}} \quad \text{(a)}\\ \nabla\Phi(x)\cdot\nu=0\,, & x\in\partial\Omega_{\mathrm{ref}} \quad \text{(b)}\end{cases} \tag{11}$$

Step 3: Define the velocity $\overline{w}:\overline{\Omega_{\mathrm{ref}}}\to\mathbb{R}^2$ as

$$\overline{w}(x)=\nabla\Phi(x)\,. \tag{12}$$

Step 4: Solve the following system of transport equations for the *flowmap* $\eta:\overline{\Omega_{\mathrm{ref}}}\times[0,1]\to\overline{\Omega}$

$$\begin{cases} \partial_\tau\eta+w\cdot\nabla\eta=0\,, & x\in\overline{\Omega_{\mathrm{ref}}}\text{ and }0<\tau\le 1 \quad \text{(a)}\\ \eta(x,0)=x\,, & x\in\overline{\Omega_{\mathrm{ref}}}\text{ and }\tau=0 \quad \text{(b)}\end{cases} \tag{13}$$

where the *transport velocity* $w:\overline{\Omega_{\mathrm{ref}}}\times[0,1]\to\mathbb{R}^2$ is defined as

$$w(x,\tau)=\frac{\overline{w}(x)}{\tau+(1-\tau)\mathsf{F}\circ\mathrm{id}(x)}\,. \tag{14}$$

Step 5: Define $\psi(x):=\eta(x,1)$. Then ψ solves (7).

3.2.1. Validity of construction

The proof that the map ψ constructed according to Algorithm 1 satisfies (7) proceeds as follows. Define the back-to-labels map $\xi:\Omega\times[0,1]\to\Omega_{\mathrm{ref}}$ by $\xi(y,\tau)=\eta^{-1}(y,\tau)$. The Eulerian transport equation for η is transformed into a Lagrangian advection equation for ξ:

$$\begin{cases} \partial_\tau\xi(y,\tau)=w\circ\xi(y,\tau)\,, & y\in\overline{\Omega}\text{ and }0<\tau\le 1 \quad \text{(a)}\\ \xi(y,0)=y\,, & y\in\overline{\Omega}\text{ and }\tau=0\,. \quad \text{(b)}\end{cases} \tag{15}$$

Note that $\xi|_{\tau=1}=\eta^{-1}|_{\tau=1}=\psi^{-1}=\phi$.

Next, define the quantity

$$\mathcal{R}(y,\tau)=J(y,\tau)\,[\tau+(1-\tau)F]\circ\xi(y,\tau)\,,$$

where $J(y,\tau)=\det D\xi(y,\tau)$ and $F=\mathsf{F}\circ\mathrm{id}$. We compute

$$\partial_\tau\mathcal{R}=\partial_\tau J\,[\tau+(1-\tau)F]\circ\xi+J\,[1-F]\circ\xi+J(1-\tau)\partial_\tau(F\circ\xi)\,.$$

We recall Euler's lemma, which states that $J(y,\tau)$ evolves according to $\partial_\tau J=J\,\mathrm{div}\,w\circ\xi$. Using (11)(a) and (12), we calculate

$$\mathrm{div}\,w=\frac{\mathrm{div}\,\overline{w}}{\tau+(1-\tau)F}-\frac{(1-\tau)\overline{w}\cdot\nabla F}{[\tau+(1-\tau)F]^2}=\frac{F-1-(1-\tau)w\cdot\nabla F}{\tau+(1-\tau)F}\,,$$

so that

$$\partial_\tau J\,[\tau+(1-\tau)F]\circ\xi=J\,[F-1-(1-\tau)w\cdot\nabla F]\circ\xi\,.$$

Next, we have that

$$\partial_\tau(F\circ\xi)=\partial_\tau\xi\cdot\nabla F\circ\xi=[w\cdot\nabla F]\circ\xi\,,$$

where we have used equation (15)(a).

Using the two formulae above, we find that $\partial_\tau R=0$, so that $\mathsf{F}(y)=R(y,0)=R(y,1)=\det D\phi(y)$ and thus ϕ satisfies (8)(a), which is in turn equivalent to (7)(a). The condition $w(x,\tau)\cdot\nu=0$ for every $x\in\partial\Omega_{\mathrm{ref}}$ and $0\le\tau\le 1$ ensures that (7)(b) is satisfied. □

3.2.2. Discussion

The first numerical implementation of the deformation method [50] utilized the no slip boundary conditions $\psi(x) = x$, $\forall x \in \partial\Omega_{\text{ref}}$, rather than the no penetration boundary conditions (7)(b) which permit tangential motion of boundary nodes. The method of proof in the original paper of [21], which includes an analysis of the Poisson problem (11), requires the domain Ω_{ref} to have smooth boundary $\partial\Omega_{\text{ref}}$ and so is not valid for the rectangular domains we consider in this work. A modified method, which avoids the use of the Poisson problem (11) via a direct construction of the deformation velocity field, is provided in [50], but the resulting numerical implementation yields poor quality grids with high levels of distortion [22]. On the other hand, as we shall demonstrate in our numerical experiments, the use of the Poisson equation (11) together with the slip boundary conditions (7)(b) produces smooth grids. Moreover, the arguments in [21] can be modified with the help of elliptic estimates on polygonal domains [33] to show that the procedure outlined in Algorithm 1 yields existence of a solution to (7).

The basic mesh generation scheme Algorithm 1 differs from other deformation methods in the literature, e.g. [54,31], in both its formulation and numerical implementation. Specifically, the use of the transport system (13) avoids costly interpolation procedures required for the solution of the Lagrangian advection equations in other deformation methods, which results in an order of magnitude speed-up. Moreover, our numerical algorithm produces solutions that converge with high-order accuracy, in contrast to other methods which only yield second-order accurate solutions, at best. In the following subsection, we describe in detail the two main steps of Algorithm 1, namely the Poisson solve in Step 2, and the transport equation solve in Step 4.

3.3. Numerical implementation details

3.3.1. FFT-based elliptic solve for Φ

The Poisson problem (11) is solved in frequency space using the Fast Fourier Transform (FFT). The solvability condition (10) is enforced by the normalization procedure described in Algorithm 1 with trapezoidal integration to compute integrals. The RHS of (11)(a) then has zero mean, and a (non-unique) solution to (11) exists. We choose a unique solution Φ with zero mean, enforced in spectral space by zeroing out the first frequency component. The use of FFT requires the forcing G to be periodic; we periodize the problem by doubling the size of the domain in each direction and extending G symmetrically to the extended domain.[8] In Step 3, the velocity $\overline{w}$ is also computed via FFT.

3.3.2. Boundary conditions and order of convergence

Solutions to the Poisson problem (11) in general have limited regularity due to the presence of corner singularities in the domain, unless the function G satisfies certain compatibility conditions [37]. In this work, we shall assume the stronger Neumann condition $D\mathsf{G}(y)\cdot\nu = 0$ for $y \in \partial\Omega$ to ensure high-order convergence of the numerical solution ψ in the limit of zero mesh size. If $D\mathsf{G}(y)\cdot\nu \neq 0$, then the symmetric extension of G is not differentiable on the boundary $\partial\Omega$ and is only Lipschitz continuous. In this case, the potential Φ, velocity $\overline{w}$, and solution ψ all converge with 2^{nd} order accuracy, but the convergence rate of the Jacobian determinant $\mathcal{J}(x)$ and cofactor matrix $a(x)$ is only 1.5.

On the other hand, if the function G does satisfy the Neumann condition $D\mathsf{G}(y)\cdot\nu = 0$ for $y \in \partial\Omega$, then the symmetric extension of G is at least twice continuously differentiable, and the quantities $\psi(x)$, $\mathcal{J}(x)$, and $a(x)$ all converge with (at least) 4^{th} order accuracy. We confirm this high order convergence with a numerical example in Section 3.4.2.

For most of the problems we consider in this work, the function G does indeed satisfy the Neumann condition. However, even if the Neumann condition is not satisfied, the errors in the numerical solution are localized to the boundary, and the meshes produced are still of high accuracy and quality. Additionally, boundary smoothing techniques [2,28] can be applied to obtain high order convergence. We implement a simplified version of this technique in Section 3.4.3 and demonstrate that the quantity $\mathcal{J}(x)$ converges with 2^{nd} order accuracy. The details of this boundary smoothing technique are provided in Appendix B.

3.3.3. Numerical solution of the transport equations

The solution η to the transport equations (13) is smooth, and we shall therefore utilize the simple 5th order linear upwind scheme to compute derivatives, with the upwind direction in the r-th coordinate determined based on the sign of w^r. For instance, if $w^1_{i,j} \geq 0$, then we approximate the derivative $\partial_1 f$ of a function f by

$$[\partial_1 f]_{i,j} = \frac{-2f_{i-3,j} + 15f_{i-2,j} - 60f_{i-1,j} + 20f_{i,j} + 30f_{i+1,j} - 3f_{i+2,j}}{60\Delta x^1} + \mathcal{O}(|\Delta x|^5)\,.$$

For time-integration, we utilize the standard explicit RK4 scheme, which has the associated stability condition

$$\text{CFL}_\tau = \Delta\tau\left(\frac{||w^1||_\infty}{\Delta x^1} + \frac{||w^2||_\infty}{\Delta x^2}\right) \leq C\,. \tag{16}$$

[8] An alternative implementation with the discrete cosine transform can also be used.

Table 1
Jacobian errors $\mathcal{E}_2$ demonstrating high order convergence of SAM solutions for the circular target Jacobian function (19).

Scheme		Cells					
		32×32	64×64	128×128	256×256	512×512	1024×1024
SAM	Error	2.85×10^{-2}	5.10×10^{-3}	5.96×10^{-4}	3.73×10^{-5}	1.87×10^{-6}	9.89×10^{-8}
	Order	–	2.5	3.1	4.0	4.3	4.2

Accordingly, the adaptive time-step $\Delta\tau$ is chosen via

$$\mathrm{CFL}_\tau = \frac{\Delta\tau}{\tau + \frac{(1-\tau)}{||\mathsf{G}||_\infty}} \left(\frac{||\overline{w}^1||_\infty}{\Delta x^1} + \frac{||\overline{w}^2||_\infty}{\Delta x^2} \right) . \tag{17}$$

Our numerical experiments have shown that $\mathrm{CFL}_\tau = 2$ is sufficient for a stable scheme.

3.4. High-order accuracy and a benchmark computation

3.4.1. Jacobian error metric

Assessing the accuracy and convergence behavior of the numerical solutions ψ produced with SAM requires an error metric. Since the exact solution ψ_{exact} to the scheme described in Algorithm 1 is not known, we shall instead use the L^2 Jacobian error, defined as

$$\mathcal{E}_2 := ||\mathcal{J}(x) - \mathsf{G} \circ \psi(x)||_{L^2} . \tag{18}$$

We use bicubic interpolation to compute the composition $\mathsf{G} \circ \psi$ in (18) and trapezoidal integration to compute the L^2 integral norm.

3.4.2. High order convergence of solutions

To demonstrate the high order convergence of numerical solutions computed with SAM, we perform a mesh generation experiment on $\Omega = [0,1]^2$ for the circular target Jacobian function

$$\bar{\mathsf{G}}(y) = 1 - \delta \exp\left\{ - \left| \sigma \left((y^1 - 0.5)^2 + (y^2 - 0.5)^2 - r^2 \right) \right|^2 \right\} , \tag{19}$$

which forces the mesh to resolve in an annular region containing the circle of radius r centered at $(0.5\,, 0.5)$. The parameters δ and σ control the smallest cell-size and width of the resolving region, respectively. We choose $\delta = 0.75$, $\sigma = 64$, and $r = 0.2$. See Fig. 5 for the meshes associated with a time-dependent version of (19).

We generate a sequence of meshes using SAM for cell resolutions $N = 32^2$ up to $N = 1024^2$. We compute the Jacobian errors $\mathcal{E}_2$ given by (18), with the Jacobian determinant $\mathcal{J}$ approximated using 4^{th} order central differencing (CD4). The errors provided in Table 1 show that SAM solutions exhibit the expected 4^{th} order accuracy. We note that, to the best of our knowledge, all other grid generation schemes are at best 2^{nd} order accurate.

3.4.3. Benchmarking against the MK mesh generation scheme

Now, we perform a numerical experiment to benchmark SAM against the MK mesh generation scheme [22], a brief description of which is provided in Appendix C. The test problem [22] we consider is as follows: the domain is $\Omega = [0,1]^2$, and the target Jacobian function is

$$\bar{\mathsf{G}}(y) = 2 + \cos(8\pi r) , \tag{20}$$

where $r = \sqrt{(y^1 - 0.5)^2 + (y^2 - 0.5)^2}$ is the radial coordinate.

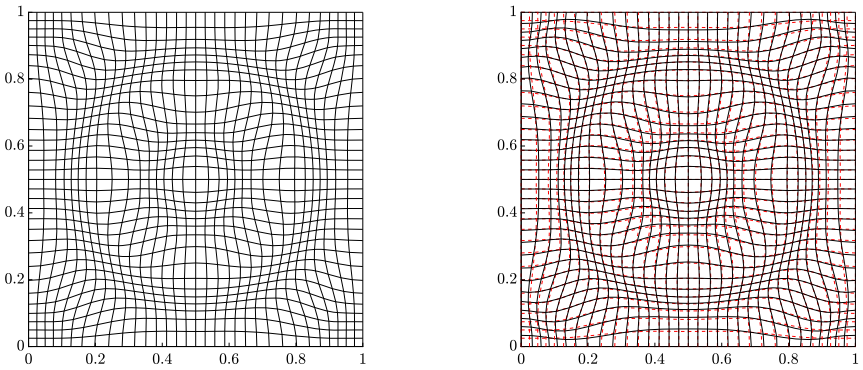
We compute a sequence of meshes for $N = 16^2$ up to $N = 256^2$ using SAM, and calculate the L^2 Jacobian errors $\mathcal{E}_2$. For the purposes of consistency with [22], we use a slightly different formula to compute $\mathcal{E}_2$ (see equations (46)-(52) in [22]). In particular, 2^{nd}-order differencing is used to calculate the Jacobian and, as such, we expect only 2^{nd} order convergence of the errors $\mathcal{E}_2$. Consequently, we instead use the 3^{rd} order linear upwind scheme for the transport equation solve. The pseudo-time step is set according to (17) with $\mathrm{CFL}_\tau = 8$.

The errors are listed in Table 2, along with the errors for the MK scheme obtained from [22]. The function $\bar{\mathsf{G}}$ is radially symmetric, and consequently does not satisfy the Neumann condition $D\mathsf{G} \cdot \nu \neq 0$. As such, the resulting solutions computed with SAM do not display 2^{nd} order accuracy in the limit $N \to \infty$, though, as shown in Table 2, the order of convergence only degrades to approximately 1.75 for the resolutions considered.

Nonetheless, we shall additionally consider a modified version of this test problem in which the function $\bar{\mathsf{G}}(y)$ in (20) is replaced by the function $\bar{\mathsf{G}}^*(y)$, where $\bar{\mathsf{G}}^*(y)$ is such that $D\bar{\mathsf{G}}^* \cdot \nu = 0$ on $\partial\Omega$. The function $\bar{\mathsf{G}}^*$ is equal to $\bar{\mathsf{G}}$ in the interior of Ω, but is mollified with an appropriate cut-off function in a small region near the boundary $\partial\Omega$ to enforce the Neumann

Table 2
Comparison of L^2 Jacobian errors and convergence rates for the MK and SAM schemes applied to (20). The data for the MK scheme is taken from Table 1 of [22].

Scheme		Cells				
		16×16	32×32	64×64	128×128	256×256
MK	Error	9.64×10^{-2}	2.80×10^{-2}	5.78×10^{-3}	1.46×10^{-3}	3.67×10^{-4}
	Order	–	1.78	2.28	1.99	1.99
SAM with $\bar{\mathsf{G}}$	Error	6.54×10^{-2}	2.05×10^{-2}	7.82×10^{-3}	2.00×10^{-3}	5.96×10^{-4}
	Order	–	1.68	1.39	1.96	1.75
SAM with $\bar{\mathsf{G}}^*$	Error	2.30×10^{-2}	1.44×10^{-2}	5.46×10^{-3}	1.25×10^{-3}	3.25×10^{-4}
	Order	–	0.68	1.40	2.12	1.94



(a) Mesh $\mathcal{T}^*$ for $\bar{\mathsf{G}}^*$.

(b) Comparison of $\mathcal{T}$ (black solid) and $\mathcal{T}^*$ (red dashed).

Fig. 2. SAM algorithm with boundary smoothing for the radial sinusoidal target function (20). Figure (a) is the 32×32 cell mesh $\mathcal{T}^*$ produced for the modified target Jacobian $\bar{\mathsf{G}}^*$, and Figure (b) is a comparison of the mesh $\mathcal{T}$ without boundary smoothing for the target Jacobian $\bar{\mathsf{G}}$ (black solid), and the mesh $\mathcal{T}^*$ for $\bar{\mathsf{G}}^*$ (red dashed). (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

Table 3
CPU runtimes for the MK scheme vs the SAM scheme. The results for the MK scheme are taken from Table 1 of [22] then divided by 2.2 to account for machine difference.

Scheme		Cells				
		16×16	32×32	64×64	128×128	256×256
MK	T_{CPU}	0.045	0.182	0.591	2.227	8.636
SAM	T_{CPU}	0.0012	0.0017	0.004	0.013	0.042
	speed-up factor	47	131	160	174	190

condition (see Appendix B for further details). The mesh $\mathcal{T}^*$ produced using SAM with $\bar{\mathsf{G}}^*$ is shown in Fig. 2(a), and a comparison with the mesh $\mathcal{T}$ for $\bar{\mathsf{G}}$ is shown in Fig. 2(b), from which it can be seen that the two meshes are very similar: they are nearly identical in the interior, with small differences near the boundary. While the solutions for $\bar{\mathsf{G}}$ do not attain the full 2^{nd} order accuracy, the solutions for $\bar{\mathsf{G}}^*$ do. The SAM solutions for $\bar{\mathsf{G}}^*$ have smaller errors than MK across all the resolutions considered. Moreover, the SAM solutions for $\bar{\mathsf{G}}^*$ display 2^{nd} order accuracy as N increases.[9] We also find that the L^2 mesh displacement $||\psi(x) - x||_{L^2} \approx 0.0178$ is comparable to the value of 0.0174 for MK reported in Table 1 of [22].

Next, we benchmark the computational efficiency of our SAM scheme against the MK scheme. We list in Table 3 the CPU runtimes for the MK scheme and the SAM scheme, where the MK runtimes are taken from Table 1 of [22]. To account for the different machines on which the MK and SAM schemes were run on, we divide the MK runtimes by 2.2, where the factor of 2.2 is determined from a machine comparison experiment, the details of which are provided in Appendix C. We then list the speed-up factor of the SAM scheme over the MK scheme in the final row of Table 3. We find that our SAM scheme is almost 200 times faster than the MK scheme at $N = 256^2$ cell resolution. We also note that the CPU times reported in [22] do not include the cost of an interpolation call, which is non-negligible at high resolutions.

[9] We have verified this up to $N = 2048^2$ but for brevity do not show the results here.

4. Fast dynamic adaptive meshing

4.1. Dynamic formulation

Given a time interval $t \in [0, T]$, we seek to construct a time dependent diffeomorphism $\psi : \Omega_{\text{ref}} \times [0, T] \to \Omega$. We denote the time-dependent mesh on Ω by $\mathcal{T}(t)$, which will be found as the image of $\mathcal{T}_{\text{ref}}$ under the action of $\psi(\cdot, t)$. The time-dependent map $\psi : \Omega_{\text{ref}} \times [0, T] \to \Omega$ is constructed by prescribing, for each $t \in [0, T]$, its Jacobian determinant. Let $\mathsf{G} : \Omega \times [0, T] \to \mathbb{R}^+$ denote a given time-dependent (physical) target Jacobian function. Then ψ satisfies

$$\begin{cases} \det \nabla \psi(x, t) = \mathsf{G} \circ \psi(x, t), & (x, t) \in \Omega_{\text{ref}} \times [0, T] \quad \text{(a)} \\ \psi(x, t) \cdot \nu = x \cdot \nu, & (x, t) \in \partial\Omega_{\text{ref}} \times [0, T] \quad \text{(b)} \end{cases} \tag{21}$$

The target Jacobian function G must satisfy, for each $t \in [0, T]$, the following integral constraint to ensure that (21) has a solution:

$$\int_{\Omega} \frac{1}{\mathsf{G}(y, t)} \, \mathrm{d}y = \int_{\Omega_{\text{ref}}} \frac{\det \nabla \psi(x, t)}{\mathsf{G} \circ \psi(x, t)} \, \mathrm{d}x = |\Omega| . \tag{22}$$

4.1.1. Temporal discretization

We uniformly discretize the time domain $[0, T]$ into K intervals of length Δt and set $t_k = k\Delta t$ for $k = 0, 1, \ldots, K$. Denote $\mathsf{G}_k := \mathsf{G}(\cdot, t_k)$, $\psi_k := \psi(\cdot, t_k)$, and $\mathcal{T}_k = \mathcal{T}(t_k)$. Then each $\psi_k : \Omega_{\text{ref}} \to \Omega$ is a diffeomorphism satisfying

$$\begin{cases} \det \nabla \psi_k(x) = \mathsf{G}_k \circ \psi_k(x), & x \in \Omega_{\text{ref}} \quad \text{(a)} \\ \psi_k(x) \cdot \nu = x \cdot \nu, & x \in \partial\Omega_{\text{ref}} \quad \text{(b)} \end{cases} \tag{23}$$

with each target Jacobian function $\mathsf{G}_k : \Omega \to \mathbb{R}^+$ satisfying the integral constraint

$$\int_{\Omega} \frac{1}{\mathsf{G}_k(y)} \, \mathrm{d}y = \int_{\Omega_{\text{ref}}} \frac{\det \nabla \psi_k(x)}{\mathsf{G}_k \circ \psi_k(x)} \, \mathrm{d}x = |\Omega| . \tag{24}$$

4.1.2. Algorithmic complexity

The simplest possible strategy for (23) is to compute each map ψ_k using static SAM. Our numerical experiments indicate that static SAM is highly efficient for a single mesh generation call. However, for unsteady fluids simulations which require dynamic meshing at every time-step, the use of static SAM can be expensive at high resolutions due to the computational bottleneck in the transport equation solve stage.

Specifically, suppose that the target Jacobian function has large deviation from the identity i.e. $||1/\mathsf{G} - 1||_{L^\infty} \gg 1$. Then the associated potential Φ solving (11) has large gradients, and the flowmap velocity (14) will therefore be large in magnitude. Consequently, many pseudo-time steps will be required in the transport equation solve to preserve stability and accuracy of the computed numerical solution for $\eta(x, \tau)$. In particular, the stability condition (16) forces the pseudo-time step to decay like $\mathcal{O}\left(N^{-1/2}\right)$, so that the overall complexity of the SAM algorithm is $\mathcal{O}\left(N^{3/2}\right)$. For large N, this can become prohibitively computationally expensive. In the next section, we resolve this issue via a novel reformulation of (23).

4.2. Reformulation in terms of near-identity maps

4.2.1. The perturbation formulation

Assume that we are given the map ψ_k and the target Jacobian functions G_k and G_{k+1}, and suppose that we wish to compute the map ψ_{k+1}. Rather than computing the map ψ_{k+1} directly by solving (23), we instead solve for the *perturbation map* $\delta\psi_{k+1} : \Omega_{\text{ref}} \to \Omega_{\text{ref}}$ defined implictly by

$$\psi_{k+1}(x) = \psi_k \circ \delta\psi_{k+1}(x) . \tag{25}$$

That is, we suppose that the map ψ_{k+1} can be found as the image of ψ_k acting on a near-identity transformation $\delta\psi_{k+1}$ on the reference domain Ω_{ref} (see Fig. 3).

By the chain rule and inverse function theorem, we find that $\delta\psi_{k+1}$ satisfies

$$\begin{cases} \det \nabla \delta\psi_{k+1}(x) = P_{k+1} \circ \delta\psi_{k+1}(x), & x \in \Omega_{\text{ref}} \quad \text{(a)} \\ \delta\psi_{k+1}(x) \cdot \nu = x \cdot \nu, & x \in \partial\Omega_{\text{ref}} \quad \text{(b)} \end{cases} \tag{26}$$

with the function $P_{k+1} : \Omega_{\text{ref}} \to \mathbb{R}^+$ defined as

$$P_{k+1}(x) = \left(\frac{\mathsf{G}_{k+1}}{\mathsf{G}_k} \right) \circ \psi_k(x) . \tag{27}$$

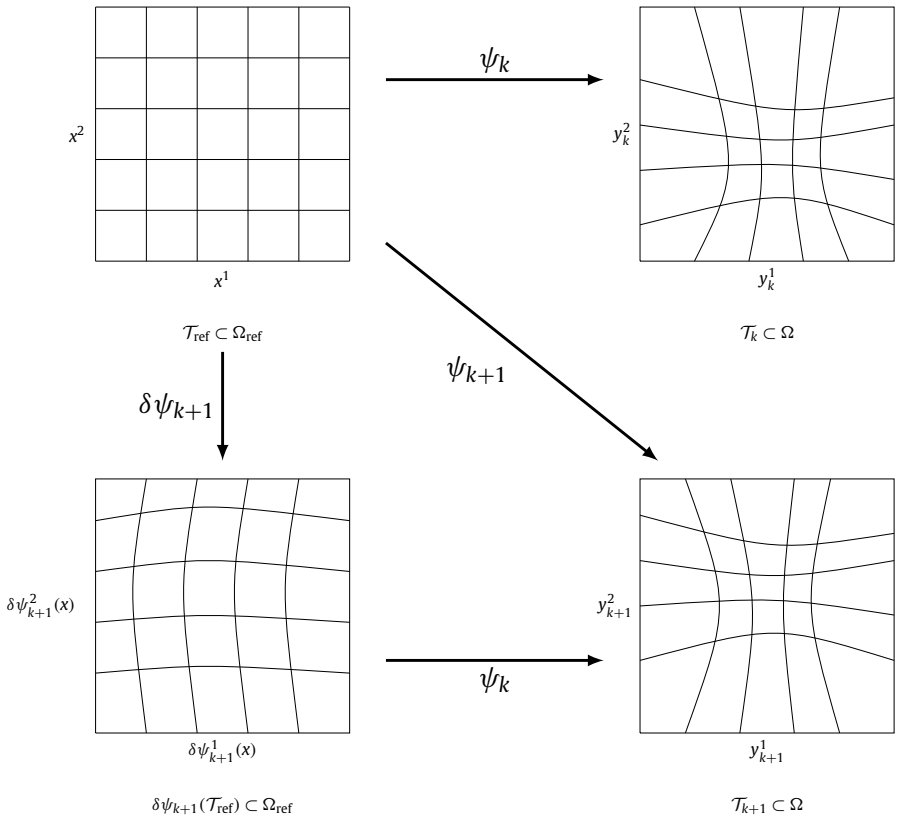


Fig. 3. Schematic of the meshes, and the maps between them, for dynamic mesh generation.

The system (26) is of exactly the same form as (7), and the identical solution procedure described in Section 3.2 for static mesh generation can therefore be used to find the solution $\delta\psi_{k+1}$. Then ψ_{k+1} is computed according to (25). A complete description of the dynamic SAM scheme is provided in Algorithm 2.

Algorithm 2 DYNAMIC SAM.

Step 1: Set $t = 0$. Given an initial target Jacobian function $\mathsf{G}_0 : \Omega \to \mathbb{R}^+$, compute the initial diffeomorphism ψ_0 according to the static solution scheme from Section 3.2.

Step 2: For $t = t_{k+1}$, assume that we are given the following: the map ψ_k and target Jacobian function $\mathsf{G}_k : \Omega \to \mathbb{R}^+$, both from the previous time step $t = t_k$, and the target Jacobian function $\mathsf{G}_{k+1} : \Omega \to \mathbb{R}^+$ at the current time level.
Define $P_{k+1} : \Omega_{\text{ref}} \to \mathbb{R}^+$ by (27), and assume that it satisfies the solvability condition

$$\int_{\Omega_{\text{ref}}} \frac{1}{P_{k+1}(x)}\,\mathrm{d}x = \int_{\Omega_{\text{ref}}} \left(\frac{\mathsf{G}_k}{\mathsf{G}_{k+1}}\right) \circ \psi_k(x)\,\mathrm{d}x = |\Omega_{\text{ref}}|\,, \tag{28}$$

In general, we will be given target Jacobian functions $\bar{\mathsf{G}}_k$ and $\bar{\mathsf{G}}_{k+1}$ such that the corresponding $\bar{P}_{k+1} = (\bar{\mathsf{G}}_{k+1}/\bar{\mathsf{G}}_k) \circ \psi_k$ does not satisfy (28). In this case, we define P_{k+1} according to the following normalization procedure:

$$\bar{Q}_{k+1}(x) = \frac{1}{\bar{P}_{k+1}(x)} \;\longrightarrow\; Q_{k+1}(x) = |\Omega| \frac{\bar{Q}_{k+1}(x)}{\int_{\Omega_{\text{ref}}} \bar{Q}_{k+1}(x)\,\mathrm{d}x} \;\longrightarrow\; P_{k+1}(x) = \frac{1}{Q_{k+1}(x)}\,.$$

Step 3: Solve (26) for the perturbation map $\delta\psi_{k+1} : \Omega_{\text{ref}} \to \Omega_{\text{ref}}$ using the solution procedure in Section 3.2.
Step 4: Define $\psi_{k+1} : \Omega_{\text{ref}} \to \Omega$ by (25). If $t_{k+1} = T$, then stop; otherwise, set $t = t_{k+2}$, and return to **Step 2**.

4.2.2. Discussion

The key to the efficiency of dynamic SAM is the reformulation in terms of the perturbation map $\delta\psi_{k+1}$ satisfying (26). Specifically, while it may be that both G_k and G_{k+1} have large deviation from 1 i.e. $||1/\mathsf{G}_k - 1||_{L^\infty} \gg 1$ and $||1/\mathsf{G}_{k+1} - 1||_{L^\infty} \gg 1$, we may nonetheless have that $||1/P_{k+1} - 1||_{L^\infty} \ll 1$. Indeed, this is the case in ALE simulations, which are naturally constrained by a CFL condition that limits the evolution of the numerical solution over a single time-step.

More precisely, the usual stability condition for the *physical* time step Δt in an (Eulerian) simulation forces the time step Δt to decay like $\Delta t \sim 1/\sqrt{N}$ as $N \to \infty$, which is a constraint of exactly the same form as the condition (16) on $\Delta\tau$. A Taylor series argument shows that

$$1 - \frac{1}{P_{k+1}(x)} = \Delta t \cdot \frac{\partial_t \mathsf{G}(\psi(x,t_k),t_k)}{\mathsf{G}(\psi(x,t_k),t_k)} + \mathcal{O}(\Delta t^2)$$
$$\implies ||1 - 1/P(\cdot,t)||_{L^\infty} = \mathcal{O}(\Delta t) = \mathcal{O}(1/\sqrt{N})\,, \tag{29}$$

since $\partial_t \mathsf{G} \sim \mathcal{O}(1)$. Since the stability condition for the pseudo-time step $\Delta\tau$ scales according to (16), and $||w||_{L^\infty} = \mathcal{O}(1/\sqrt{N})$ by (29), we see that $\Delta\tau = \mathcal{O}(1)$ i.e. the pseudo-time step $\Delta\tau$ *can be kept fixed across resolutions* N, resulting in a dynamic SAM algorithm with optimal complexity.

We emphasize here that our perturbation formulation (26) differs from the methods considered in [67,32] in an important way. In particular, our perturbation map $\delta\psi_{k+1}$ is a near identity transformation of the *uniform reference domain* Ω_{ref}, whereas the schemes in [67,32] define a perturbation map via $\overline{\psi}_{k+1} = \delta\overline{\psi}_{k+1} \circ \overline{\psi}_k$, rather than through (25). This means that the $\overline{\psi}_k$ solutions in [67,32] are different from our ψ_k SAM solutions. Moreover, the equation for $\delta\overline{\psi}_{k+1}$ is posed on the deformed mesh $\mathcal{T}_k$; this means that the solvers for the Poisson problem and transport equation must be appropriately modified at each time-step t_k, requiring, for example, the costly recalculation of the stiffness and mass matrices. Consequently, SAM is simpler, faster, and more accurate than the schemes in [67,32]. For instance, the scheme in [32] has order of accuracy 1.5, whereas our dynamic SAM solutions converge with 4^{th} order accuracy if the data is sufficiently smooth. Additionally, the interpolation routine in [32] requires $\mathcal{O}(N^{1.5})$ grid searching on deformed grids, in contrast to our $\mathcal{O}(N)$ SAM algorithm.

4.3. Restarted dynamic mesh generation

The perturbation formulation (26), by design, follows the time history of $\psi(x,t)$. That is to say, the solution $\psi(x,t)$ at time $t = t_k$ depends upon the solution for all $t < t_k$. As such, numerical solutions to (26) are susceptible to increasing grid distortion and mesh tangling, a common ailment of Lagrangian-type methods. To mitigate this issue, we can periodically *restart* the dynamic mesh generation by computing at time $t = t_k$ the map ψ_k directly with static SAM, rather than with dynamic SAM. In this way, the greater efficiency of dynamic SAM is utilized, while grid distortion errors are controlled with the use of static SAM, thereby preventing mesh tangling. The restarting criterion is chosen as $\lambda_k > \Lambda\lambda_{\text{ref}}$, where λ_k is the L^1 grid distortion at time step t_k, λ_{ref} is a "reference" grid distortion (defined in Algorithm 3), and Λ is a user prescribed parameter. A description of our restarted dynamic SAM scheme is provided in Algorithm 3.

Algorithm 3 RESTARTED DYNAMIC SAM.

Step 0: Choose the maximum grid distortion parameter $\Lambda > 1$.

Step 1: Set $t = 0$. Given an initial target Jacobian function $\mathsf{G}_0 : \Omega \to \mathbb{R}^+$, compute the initial diffeomorphism ψ_0 according to the static solution scheme from Section 3.2. Let λ_{ref} be the (reference) L^1 grid distortion of the adaptive mesh $\mathcal{T}_0$, computed according to (30).

Step 2: For $t = t_{k+1} > 0$, compute the average grid distortion of the map ψ_k

$$\lambda_k := \left\| \frac{1}{2}\text{Tr}\left(\nabla\psi_k \nabla\psi_k^{\mathcal{T}}\right) \right\|_{L^1}. \tag{30}$$

Step 3: If $\lambda_k > \Lambda\lambda_{\text{ref}}$, then compute the map ψ_{k+1} using static SAM Algorithm 1 and recalculate λ_{ref} according to (30). Otherwise, compute ψ_{k+1} using dynamic SAM Algorithm 2. If $t_{k+1} = T$, then stop; otherwise, set $t = t_{k+2}$, and return to **Step 2**.

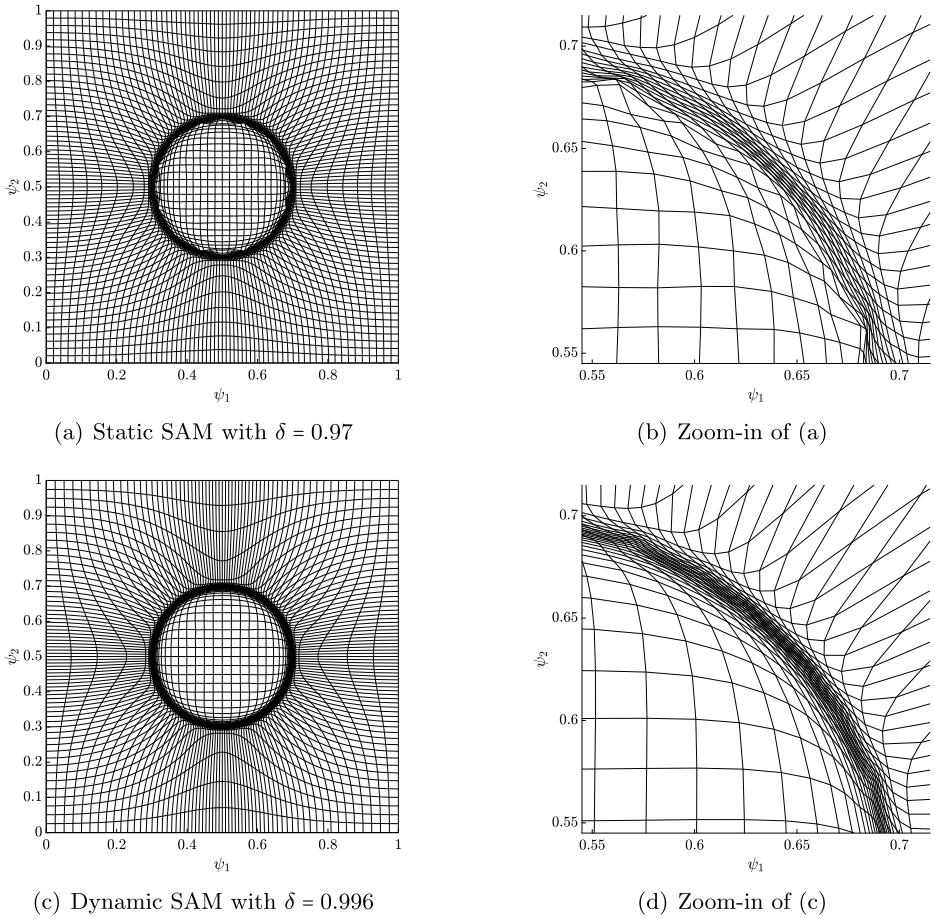
5. Dynamic mesh generation experiments

In this section, we present and discuss the results of several dynamic mesh generation experiments conducted with the static, dynamic, and restarted SAM algorithms. Unless otherwise stated, all experiments are conducted on the unit square $\Omega = [0,1]^2$ with an equal number of cells in the horizontal and vertical directions $m = n = \sqrt{N}$.

5.1. Static mesh with large zoom-in factor

This static test problem demonstrates the ability of dynamic SAM to generate smooth meshes with large zoom-in factors which, in practical applications, can be used to track very small scale structures with only a few total number of cells. On the other hand, when the target function G has large gradients (as is the case for large zoom-in meshes), numerical errors in the Poisson solve often lead to poor quality grids containing non-convex elements [22,32].

As an example, consider the circular target Jacobian function $\mathsf{G}_\delta(y)$ given by (19) with $\sigma = 64$, $r = 0.2$, and $\delta \in [0,1)$. More generally, we have a family of target functions $\{\mathsf{G}_\delta(y)\}_{0\le\delta<1}$ parametrized by δ, with each such G_δ forcing the mesh to resolve around some given curve (see equation (59)). The zoom-in parameter δ determines the zoom-in factor $\Upsilon = 1/\min\mathcal{J}$ of the adaptive mesh $\mathcal{T}$, and thus the smallest scales that can be represented on $\mathcal{T}$. When $\delta = 0$ (uniform mesh) we have



(a) Static SAM with $\delta = 0.97$ (b) Zoom-in of (a)

(c) Dynamic SAM with $\delta = 0.996$ (d) Zoom-in of (c)

Fig. 4. Test Problem 5.1 demonstrating smooth large zoom-in meshing using dynamic SAM. Shown are the 64^2 cell meshes with large zoom-in parameter δ for the circular target Jacobian function (19). Figure (a) is the poor quality mesh containing non-convex elements produced with static SAM with $\delta = 0.97$, and (b) is a zoom-in of (a) near the refining region. Figure (c) is the smooth large zoom-in mesh produced with dynamic SAM with $\delta = 0.996$ and with smallest cell 100 times smaller than a uniform cell, and (d) is a zoom-in.

$\Upsilon = 1$. As δ increases, so does Υ, with smaller and smaller scales captured on $\mathcal{T}$. As $\delta \to 1$, $\Upsilon \to \infty$ and the mapping is degenerate at $\delta = 1$. From the point of view of efficiency, we would like to have Υ large since we then require few total number of grid points. Moreover, for unstable RT problems that have evolving interfaces with large curvature, it is essential that we construct adaptive meshes with large enough Υ such that they can capture the small-scale vortical structures of the flow. As such, we often want to choose $\delta \approx 1$, and refer to the associated meshes as "large zoom-in" meshes.

While the continuous mapping ψ is non-degenerate for all $0 \leq \delta < 1$, in practice numerical errors in static SAM will produce folded grids if δ is sufficiently close to 1. That is, for each N, there exists a corresponding $\delta_{\max}$ such that the grids produced with static SAM for $\delta > \delta_{\max}$ contain non-convex elements. An example of such a grid is shown for $N = 64^2$ and $\delta = 0.97$ in Figs. 4(a) and 4(b). The function G_δ is such that $||1 - 1/\mathsf{G}_\delta||_{L^\infty} \approx \frac{1}{1-\delta} \to \infty$ as $\delta \to 1$. When $\delta \approx 1$, large errors in the numerical solution of the Poisson problem lead to grids with non-convex elements.

Dynamic SAM provides a simple method for producing smooth grids with $\delta \approx 1$. We define the time-dependent function

$$\mathsf{G}(y, s) = (1 - s) + s\mathsf{G}_\delta(y)\,. \tag{31}$$

Then (31) linearly interpolates between 1 at $s = 0$ and G_δ at $s = 1$, and applying dynamic SAM with sufficiently many time steps Δs yields smooth grids with no non-convex elements. As an example, we set $\Delta s = 0.05$ and construct a 64^2 cell mesh using Algorithm 2 with $\delta = 0.996$ in (19). The resulting grid, shown in Figs. 4(c) and 4(d), is smooth with $\Upsilon \approx 100$. In Section 7, we consider large zoom-in meshing for the more complicated Rayleigh-Taylor test.

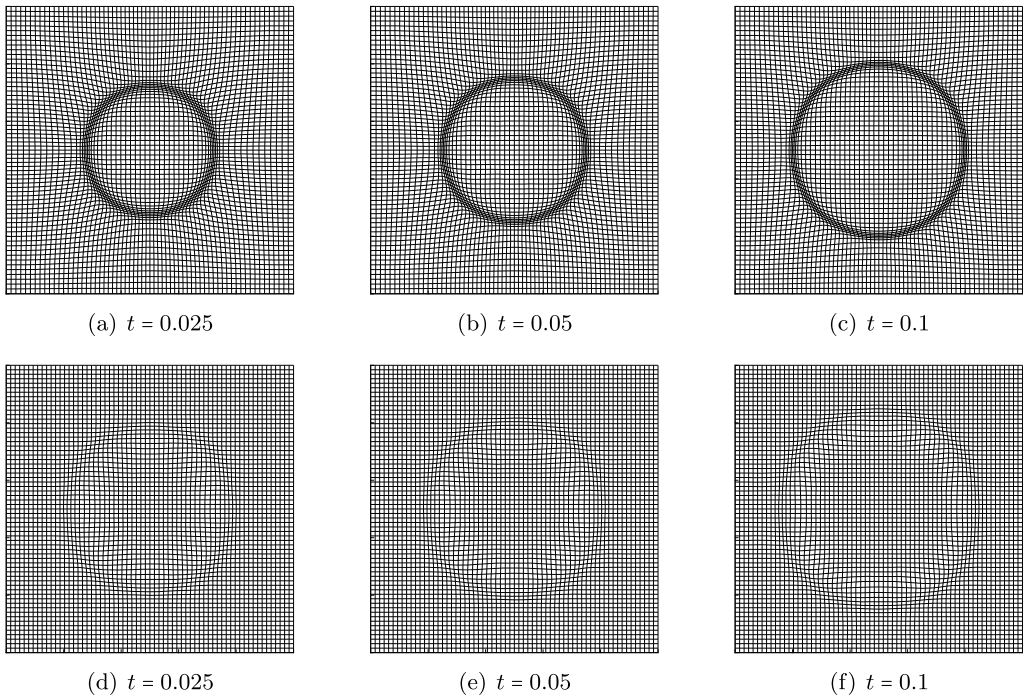


Fig. 5. Test Problem 5.2: tracking a propagating circular front modeling a shock wave. Shown are the 64^2 cell meshes produced using dynamic SAM for the circular target Jacobian function (32). The top row shows the adaptive meshes $\mathcal{T}(t)$, while the bottom row shows the corresponding "perturbation meshes" $\delta\psi_k(\mathcal{T}_{\text{ref}})$.

5.2. Propagating circular front

5.2.1. Problem description

Our first dynamic mesh generation experiment tracks a circular front propagating radially outwards with radial velocity 1. The time-dependent target Jacobian function is defined as

$$\bar{\mathsf{G}}(y,t) = 1 - \delta \exp\left\{ - \left| \sigma \left[(y^1 - 0.5)^2 + (y^2 - 0.5)^2 - r(t)^2 \right] \right|^2 \right\} . \tag{32}$$

The parameters are chosen as $\delta = 0.75$, $\sigma = 64$, and the radius is $r(t) = 0.2 + t$. We generate a sequence of meshes for $0 \le t \le 0.1$.

The choice of time step Δt depends upon N as

$$\Delta t = \frac{0.64}{2\sqrt{N}} . \tag{33}$$

This choice of scaling for Δt is motivated by the CFL condition. Since the radial velocity of the propagating front is 1, we can estimate that the CFL number associated with (33) is 0.64.

5.2.2. Results

The 64^2 cell adaptive meshes $\mathcal{T}(t)$ for (32) are shown in the top row of Fig. 5 at various times t. The computed meshes $\mathcal{T}_k$ are smooth and are correctly resolved around the evolving circular front. The meshes $\delta\psi_k(\mathcal{T}_{\text{ref}})$ are shown at the same times in the bottom row of Fig. 5; from these figures, it is clear that $\delta\psi_k(\mathcal{T}_{\text{ref}})$ is a near-identity transformation of the uniform mesh $\mathcal{T}_{\text{ref}}$. For this problem, the function G is such that $||1 - 1/\mathsf{G}(\cdot,t)||_{L^\infty} \approx 2.43$, whereas the perturbation density P is such that $||1 - 1/P(\cdot,t)||_{L^\infty} \approx 0.3$.

5.2.3. Comparison with static SAM

Next, we conduct a grid resolution study with N ranging from $N = 32^2$ to $N = 512^2$, and compare the results of dynamic SAM with those of static SAM. The Jacobian errors $\mathcal{E}_2$ at the final time $t = 0.1$ are shown in Table 4. Both schemes exhibit 4^{th} order accuracy, as expected, but the dynamic SAM solutions have smaller errors. This is due to the higher accuracy of the Poisson solve in the dynamic method vs the static method.

Table 4
Test Problem 5.2: tracking a propagating circular front. We list the L^2 Jacobian errors $\mathcal{E}_2$ at $t = 0.1$, convergence rates, and total CPU runtimes for static and dynamic SAM. The results confirm that dynamic SAM produces high order accurate solutions and is of optimal complexity.

Scheme		Cells					
		32×32	64×64	128×128	256×256	512×512	1024×1024
Static SAM	$\mathcal{E}_2$	4.23×10^{-2}	1.15×10^{-2}	1.22×10^{-3}	9.29×10^{-5}	4.98×10^{-6}	2.65×10^{-7}
	Order	–	1.9	3.2	3.7	4.2	4.2
	T_{CPU} (sec)	0.006	0.049	0.59	7.26	112.8	1663
Dynamic SAM	$\mathcal{E}_2$	4.05×10^{-2}	6.66×10^{-3}	6.18×10^{-4}	4.00×10^{-5}	2.23×10^{-6}	1.33×10^{-7}
	Order	–	2.6	3.4	3.9	4.2	4.1
	T_{CPU} (sec)	0.017	0.101	0.869	7.31	65.2	582
	speed-up factor	0.33	0.48	0.68	0.99	1.73	2.86

At low resolutions, the dynamic SAM runtimes are greater than those for static SAM. This is due to the interpolation required in the dynamic SAM algorithm. On the other hand, static SAM is of complexity $\mathcal{O}\left(N^{3/2}/\Delta t\right) = \mathcal{O}\left(N^2\right)$, whereas dynamic SAM is of optimal complexity $\mathcal{O}\left(N/\Delta t\right) = \mathcal{O}\left(N^{3/2}\right)$. For this test, dynamic SAM becomes more efficient than static SAM at $N = 256^2$.

5.3. Uniformly rotating patch

5.3.1. Problem description

Our next mesh generation experiment assesses the performance of SAM for target Jacobian functions of the form

$$\bar{\mathsf{G}}(y,t) = \frac{1}{1 + M \exp\left\{ -\left(\sigma \left| \left[(y^1 - 0.5 - r\cos(2\pi t))^2 + (y^2 - 0.5 - r\sin(2\pi t))^2 - R^2 \right] \right| \right)^2 \right\}}. \tag{34}$$

Equation (34) forces the mesh to concentrate nodes within a uniformly rotating (with angular velocity $\omega = 2\pi$) circular patch of radius $R > 0$, whose center is a distance $r \geq 0$ from $(0.5, 0.5)$. The constant $M \geq 0$ determines the zoom-in factor, and σ controls the width of the transition region from fine to coarse scale of the mesh.

5.3.2. Comparison with the schemes in [71]

The case $M = 5$, $\sigma = 50$, $r = 0.25$, and $R = 0.1$ in (34) corresponds to a test problem from [71]. Therein, the authors compare four different mesh generation methods and conclude that the so-called Parabolic Monge-Kantorovich method (PMKP) is the best method among the four for (34), both in terms of accuracy as well as efficiency. The PMKP method is similar to the MK scheme, but replaces the nonlinear Newton-Krylov solver in MK with a parabolization (in pseudo-time τ) and time-stepping until a steady state is reached. The solution in PMKP is only found in the asymptotic limit $\tau \to \infty$, whereas the SAM solution is computed at pseudo-time $\tau = 1$. Moreover, the explicit integration of the parabolic PDE requires that the pseudo-time step scales like $\Delta\tau \sim \frac{1}{N}$ to ensure 2^{nd} order convergence. In contrast, static SAM requires only that $\Delta\tau \sim \frac{1}{\sqrt{N}}$, while for dynamic SAM we can keep $\Delta\tau = \mathcal{O}(1)$.

We set $N = 40^2$, $\Delta t = 0.01$, and generate a sequence of meshes for $0 \leq t \leq 1$. The adaptive meshes generated with static, dynamic, and restarted SAM are shown in Fig. 6 at various times t. The Jacobian errors, mean grid distortion, and cumulative simulation runtimes are provided in Table 5. For the purposes of comparison with [71], we also provide the *mesh fidelity measure* $\hat{\mathcal{E}}_2$, defined by

$$\hat{\mathcal{E}}_2 \coloneqq \left| ||\mathcal{J}(\cdot,t)/\mathsf{G} \circ \psi(\cdot,t)||_{L^2} - 1 \right|. \tag{35}$$

The superior accuracy of SAM produces fidelity measures $\hat{\mathcal{E}}_2$ that are an order of magnitude smaller than those produced with the PMKP method (see Tables 6 and 7 in [71]). Moreover, the SAM runtimes are more than two orders of magnitude smaller than the PMKP runtimes provided in [71] e.g. 0.179 sec vs 75 sec for static SAM vs PMKP.

Since static SAM constructs the map ψ directly from the uniform mesh, the meshes at $t = 0.25, 0.5, 0.75, 1.0$ are simply rotated versions of the initial grid. This is confirmed in Table 5, which shows that static SAM produces grids with identical grid quality metrics at these times. Dynamic SAM, on the other hand, necessarily tracks the history of the simulation, and the rotating target Jacobian produces grids with increasing levels of distortion. The Jacobian errors of dynamic SAM are smaller than static SAM for $t = 0.25$ and $t = 0.5$ due to the high accuracy with which the Poisson problem is solved for in the perturbation formulation. For $t > 0.6$, however, grid distortion errors outweigh the improved accuracy for the Poisson solve, and dynamic SAM errors become larger than static SAM errors. The restart criterion parameter in restarted SAM is set as $\Lambda = 1.01$. The mesh restarting controls the grid distortion errors, which in turn prevents the Jacobian errors from

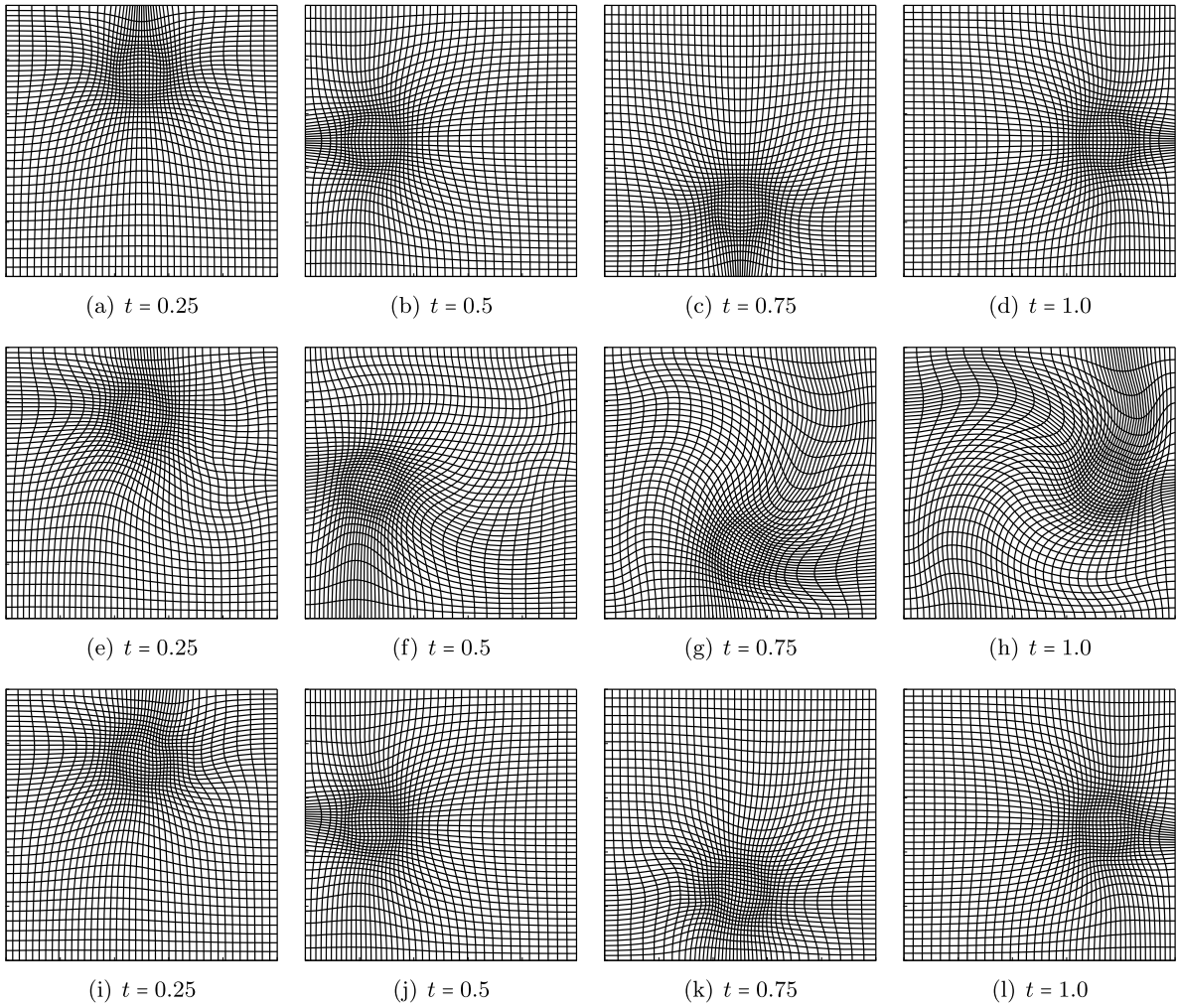


Fig. 6. Test Problem 5.3: tracking a uniformly rotating patch with target function (34). Shown are plots of the 40^2 cell adaptive meshes at various times t. The meshes are produced using static SAM (top), dynamic SAM (middle), and restarted SAM (bottom). Restarted SAM removes the grid distortion errors associated with Lagrangian methods.

growing. As shown in the bottom row of Fig. 6, and confirmed in Table 5, restarted SAM grids are of comparable accuracy and smoothness to static SAM grids

At this low resolution, dynamic SAM is actually slower than the static algorithm. For higher resolutions, however, dynamic SAM is much more efficient than static SAM. To demonstrate this, we repeat the above experiment with $N = 400^2$ cells. For brevity, we do not report the Jacobian errors or L^1 distortion, since the conclusions are similar to the $N = 40^2$ case. The computational runtimes, however, are very different: 1414 sec for static SAM vs 184 sec for dynamic SAM, and 194 sec for restarted SAM. We thus see that restarted SAM combines the best aspects of static and dynamic SAM i.e. smoothness and efficiency, respectively.

5.4. Differential rotation with small scales

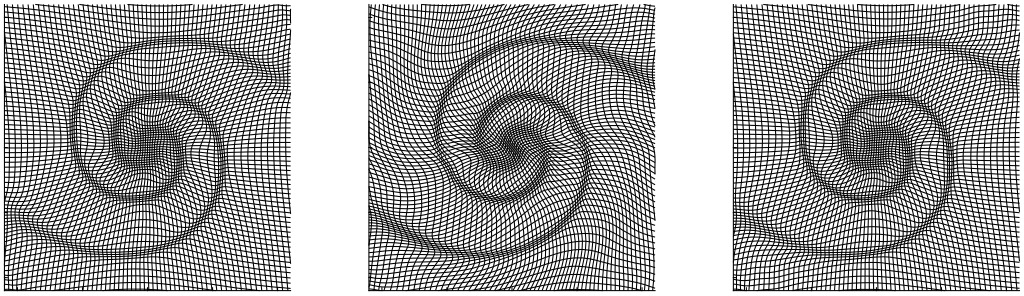
5.4.1. Problem description

This dynamic mesh generation test models a Gaussian "blob" deforming under a rotating flow in which the angular velocity is dependent upon the distance from the center of the blob [18]. The time-dependent target Jacobian function is defined as

$$\bar{\mathsf{G}}(y,t) = \frac{1}{1 + 4\exp\left[-r(y)^2\left(\frac{\cos^2\theta_0(y,t)}{\sigma_1} + \frac{\sin^2\theta_0(y,t)}{\sigma_2}\right)\right]}, \tag{36}$$

Table 5
Test Problem 5.3: tracking a uniformly rotating patch. We provide the mesh fidelity measure $\hat{\mathcal{E}}_2$, L^2 Jacobian error $\mathcal{E}_2$, L^1 distortion λ, and cumulative CPU runtime T_{CPU} at various times t for the static, dynamic, and restarted SAM schemes. The mesh fidelity measures $\hat{\mathcal{E}}_2$ of SAM solutions are an order of magnitude smaller than those provided in [71]. Additionally, static SAM is more than 400 times faster than the schemes in [71].

Scheme		Time				
		$t=0$	$t=0.25$	$t=0.5$	$t=0.75$	$t=1.0$
Static SAM	$\hat{\mathcal{E}}_2$	3.86×10^{-3}	3.86×10^{-3}	3.86×10^{-3}	3.86×10^{-3}	3.86×10^{-3}
	$\mathcal{E}_2$	3.79×10^{-2}	3.79×10^{-2}	3.79×10^{-2}	3.79×10^{-2}	3.79×10^{-2}
	λ	1.251	1.251	1.251	1.251	1.251
	T_{CPU} (sec)	0.003	0.048	0.091	0.134	0.179
Dynamic SAM	$\hat{\mathcal{E}}_2$	3.86×10^{-3}	1.13×10^{-3}	1.13×10^{-3}	1.49×10^{-3}	6.65×10^{-3}
	$\mathcal{E}_2$	3.79×10^{-2}	3.28×10^{-2}	3.49×10^{-2}	4.79×10^{-2}	6.31×10^{-2}
	λ	1.251	1.328	1.544	1.877	2.311
	T_{CPU} (sec)	0.003	0.049	0.094	0.139	0.185
Restarted SAM	$\hat{\mathcal{E}}_2$	3.86×10^{-3}	2.85×10^{-3}	1.18×10^{-3}	3.86×10^{-3}	2.85×10^{-3}
	$\mathcal{E}_2$	3.79×10^{-2}	3.05×10^{-2}	2.15×10^{-2}	3.79×10^{-2}	3.05×10^{-2}
	λ	1.251	1.252	1.259	1.251	1.252
	T_{CPU} (sec)	0.003	0.048	0.093	0.138	0.183



(a) static SAM (b) dynamic SAM (c) restarted SAM

Fig. 7. Test Problem 5.4: tracking small scale vortical structures in flows with differential rotation using (36). Shown are zoomed in plots of the 128^2 cell adaptive meshes at $t=90$. SAM produces smooth meshes without the grid distortion errors associated with Lagrangian-type schemes.

where $r(y)=|y-0.5|$ is the radial coordinate, $\theta_0(y,t)=\theta(y)+\omega(r)t$, and $\theta(y)=\arctan\left(\frac{y^2-0.5}{y^1-0.5}\right)$. The parameters σ_1 and σ_2 control the aspect ratio of the blob, while $\omega(r)$ is the angular velocity. As in [18], we set $\sigma_1=0.05$, $\sigma_2=0.001$, and

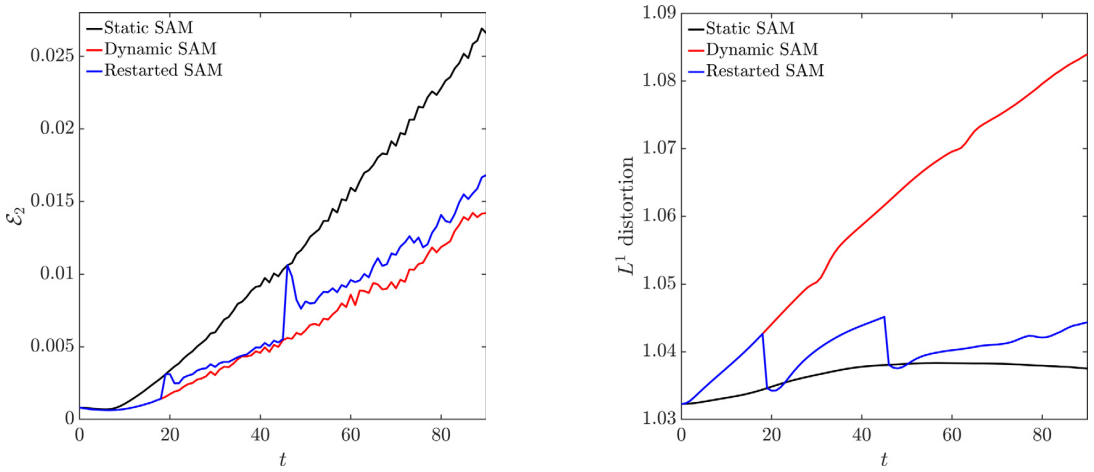
$$\omega(r)=1.6\max[(0.5-r)r,0]\,.$$

The function (36) describes the evolution of an initially smooth Gaussian blob $\bar{\mathsf{G}}(y,0)$ advected by an incompressible velocity field $V=(V_r\,,V_\theta)=(0,r\omega(r))$, where V_r and V_θ are the velocity components in the r and θ directions, respectively. The initial blob is smooth but will develop arbitrarily small scales for $t>0$ due to the radial dependence of the angular velocity. As in [18], we set the grid resolution at $N=128^2$, the time-step as $\Delta t=1$, and generate a sequence of meshes for $0\le t\le 90$.

5.4.2. Comparison of static, dynamic, and restarted SAM

The results of static, dynamic, and restarted SAM simulations are provided in Fig. 7, which shows zoomed-in plots of the meshes near $(0.5,0.5)$ at the final time $t=90$. All three schemes produce grids that are untangled, but while static SAM grids are smooth, the dynamic SAM grids contain more distorted elements. This is confirmed in Fig. 8, which provides plots of the time history of the L^2 Jacobian error (18) and L^1 distortion (30). As with the rotating patch problem, the distorted dynamic SAM grids are still more accurate than the static SAM grids, though the Jacobian errors are roughly comparable.

For restarted SAM, the restart criterion $\lambda_k>\Lambda\lambda_{\text{ref}}$, with $\Lambda=1.01$, forces the grid to reset 2 times during the simulation. As shown in Fig. 8 and in the final row of Fig. 7, the restarted SAM grids are smooth and comparable to static SAM grids, and are almost as accurate as the dynamic SAM grids. A comparison of Fig. 8 with Figure 10 in [18] shows that static and restarted SAM grids are of similar quality to the MK grids.



(a) $\mathcal{E}_2$ vs t (b) L^1 grid distortion vs t

Fig. 8. Test Problem 5.4: tracking small scale vortical structures in flows with differential rotation using (36). Shown are (a) L^2 Jacobian error $\mathcal{E}_2$ and (b) L^1 grid distortion history of the grids produced using static, dynamic, and restarted SAM. The errors are comparable to those provided in [18], and restarted SAM controls the grid distortion associated with Lagrangian-type schemes.

5.5. 3D swirling flow

5.5.1. Problem description

Our final experiment is a $3D$ dynamic version of the test in [9]. The domain is $\Omega = [0,1]^3$, the time interval is $0 \le t \le 1$, and the target Jacobian function is given by

$$\bar{\mathsf{G}}(y^1, y^2, y^3, t) = \frac{1}{1 + 5e^{-36(y^3 - \frac{1}{2})^2} \exp\left(-\omega_1 R(y^1, y^2, y^3, t)\right)}, \tag{37}$$

with

$$R(y^1, y^2, y^3, t) = \left(y^1 - \frac{1}{2} - \omega_2 \cos(4\pi\,(y^3 - t/4)\right)^2 + \left(y^2 - \frac{1}{2} - \omega_2 \sin(4\pi\,(y^3 - t/4)\right)^2,$$

and $\omega_1 = 100$ and $\omega_2 = 0.25$. As discussed in [9], the target function (37) describes a complex $3D$ helical surface and poses a major challenge for mesh generation algorithms since it leads to highly non-uniform and twisted meshes. See Fig. 9 for plots of the target function and (a portion of) the associated mesh at $t = 1$ and at $N = 128^3$ cell resolution. In Fig. 10, we provide plots of $\psi(P)$, where $P \subset \mathcal{T}_{\text{ref}}$ is some planar subset (lying in either the x^1x^2-, x^2x^3-, or x^1x^3-planes) of the reference mesh.

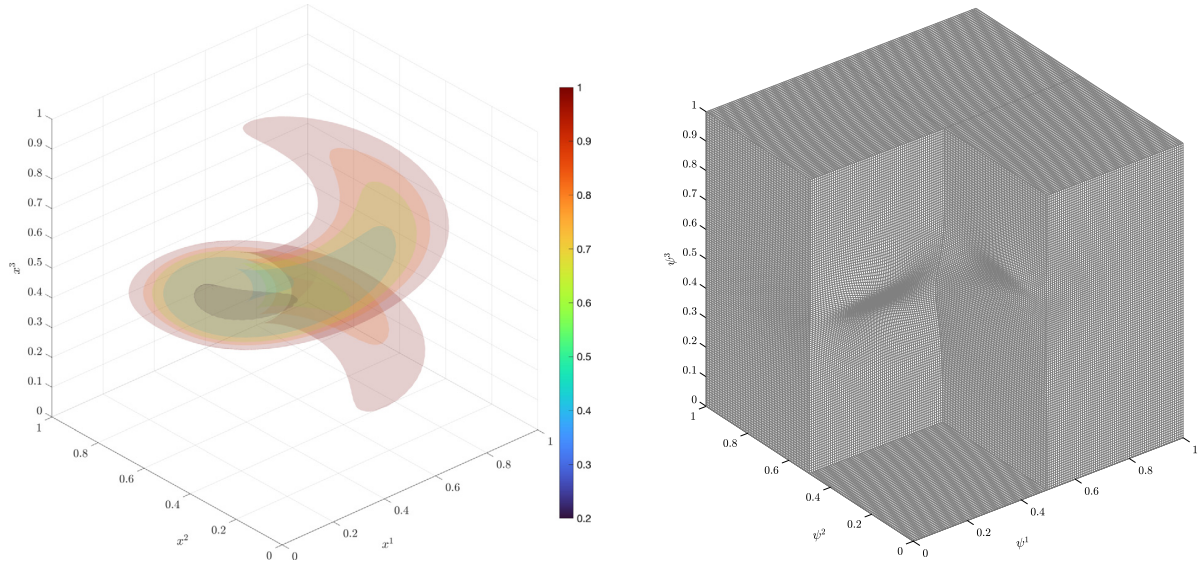
We generate a sequence of meshes starting with $N = 32^3$ resolution and doubling in each direction until $N = 256^3$. The time-step Δt depends on N according to the CFL scaling and is set as $\Delta t = \frac{2}{\sqrt[3]{N}}$. It is straightforward to adapt the $2D$ numerical scheme described in Section 3.3 to the $3D$ setting, and for brevity we omit the details.

5.5.2. $N = 128^3$ simulations using static, dynamic, and restarted SAM

Plots of the time-history of the Jacobian errors $\mathcal{E}_2$ and L^1 distortion at $N = 128^3$ are shown in Fig. 11. For $0 \le t \le 0.5$, dynamic SAM produces the smallest errors, due to the greater accuracy with which the Poisson and transport problems are solved. As expected, dynamic SAM meshes exhibit increasing grid distortion, which causes growth of the Jacobian error. The restart criterion in restarted SAM forces the mesh to reset two times during the simulation, which controls the growth of both the mesh distortion as well as the Jacobian error; for this example, $\Lambda = 1.003$.

5.5.3. Resolution study

Next, we provide in Fig. 12 plots (as a function of the resolution N) of the Jacobian error, L^1 distortion, and CPU runtime at $t = 1$. Fig. 12(a) shows that restarted SAM produces grids with the smallest Jacobian errors, but the errors for the various schemes are comparable for all the resolutions considered; as expected, we observe 4^{th} order convergence for all the schemes. Fig. 12(b) shows that the L^1 distortion for both static and dynamic SAM is consistent across resolutions, with the grid distortion for restarted SAM bounded between the two. Finally, Fig. 12(c) shows that, while static SAM is of complexity $\mathcal{O}(N \cdot N^{1/3}/\Delta t) = \mathcal{O}(N^{5/3})$, both dynamic and restarted SAM are of optimal complexity $\mathcal{O}(N/\Delta t) = \mathcal{O}(N^{4/3})$. Based on this, we can estimate that, for this test, restarted SAM becomes more efficient than static SAM for $N > 735^3$.



(a) $\mathsf{G} \circ \psi(x,t)$ at $t = 1$ (b) $\mathcal{T}(t)$ at $t = 1$

Fig. 9. Test Problem 5.5: $3D$ swirling flow with the helical target function (37). Fig. 9(a) shows isosurfaces of the target function G and Fig. 9(b) shows a portion of the corresponding mesh.

6. SAM-ALE scheme for gas dynamics

We next couple our SAM scheme to a very simple FD WENO-based ALE scheme. The purpose of this section is to demonstrate the ability of SAM-ALE to reproduce high-resolution uniform runs using fewer cells and less total CPU time. The numerical method for the ALE system of equations we use is highly simplified and not meant to be representative of the full class of ALE solvers. Nonetheless, even for the two very difficult test problems presented in Section 7, the highly simplified scheme performs remarkably well.

For the notation used in this section, we refer the reader to Section 2.

6.1. The 2D ALE-Euler system

6.1.1. Equations in Eulerian coordinates

The $2D$ compressible Euler system in Eulerian coordinates $y = (y^1, y^2) \in \Omega$ can be written in the following compact conservation-law form

$$\partial_t \mathsf{Q} + D_i \mathsf{F}^i(\mathsf{Q}) = 0, \quad (y,t) \in \Omega \times (0,T), \tag{38a}$$

$$\mathsf{Q}(y,0) = \mathsf{Q}_0(y), \quad (y,t) \in \Omega \times \{0\}. \tag{38b}$$

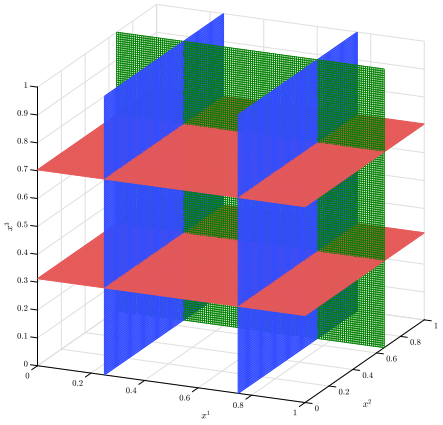
Here, Q is the vector of conserved variables, and $\mathsf{F}^1(\mathsf{Q})$ and $\mathsf{F}^2(\mathsf{Q})$ are the *flux functions*, defined as

$$\mathsf{Q} = \begin{pmatrix} \rho \\ \rho \mathsf{u}^1 \\ \rho \mathsf{u}^2 \\ \mathsf{E} \end{pmatrix} \quad \text{and} \quad \mathsf{F}^i(\mathsf{Q}) = \begin{pmatrix} \rho \mathsf{u}^i \\ \rho \mathsf{u}^1 \mathsf{u}^i + \delta^i_1 \mathsf{p} \\ \rho \mathsf{u}^2 \mathsf{u}^i + \delta^i_2 \mathsf{p} \\ \mathsf{u}^i (\mathsf{E} + \mathsf{p}) \end{pmatrix}. \tag{39}$$

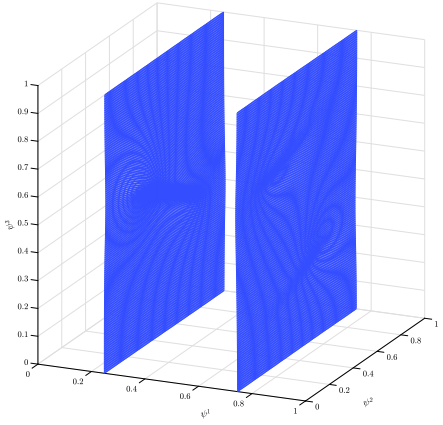
The velocity vector is $\mathsf{u} = (\mathsf{u}^1, \mathsf{u}^2)$ with horizontal component u^1 and vertical component u^2, $\rho > 0$ is the fluid density (assumed strictly positive), E denotes the energy, and p is the pressure defined by the ideal gas law,

$$\mathsf{p} = (\gamma - 1)\left(\mathsf{E} - \frac{1}{2}\rho |\mathsf{u}|^2\right), \tag{40}$$

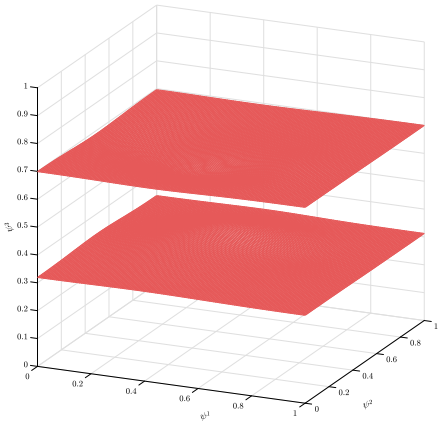
where γ is the adiabatic constant, which we will assume takes the value $\gamma = 1.4$, unless otherwise stated.



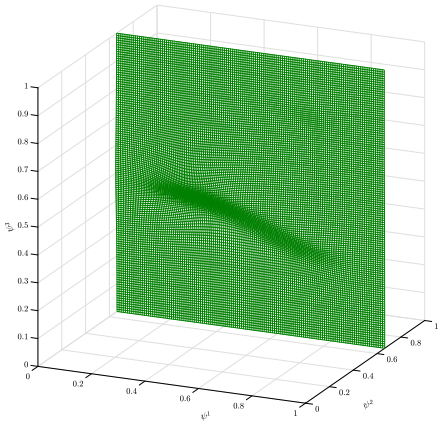
(a) Planes in $\mathcal{T}_{\text{ref}}$ showing the pre-image of the meshes in Figures 10(b) to 10(d).



(b) Image of the planes $\{x^1 = 32/128\}$ and $\{x^1 = 96/128\}$ under the map ψ.



(c) Image of the planes $\{x^3 = 40/128\}$ and $\{x^3 = 90/128\}$ under the map ψ.



(d) Image of the planes $\{x^2 = 85/128\}$ under the map ψ.

Fig. 10. Test Problem 5.5: $3D$ swirling flow with the helical target function (37). Shown are plots of the images of various planes $P \subset \mathcal{T}_{\text{ref}}$ under the map ψ.

6.1.2. Equations in ALE coordinates

Let Ω_{ref} be the fixed reference domain with coordinates (x^1, x^2), and assume that we have, for each $t \geq 0$, a smooth ALE map $\psi(\cdot, t) : \Omega_{\text{ref}} \to \Omega$. Denote by the regular font f the ALE counterpart to the Eulerian variable written with upright font f i.e. $f(x,t) = \mathsf{f} \circ \psi(x,t)$. The $2D$ ALE-Euler system can then be written in conservation law form as

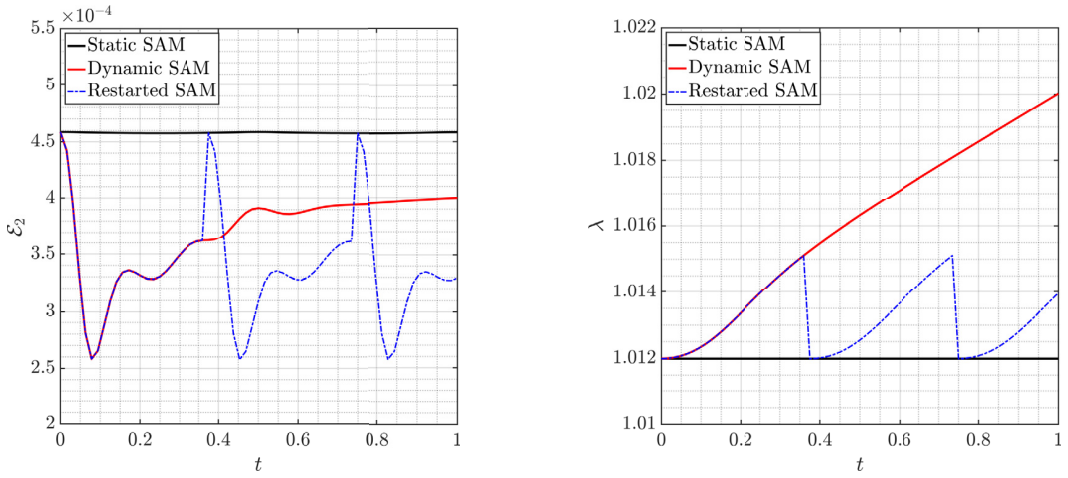
$$\partial_t Q + \partial_j F^j(Q) = 0, \quad (x,t) \in \Omega_{\text{ref}} \times (0,T), \tag{41a}$$

$$Q(x,0) = Q_0(x), \quad (x,t) \in \Omega_{\text{ref}} \times \{0\}, \tag{41b}$$

where the conserved ALE variables Q and flux functions $F^j(Q)$ are given as

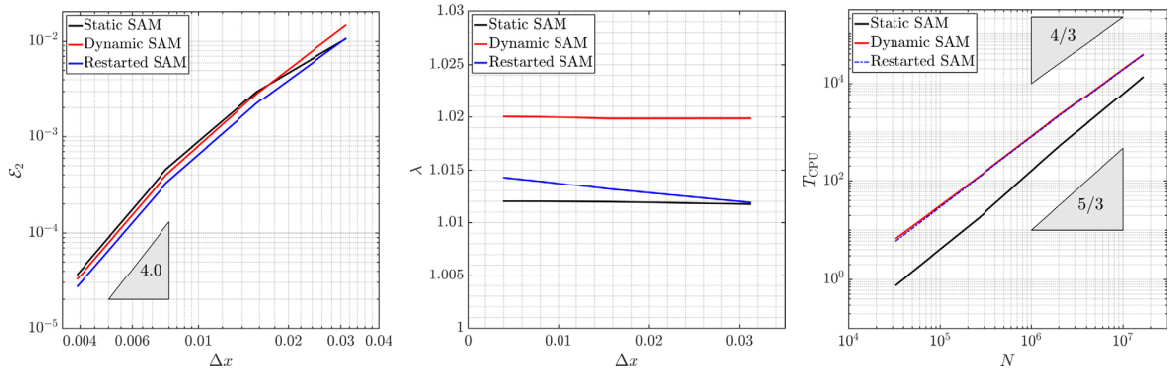
$$Q = \begin{pmatrix} \mathcal{J}\rho \\ \mathcal{J}\rho u^1 \\ \mathcal{J}\rho u^2 \\ \mathcal{J}E \end{pmatrix} \quad \text{and} \quad F^j(Q) = \begin{pmatrix} \rho a_i^j (u^i - \psi_t^i) \\ \rho u^1 a_i^j (u^i - \psi_t^i) + a_1^j p \\ \rho u^2 a_i^j (u^i - \psi_t^i) + a_2^j p \\ E a_i^j (u^i - \psi_t^i) + p a_i^j u^i \end{pmatrix}. \tag{42}$$

Here, a_i^j denotes the components of the cofactor matrix defined by (3), and ψ_t^i is the i^{th} component of the mesh velocity. It is also convenient to introduce the *ALE transport velocity* $v(x,t)$ with j^{th} component $v^j := \frac{1}{\mathcal{J}} a_i^j (u^i - \psi_t^i)$. The $2D$ ALE-Euler



(a) $\mathcal{E}_2$ vs t (b) L^1 grid distortion vs t

Fig. 11. Test Problem 5.5: $3D$ swirling flow with the helical target function (37). Shown are (a) L^2 Jacobian error $\mathcal{E}_2$ and (b) L^1 grid distortion history of the grids produced using static, dynamic, and restarted SAM at $N = 128^3$. Restarted SAM controls the grid distortion associated with Lagrangian-type schemes.



(a) log log plot $\mathcal{E}_2$ vs Δx (b) L^1 grid distortion vs Δx (c) log log plot T_{CPU} vs N

Fig. 12. Test Problem 5.5: $3D$ swirling flow with the helical target function (37). Shown are (a) log log plots of L^2 Jacobian error $\mathcal{E}_2$ vs Δx, (b) L^1 grid distortion vs Δx, and (c) log log plots of total runtime T_{CPU} vs N of the grids produced using static, dynamic, and restarted SAM for $N = 32^3, \ldots, 256^3$.

system (41) is hyperbolic in the sense that each of $\nabla_Q F^j(Q)$ is diagonalizable with real eigenvalues (or wave speeds), which are given explicitly by

$$\lambda^{j,\pm} = \frac{1}{\mathcal{J}}(v^j \pm c) \quad \text{and} \quad \lambda^{j,0} = \frac{1}{\mathcal{J}} v^j \,(\text{repeated}), \tag{43}$$

with $c = \sqrt{\gamma p/\rho}$ the sound speed.

6.1.3. Geometric conservation law and free-stream preservation

An explicit computation shows that the Jacobian determinant $\mathcal{J}(x,t)$ satisfies the *geometric conservation law* (GCL) [75]

$$\partial_t \mathcal{J} - \partial_j (a_i^j \psi_t^i) = 0\,. \tag{44}$$

For (41), an equivalent property to the GCL is the *free-stream preservation property*, which states that an initially uniform flow (i.e. $\mathsf{Q}_0 \equiv$ constant) is preserved under evolution by (41a) i.e. $\mathsf{Q} \equiv$ constant for every $t > 0$. Numerical schemes that fail to preserve the free-stream produce unacceptably large errors that corrupt small-scale vortical structures [38,79,17,60,45].

Finite difference schemes on static uniform meshes preserve the free-stream. On dynamic adaptive meshes, however, this is no longer a given, and indeed many standard schemes (including WENO [43]) fail to preserve the free-stream. As such, we design our numerical scheme to ensure free-stream preservation by explicitly incorporating (44) into the system of conservation laws to be solved [38,84]. Specifically, we append to (41) the equation (44) and consider the modified system

$$\partial_t \tilde{Q} + \partial_j \tilde{F}^j(\tilde{Q}) = 0, \quad (x,t) \in \Omega_{\text{ref}} \times (0,T), \tag{45a}$$

$$\tilde{Q}(x,0) = \tilde{Q}_0(x), \quad (x,t) \in \Omega_{\text{ref}} \times \{0\}, \tag{45b}$$

with

$$\tilde{Q} = \begin{pmatrix} \mathcal{J}\rho \\ \mathcal{J}\rho u^1 \\ \mathcal{J}\rho u^2 \\ \mathcal{J}E \\ \mathcal{J} \end{pmatrix} \quad \text{and} \quad \tilde{F}^j(\tilde{Q}) = \begin{pmatrix} \mathcal{J}\rho v^j \\ \mathcal{J}\rho u^1 v^j + a_1^j p \\ \mathcal{J}\rho u^2 v^j + a_2^j p \\ \mathcal{J}E v^j + p a_i^j u^i \\ -a_i^j \psi_t^i \end{pmatrix}. \tag{46}$$

We emphasize that, while the cofactor matrix a_i^j is computed directly from the map ψ according to (3), the Jacobian determinant $\mathcal{J}$ is computed (using the same numerical method used for the other equations in (45)) via (44) and *not* by the usual determinant formula $\mathcal{J} = \partial_1\psi^1\,\partial_2\psi^2 - \partial_1\psi^2\,\partial_2\psi^1$ *except at the initial time* $t = 0$.

6.2. The C-method for 2D ALE-Euler

Next, we describe some aspects of our numerical framework for solving (45). Specifically, we adapt the C-method, introduced in the Eulerian setting in [64,65], to the ALE setting. One of the key features of the C-method is space-time smooth tracking of shock/contact fronts and their geometries via so-called C-functions. The C-functions are space-time smoothed versions of localized solution gradients, and are found as the solutions to auxiliary scalar reaction-diffusion equations. These C-functions in turn allow us to implement both directionally isotropic (for shock stabilization) and anisotropic (for contact stabilization) artificial viscosity schemes. In particular, the C-method is a PDE-level modification of (45). Consequently, the methods developed in [64,65] can be implemented in the ALE context in a straightforward manner. For the purposes of brevity, we omit some of the details here and refer the reader to [65] and Appendix A.

6.2.1. WENO-type reconstruction and computation of a_i^j

We discretize the uniform mesh and index the nodes by $x_{r,s} = (x_r^1, x_s^2)$. At each $x_{r,s}$ we construct numerical flux functions $\hat{F}^1_{r+\frac{1}{2},s}$ and $\hat{F}^2_{r,s+\frac{1}{2}}$ that will be used to approximate the derivatives $\partial_1 \tilde{F}^1(\tilde{Q})|_{x_{r,s}}$ and $\partial_2 \tilde{F}^2(\tilde{Q})|_{x_{r,s}}$, respectively. We describe the procedure for $\hat{F}^1_{r+\frac{1}{2},s}$. For ease of notation, we drop the superscript 1 and let $\tilde{F}^1 \equiv \tilde{F}$. Decompose $\tilde{F} = \tilde{F}^v + \tilde{F}^p + \tilde{F}^E + \tilde{F}^{\mathcal{J}}$ with

$$\tilde{F}^v = \begin{pmatrix} \mathcal{J}\rho v^j \\ \mathcal{J}\rho u^1 v^j \\ \mathcal{J}\rho u^2 v^j \\ \mathcal{J}E v^j \\ 0 \end{pmatrix}, \quad \tilde{F}^p = \begin{pmatrix} 0 \\ a_1^j p \\ a_2^j p \\ 0 \\ 0 \end{pmatrix}, \quad \tilde{F}^E = \begin{pmatrix} 0 \\ 0 \\ 0 \\ p a_i^j u^i \\ 0 \end{pmatrix}, \quad \tilde{F}^{\mathcal{J}} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ -a_i^j \psi_t^i \end{pmatrix}. \tag{47}$$

Each component of the advection term $\tilde{F}^v$ is approximated at the half-point $x_{r+\frac{1}{2},s}$ as

$$\hat{F}^v_{r+\frac{1}{2},s} = \text{WENO}\left(q, \mathcal{J}v^j\right) := q_{r+\frac{1}{2},s}(\mathcal{J}v^j)_{r+\frac{1}{2},s}, \tag{48}$$

where q denotes one of the variables $q \in \{\rho, \rho u^1, \rho u^2, E\}$ and $q_{r+\frac{1}{2},s}$ is computed using a standard 5^{th} order WENO reconstruction [68] of q with upwinding based on the sign of $(\mathcal{J}v^j)_{r+\frac{1}{2},s}$. The velocity $(\mathcal{J}v^j)_{r+\frac{1}{2},s}$ is computed according to the 4^{th} order average

$$(w)_{r+\frac{1}{2},s} := \frac{-w_{r-1,s} + 7w_{r,s} + 7w_{r+1,s} - w_{r+2,s}}{12}. \tag{49}$$

The additional advection terms $\hat{F}^E_{r+\frac{1}{2},s} = \text{WENO}(p, a_i^j u^i)$ and $\hat{F}^{\mathcal{J}}_{r+\frac{1}{2},s} = \text{WENO}(1, -a_i^j \psi_t^i)$ can be approximated in a similar fashion to (48). The pressure term $\hat{F}^p_{r+\frac{1}{2},s}$ is approximated by the 4^{th} order average (49). Finally, the total flux is given by the sum $\hat{F}_{r+\frac{1}{2},s} = \hat{F}^v_{r+\frac{1}{2},s} + \hat{F}^p_{r+\frac{1}{2},s} + \hat{F}^E_{r+\frac{1}{2},s} + \hat{F}^{\mathcal{J}}_{r+\frac{1}{2},s}$. The semi-discrete scheme for (45) then reads

$$\partial_t \tilde{Q}_{r,s} + \frac{\hat{F}^1_{r+\frac{1}{2},s} - \hat{F}^1_{r-\frac{1}{2},s}}{\Delta x^1} + \frac{\hat{F}^2_{r,s+\frac{1}{2}} - \hat{F}^2_{r,s-\frac{1}{2}}}{\Delta x^2} = 0. \tag{50}$$

For free-stream flows, we have that $q_{r,s} \equiv$ constant and the scheme becomes linear, due to the linear averaging (49). In particular, it is easy to verify that the free-stream is preserved, provided the components of the cofactor matrix a_i^j are computed by 4^{th} order central differencing of the map ψ as

$$\left[\partial_1 \psi^j\right]_{r,s} = \frac{\psi^j_{r-2,s} - 8\psi^j_{r-1,s} + 8\psi^j_{r+1,s} - \psi^j_{r+2,s}}{12\Delta x^1}\,, \tag{51}$$

and similarly for $\left[\partial_2 \psi^j\right]_{r,s}$.

To confirm this, we perform a free-stream test on the 50×50 time-dependent moving-mesh defined by

$$\begin{cases} \psi^1(x^1, x^2, t) = x^1 + 0.4 \sin\left(\frac{3\pi t}{T}\right) \sin\left(\frac{3\pi}{8}(x^2+8)\right) & \text{(a)} \\ \psi^2(x^1, x^2, t) = x^2 + 0.8 \sin\left(\frac{3\pi t}{T}\right) \sin\left(\frac{3\pi}{8}(x^1+8)\right) & \text{(b)} \end{cases} \tag{52}$$

for $(x^1, x^2) \in [-8\,, +8]^2$ and $0 \le t \le T = 80$. The initial data is uniform $\mathsf{U}_0 \equiv 1$ and we employ periodic boundary conditions. The magnitude of the density error at the final time $t = T$ is $||\rho(\cdot, T) - 1||_{L^\infty} = 9.10 \times 10^{-14}$ i.e. the scheme maintains free stream flows to machine precision.

For non-smooth problems with shocks or contacts, it is necessary to add an artificial viscosity term to the right-hand side of (45a), and the semi-discrete scheme (50) must be modified appropriately. The details of the particular form of artificial viscosity we use are provided in Appendix A.

Remark 1. The simplified WENO-type reconstruction procedure outlined above is similar in some respects to the WENO schemes based on the so-called *alternative flux formulation*, first introduced in [69] and explored extensively in several recent papers [44,45,61,19,55]. In particular, both schemes define the flux $\hat{F}_{r+\frac{1}{2},s}$ by first reconstructing the variables $q_{r+\frac{1}{2},s}$. On the other hand, the alternative flux formulation WENO schemes utilize characteristic decompositions and (exact or approximate) Riemann solvers. The resulting algorithms are more expensive but also more robust. Nonetheless, for simple problems, both the simplified WENO and alternative flux WENO schemes produce similar results [64,65]. For more challenging problems, the simplified WENO scheme produces oscillatory solutions; these oscillations can be suppressed with C-method artificial viscosity.

6.2.2. Explicit interface tracking

The C-method utilizes a simple method for tracking of contact discontinuities which we first describe in the Eulerian setting i.e. for the system (38). Let $\mathsf{z} : \mathcal{I} \times [0, T] \to \Omega$ be a parametrization of the material interface with parameter $\alpha \in \mathcal{I} \subset \mathbb{R}$, and with components $\mathsf{z} = (\mathsf{z}^1\,, \mathsf{z}^2)$. In many simulations, the contact discontinuity is a closed or periodic curve, and in this case we take $\mathcal{I} = [-\pi\,, \pi]$. Given an initial parametrization z_0 of the contact discontinuity, the interface $\mathsf{z}(\alpha, t)$ is found as the solution to

$$\begin{cases} \partial_t \mathsf{z}(\alpha, t) = \bar{\mathsf{u}} \circ \mathsf{z}(\alpha, t)\,, & \alpha \in \mathcal{I} \text{ and } 0 < t \le T \quad \text{(a)} \\ \mathsf{z}(\alpha, 0) = \mathsf{z}_0(\alpha)\,, & \alpha \in \mathcal{I} \text{ and } t = 0 \quad \text{(b)} \end{cases} \tag{53}$$

Here, the velocity $\bar{\mathsf{u}}$ is defined as the average $\bar{\mathsf{u}} = \frac{1}{2}(\mathsf{u}^+ + \mathsf{u}^-)$, with $\mathsf{u}^\pm$ denoting the fluid velocity on either side of the interface. In a numerical implementation, the average $\bar{\mathsf{u}}$ is approximated by bilinear interpolation of u onto z.

The ALE analog of the (Lagrangian) interface tracking algorithm described above can be derived by defining the ALE interface parametrization $z : \mathcal{I} \times [0, T]$ as the image of z under the action of the inverse ALE map $\psi^{-1} : \Omega \times [0, T] \to \Omega_{\text{ref}}$ i.e.

$$z(\alpha, t) = \psi^{-1} \circ \mathsf{z}(\alpha, t)\,.$$

If the map ψ resolves mesh points around z, then the ALE interface z represents a "zoomed-in" version of z that magnifies small scale structures cf. Fig. 17(d).

A chain rule computation shows that z is the solution to

$$\begin{cases} \partial_t z(\alpha, t) = \bar{v} \circ z(\alpha, t)\,, & \alpha \in \mathcal{I} \text{ and } 0 < t \le T \quad \text{(a)} \\ z(\alpha, 0) = z_0(\alpha)\,, & \alpha \in \mathcal{I} \text{ and } t = 0 \quad \text{(b)} \end{cases} \tag{54}$$

where $\bar{v} = \frac{1}{2}\left(v^+ + v^-\right)$. The initial interface z_0 is defined by

$$z_0(\alpha) = \psi^{-1} \circ \mathsf{z}(\alpha, 0)\,. \tag{55}$$

In a numerical implementation, the initial ALE interface z can be computed as the roots of $\psi_0(z_0) = \mathsf{z}_0$ using e.g. Newton's method.

Algorithm 4 COUPLED SAM-ALE.

Step 0: Initialization $t = 0$.
(a) Define the initial Eulerian data Q_0 on the uniform mesh $\mathcal{U} \subset \Omega$ and the initial interface parametrization $z_0(\alpha)$.
(b) Define the initial target Jacobian function G_0 on $\mathcal{U}$. Compute the initial ALE map $\psi_0 : \Omega_{\text{ref}} \to \Omega$ and adaptive mesh $\mathcal{T}_0 = \psi_0(\mathcal{T}_{\text{ref}}) \subset \Omega$ using static SAM Algorithm 1.
(c) Define the initial ALE data Q_0. Compute the initial ALE interface z_0 using Newton's method.
Step 1: Time-stepping $t = t_k \geq 0$. Assume that we are given all quantities at $t = t_k$.
(a) Define the target Jacobian function G_{k+1} and compute the map ψ_{k+1} and adaptive mesh $\mathcal{T}_{k+1}$ according to restarted dynamic SAM Algorithm 3.
(b) Compute the cofactor matrix a_i^j using (51). Define the mesh velocity $\partial_t \psi_{k+1} = \frac{\psi_{k+1} - \psi_k}{\Delta t}$.
(c) Compute the ALE variables $\tilde{Q}_{k+1}$ and z_{k+1} using the C-method and RK4 time-stepping. The mesh, cofactor matrix, and mesh velocity are kept fixed over the time step.
(d) Compute the interface $\mathsf{z}_{k+1} = \psi_{k+1} \circ z_{k+1}$.
(e) If $t_{k+1} = T$, then stop; else, set $t = t_{k+1}$ and return to **Step 1**(a).

6.3. Coupled SAM-ALE algorithm

Our SAM algorithm is coupled to the ALE C-method by defining an appropriate target Jacobian function G_k. In this work, for simplicity, we shall assume that G_k is explicitly defined, either by some particular formula (as in the Noh test), or via the interface z_k (for the RT test). Future work will investigate coupling of SAM-ALE by means of balanced monitoring of solution gradients [77]. In the case of RT instability, it is important to use the interface z to control adaptation since it allows high mesh concentration in KH roll up zones, in contrast to the balanced monitoring approach in which the magnitudes of solution gradients decrease in KH zones due to mixing [73].

The complete SAM-ALE algorithm is provided in Algorithm 4.

7. SAM-ALE simulations of gas dynamics

7.1. Noh implosion

The first test is the $2D$ Noh implosion: an initially cold gas is directed towards the origin with speed 1 and instantaneously implodes at the origin, resulting in a radially symmetric infinite strength shock propagating outwards with speed $1/3$. This is an extremely difficult test problem and almost all codes report errors in the form of wall heating, lack of symmetry, incorrect shock speeds, or even failure to run [53]. This is the case for both Lagrangian-type codes with artificial viscosity [52,7,20], as well as AMR codes such as RAGE [30]. Extensive numerical testing in [76] showed that catastrophic anomalies occur in AMR solutions, with the anomalies persisting, or even worsening as the grid is refined. These anomalies occur due to spurious wave reflections on discontinuous grids [78,29].

7.1.1. Problem description

The domain as $\Omega = [0,1]^2$, the adiabatic constant is $\gamma = 5/3$, and the initial data is

$$\begin{bmatrix} \rho_0 \\ (\rho u^1)_0 \\ (\rho u^2)_0 \\ \mathsf{E}_0 \end{bmatrix} = \begin{bmatrix} 1 \\ -\cos(\theta) \\ -\sin(\theta) \\ 0.5 + 10^{-6}/(\gamma - 1) \end{bmatrix} \chi_{r>0} + \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0.5 + 10^{-6}/(\gamma - 1) \end{bmatrix} \chi_{r=0}\,, \tag{56}$$

where $r = |y|$ is the radial coordinate, $\theta \in [0, \frac{\pi}{2})$ is the polar angle, and χ_A is the indicator function on the set A. We employ reflecting boundary conditions on the left and bottom boundaries and use the exact solution to impose the boundary conditions at the top and right boundaries. The problem is run until the final time $T = 2$.

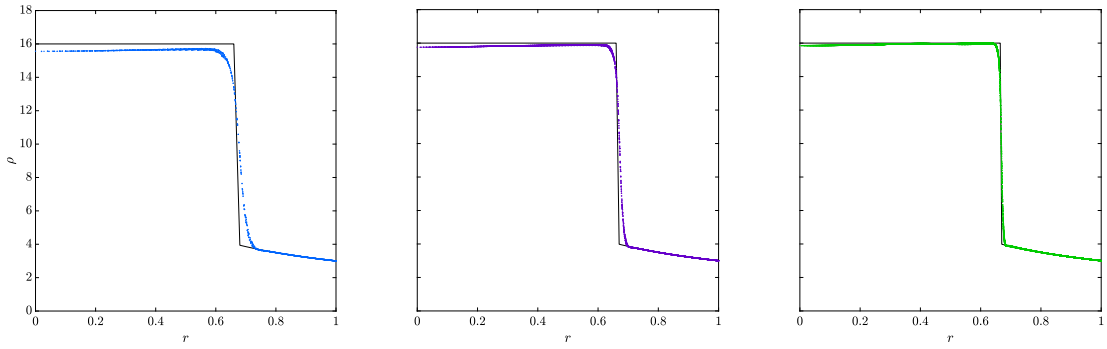
7.1.2. Uniform mesh simulations

We apply the C-method as described in [65] on 50×50, 100×100, and 200×200 meshes with time step Δt set so that CFL ≈ 0.2. The C-method artificial viscosity coefficients in (65) are fixed as $\beta_u = 0.35$, $\beta_E = 2.5$, and $\mu = 0$. The scatter plots of density vs r in Fig. 13 show that the C-method produces stable non-oscillatory solutions that maintain radial symmetry. Moreover, the smooth artificial viscosity almost entirely removes the wall-heating error in the higher resolution runs.

7.1.3. SAM-ALE simulations

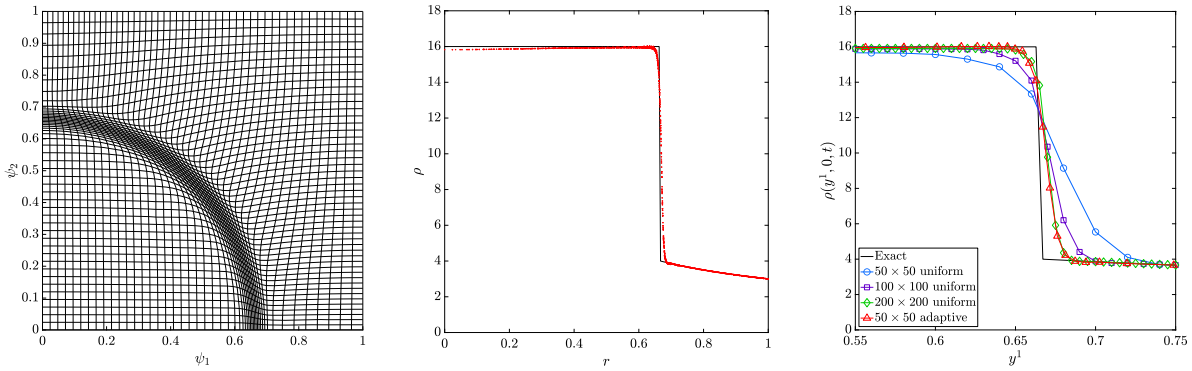
Next, we apply SAM-ALE on a 50×50 dynamic adaptive mesh. For simplicity, we choose a specially designed forcing function G for the mesh generation, defined as

$$C_\psi(y^1, y^2, t) = \exp\left[-400\left(r^2 - t^2/9\right)\right],$$
$$\bar{\mathsf{G}}(y,t) = \frac{1}{1 + \frac{\kappa}{1-\kappa}\frac{C_\psi(y,t)}{\int_\Omega C_\psi(y,t)\,dy}}. \tag{57}$$



(a) $N = 50 \times 50$ (b) $N = 100 \times 100$ (c) $N = 200 \times 200$

Fig. 13. Uniform mesh runs for the $2D$ Noh problem. Shown are the density scatter plots vs radial coordinate r. The black curve in each subfigure is the exact solution. The shock fronts are sharp and the solutions free of the spurious asymmetry, wall-heating, oscillation, and shock-racing errors associated with the majority of numerical methods for this test.



(a) $N = 50 \times 50$ SAM-ALE mesh (b) Density scatter plot (c) Density comparison

Fig. 14. SAM-ALE simulations of the Noh implosion. Shown are (a) adaptive mesh $\mathcal{T}$, (b) density scatter plot, and (c) comparison of uniform vs SAM-ALE density zoom-in at the shock. The smooth concentration and alignment of the mesh in the vicinity of the shock front allows for a sharp shock representation in the SAM-ALE solution, comparable to the high-resolution 200×200 uniform mesh solution.

This forcing function is designed, using the known analytical solution, to track the moving shock. In the future, a shock-tracking scheme analogous to the z-type advection (53) for contract tracking will be employed to define $\bar{\mathsf{G}}$. The z-type advection can track the shock with high accuracy, and the resulting $\bar{\mathsf{G}}$ is almost exactly the same as (57). As such, for simplicity we use the specially designed function (57) in this work, with the understanding that similar results can be obtained when z-type shock tracking is used instead. The particular normalization used to define $\bar{\mathsf{G}}$ is motivated by the balanced monitoring method [77]. We set $\kappa = 0.3$ and the time-step as $\Delta t = 5 \times 10^{-4}$, which yields CFL ≈ 0.2, and choose artificial viscosity parameters $\beta_u = 0.1$, $\beta_E = 0.7$, and $\mu = 0$.

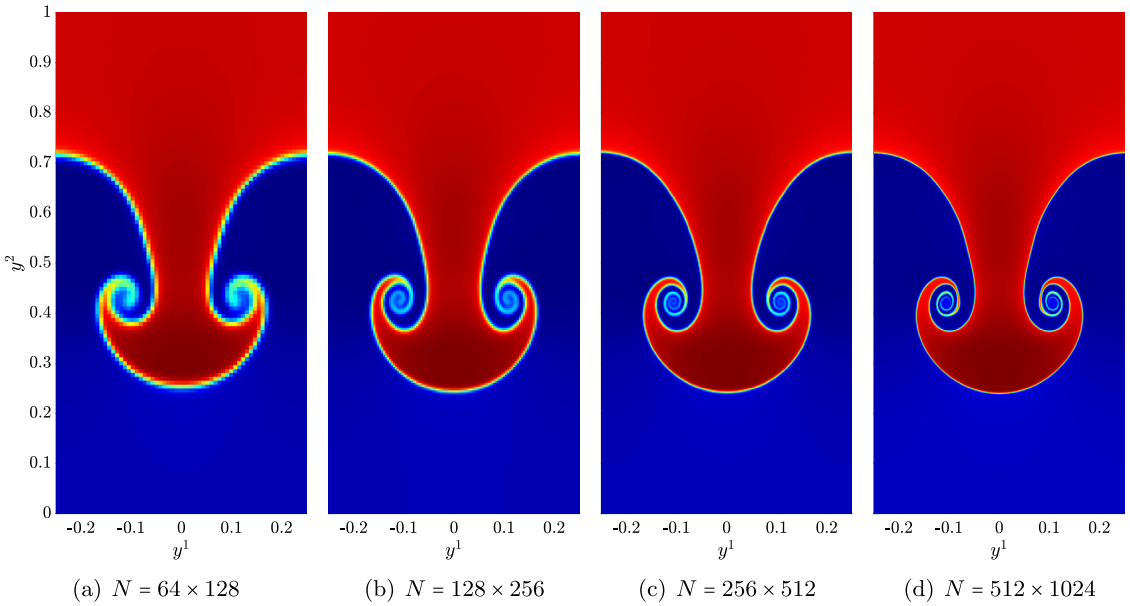
The results are shown in Fig. 14. The shock front is sharp, the wall-heating error is very small, and solution symmetry is well preserved. The latter is a consequence of both C-method artificial viscosity as well as grid alignment with the shock front. The density cross sections $\rho(y^1, 0, t)$ along the y^1-axis for the various simulations are shown in Fig. 14(c), which clearly shows that the 50×50 adaptive simulation outperforms the low-res and mid-res uniform simulations, and is comparable to the high-res uniform simulation. The wall heating error is smallest for the adaptive simulation, and the sharpness of the shock fronts for the 200×200 uniform and 50×50 adaptive simulations is comparable. As shown in Table 6, the adaptive mesh simulation produces the solution with the smallest L^2 error in the density. Moreover, the adaptive simulation is approximately 6 times faster than the high-res uniform simulation, and requires roughly the same amount of memory as the lowest-resolution uniform run.

7.2. Rayleigh-Taylor instability

Our second test problem is the classical RT instability. This test poses a huge challenge for Lagrangian and ALE methods due to the complex geometry of the evolving unstable interface. As such, limited RT ALE simulations are available in the literature (but see [85,57,24,34] for some examples). In fact, the RT problem is so challenging for ALE codes that very often

Table 6
Comparison of simulation statistics for the uniform and adaptive mesh C-method simulations for the Noh problem. The low-res SAM-ALE simulation is more accurate than the high-res uniform simulation, while running 6 times faster and requiring only 18% as much memory.

Simulation	Simulation statistic		
	L^2 density error	CPU time (secs)	Memory usage (MBs)
50×50 uniform	1.019×10^{0}	4.3	7.5
100×100 uniform	6.917×10^{-1}	35.3	14.6
200×200 uniform	5.406×10^{-1}	289	43.7
50×50 adaptive	4.897×10^{-1}	45.6	7.8



(a) $N = 64 \times 128$ (b) $N = 128 \times 256$ (c) $N = 256 \times 512$ (d) $N = 512 \times 1024$

Fig. 15. Uniform mesh simulations of RT instability with sharper fronts and more small scale structure in the KH zone as the resolution increases.

the goal is simply to perform a simulation that runs until the final time without excessive mesh tangling, at which point the simulation breaks down [57,4].

7.2.1. Problem description

We add the source term $\tilde{S}(x,t) = (0, 0, -\mathcal{J}\rho g, -\mathcal{J}\rho g u^2, 0, 0)^{\mathcal{T}}$ to the right-hand side of (45a). The domain is $\Omega = [-0.25, 0.25] \times [0, 1]$ and we apply periodic and free-flow conditions in the y^1 and y^2 directions [63]. The initial data is $u_0 = 0$, and

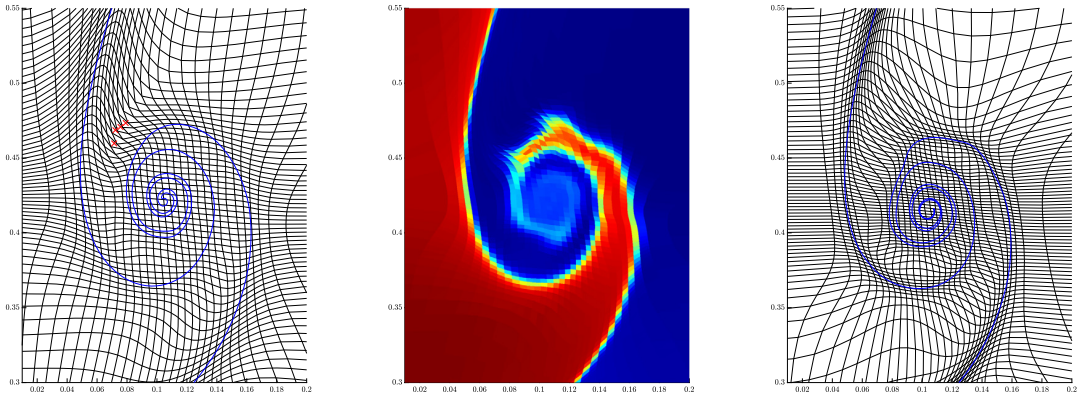
$$p_0 = \begin{cases} 5 - \rho^- g y^2 & , \text{ if } y^2 < 0.5 \\ 5 - 0.5\rho^- g - \rho^+ g(y^2 - 0.5) & , \text{ if } y^2 \geq 0.5 \end{cases}, \tag{58a}$$

$$\rho_0(y^1, y^2) = \rho^- + \frac{\rho^+ - \rho^-}{2}\left[1 + \tanh\left(\frac{y^2 - \eta_0(y^1)}{h}\right)\right], \tag{58b}$$

where $\rho^+ = 2$ and $\rho^- = 1$, $\eta_0(y^1) = 0.5 - 0.01\cos(4\pi y^1)$, $h = 0.005$, and $g = 1$. The problem is run until the final time $T = 2.5$.

7.2.2. Uniform mesh simulations

We compute a sequence of uniform mesh simulations for resolutions $N = 64 \times 128$ through $N = 512 \times 1024$ with CFL $\approx$ 0.45. The artificial viscosity parameters are set as $\mu = 7.5 \times 10^{-4}$ and $\beta_u = \beta_E = 0$, and we show heatmap plots of the density in Fig. 15. As the resolution is increased, more small-scale structure can be seen in the main KH roll up region. The artificial viscosity term suppresses further secondary instabilities that usually occur with other dimensionally split numerical methods [53,1].



(a) $\delta = 0.85$ (b) $\delta = 0.92$ (c) $\delta = 0.97$

Fig. 16. 64×128 adaptive mesh simulations of RT with large zoom-in factor. Figure (a) is a zoom-in of the mesh computed with restarted SAM and $\delta = 0.85$. The interface z is shown as the blue curve, and the non-convex elements are indicated by red crosses. Figure (b) is a zoom-in of the density with $\delta = 0.92$. The non-convex elements cause spurious instabilities along the interface. Figure (c) shows the mesh computed with the large zoom-in algorithm; all the elements are convex and the mesh is smooth.

Table 7
Total CPU runtime for uniform and adaptive simulations of RT instability.

Runtime (sec)	Cells				
	64×128	128×256	256×512	512×1024	64×128 SAM-ALE
T_{CPU}	2.21×10^1	1.67×10^2	1.37×10^3	1.21×10^4	1.38×10^2

7.2.3. Mesh generation with large zoom-in factor

Next, we aim to produce a 64×128 adaptive mesh with large zoom-in factor that resolves around the material interface z and define a target Jacobian function as

$$\mathsf{G}_\delta(y,t) = 1 - \delta \exp\left(-\left|\sigma \min_\alpha |y - \mathsf{z}(\alpha,t)|\right|^2\right), \tag{59}$$

with $\sigma = 25$. For this resolution, the meshes produced with dynamic SAM contain non-convex elements for δ larger than approximately 0.85, as shown in Fig. 16(a). These non-convex elements arise due to a strong cusp-type flow in the region between the "stem" of the mushroom and the roll up region. The choice $\delta = 0.85$ produces a mesh with smallest cell size only approximately 3.8 times smaller than a uniform mesh cell. Increasing the value of δ further produces a mesh with more non-convex elements, which in turn causes spurious errors in the computed numerical solution as shown in Fig. 16(b).

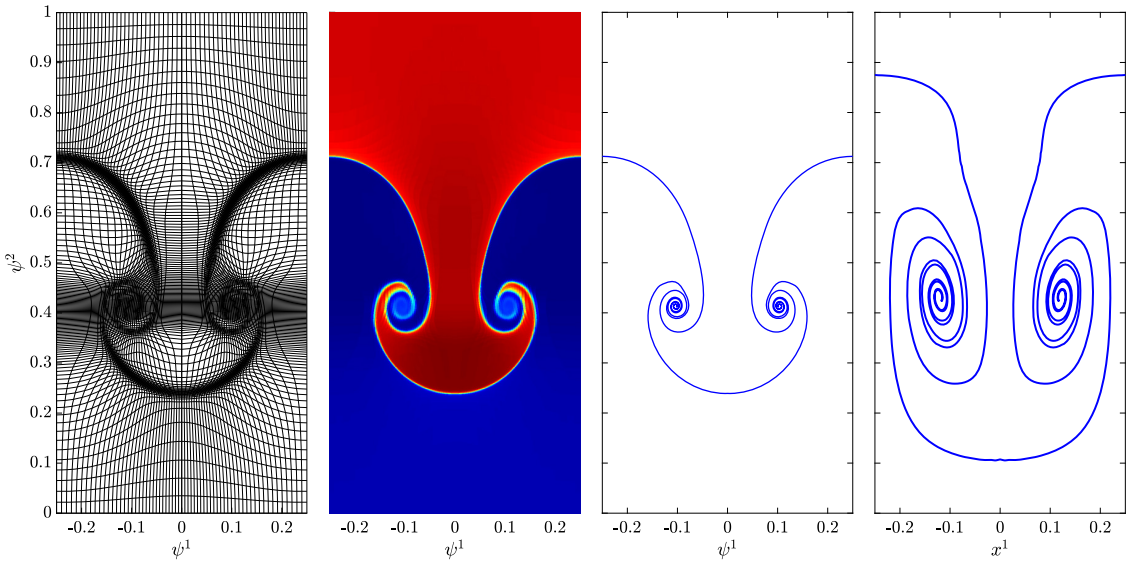
A simple technique to resolve this issue is to use the large zoom-in algorithm described in Section 5.1. Specifically, we use the large zoom-in algorithm (with 25 sub time steps) in combination with restarted dynamic SAM. The 64×128 adaptive mesh with $\delta = 0.97$ is shown in Fig. 16(c), from which it can be seen that the mesh is smooth and all elements are convex. The smallest cell size in the mesh is approximately 13 times smaller than a uniform cell. The large zoom-in algorithm is applied only when the mesh resets, and the increase in CPU runtime is therefore negligible.

7.2.4. Comparison of adaptive and uniform simulations

We perform a 64×128 cell SAM-ALE simulation with zoom-in parameter $\delta = 0.97$ and $\Delta t = 1.5625 \times 10^{-4}$. Plots of the adaptive mesh and density heatmap are provided in Fig. 17(a) and Fig. 17(b), and we refer to Fig. 16(c) for the mesh zoom-in. A comparison with the uniform mesh simulations in Fig. 15 shows that the 64×128 SAM-ALE simulation has a much sharper interface and exhibits more small-scale roll-up than the 64×128 uniform simulation, and is roughly comparable to the $N = 256 \times 512$ simulation. However, some of the small-scale structure is not observed in the SAM-ALE density. Interestingly, this roll up is captured by the interface z, shown in Fig. 17(c). This suggests that a more robust ALE solver (e.g. WENO with alternative flux formulation) may produce improved results.[10] The ALE interface z is shown in Fig. 17(d) and is clearly a zoomed-in version of z, with the small scale KH zones magnified and represented over a much larger region.

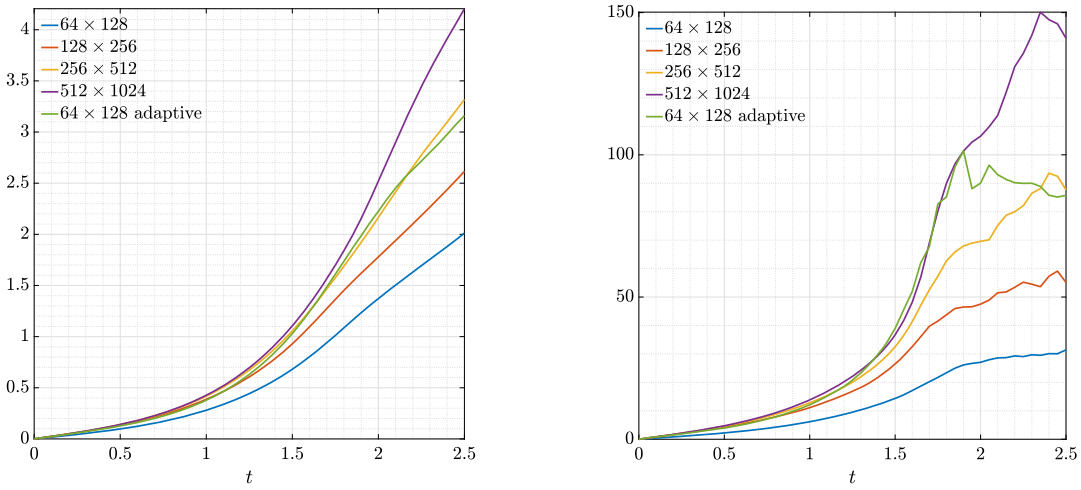
The CPU runtimes of the various simulations are provided in Table 7, from which we see that the SAM-ALE simulation is approximately 10 times and 88 times faster than the 256×512 and 512×1024 uniform runs, respectively. For this problem, the CPU time spent on mesh generation is roughly the same as the time spent on ALE calculations. Since SAM is

[10] See also [77] for a comparison of Lax-Friedrichs vs low dissipation HLLC flux reconstruction in the FV framework.



(a) Adaptive mesh $\mathcal{T}$ (b) Density ρ on $\mathcal{T}$ (c) Interface z (d) ALE interface z

Fig. 17. 64×128 SAM-ALE simulation of RT instability with $\delta = 0.97$.



(a) $||\omega(\cdot,t)||_{L^2}$ vs t (b) $||\omega(\cdot,t)||_{L^\infty}$ vs t

Fig. 18. Time history of the L^2 and L^∞ norms of the vorticity for uniform and adaptive mesh simulations of RT instability.

roughly 100-200 times faster than MK mesh generation, it is clear that an MK-ALE scheme cannot provide a speed-up over uniform mesh simulations. On the other hand, the use of a more robust ALE solver can only improve the relative efficiency of SAM-ALE, since the main computational expense will be the ALE calculations rather than mesh generation.

The time histories of the L^2 and L^∞ norms of the vorticity ω for the uniform and adaptive mesh simulations are shown in Fig. 18. These figures confirm that the 64×128 SAM-ALE run is comparable to the 256×512 uniform run. In fact, for $t \leq 1.75$, when the mesh zoom-in factor is approximately 20 times, the 64×128 SAM-ALE run closely approximates the 512×1024 uniform run. For $t > 1.75$, the mesh zoom-in factor decreases due to the stretching of the interface and the adaptive mesh is no longer able to capture the smallest scales that are present in the 512×1024 run. The decrease in the mesh zoom-in factor is a consequence of the fact that the number of cells in the mesh are fixed. So-called h-r adaptive mesh methods [23] are a way to overcome this issue; the simplicity of our algorithmic framework suggests that a dynamic h-r method based on SAM can be readily formulated and implemented, and this will be investigated in future work.

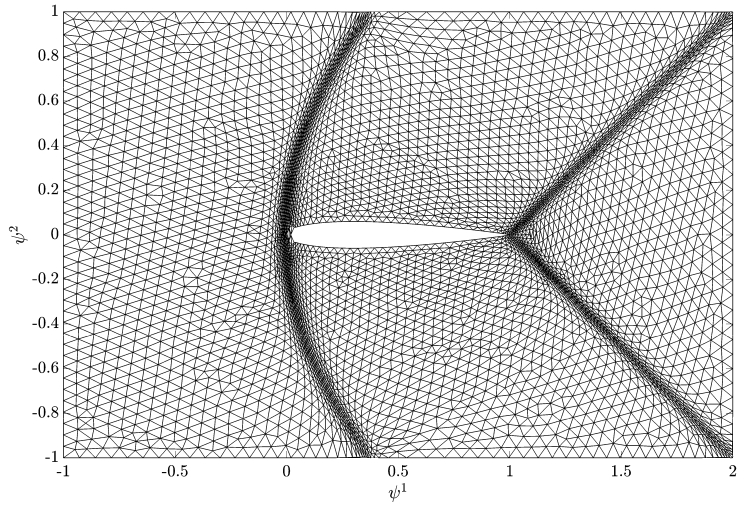


Fig. 19. Unstructured mesh modeling compressible flow past an airfoil. The mesh is constructed using (a preliminary version of) unstructured SAM in the Firedrake framework [66].

8. Concluding remarks

In this work, we developed a new Smooth Adaptive Meshing (SAM) algorithm based on a new perturbation formulation and implementation of the deformation method. The resulting numerical algorithm is simple, stable, automated, high-order accurate, and able to generate smooth and untangled meshes resolving around complex multi-D flows. We coupled SAM to a simple ALE scheme for gas dynamics and presented adaptive-simulation speed-up results for the challenging Noh and Rayleigh-Taylor problems.

Several aspects of our SAM formulation and algorithm require further investigation and improvement. As discussed in Section 7.2, we are interested in developing an h-r-refinement scheme based on SAM and, more generally, a dynamic SAM algorithm on general unstructured meshes. The numerical implementation of unstructured SAM is obviously more delicate than the simple uniform-mesh scheme presented in the current paper, and will be thoroughly investigated in future work. Nonetheless, we provide in Fig. 19 a preliminary result showing an unstructured SAM mesh that models compressible flow past an airfoil. This mesh was produced[11] within the finite-element based Firedrake code [66]. In the future, we will investigate the theoretical properties of SAM solutions on general domains, and their connections to the regularity of the discrete mesh $\mathcal{T}$.

CRediT authorship contribution statement

Raaghav Ramani: Conceptualization, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. **Steve Shkoller:** Conceptualization, Methodology, Writing – original draft, Writing – review & editing.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Raaghav Ramani reports equipment, drugs, or supplies was provided by University of New Mexico Center for Advanced Research Computing.

Data availability

No data was used for the research described in the article.

Acknowledgements

Research reported in this publication was supported by NSF grant DMS-2007606 and DTRA grant HDTRA11810022. This research was also supported by Office of Defense Nuclear Nonproliferation, NA-22 and NA-24; we note that the views of the authors do not necessarily reflect the views of the USG. This work was also supported by the Laboratory Directed Research

[11] We express our gratitude to Dr. Mariana Clare for her assistance with writing the code and generating the result shown in Fig. 19.

and Development Program of the Los Alamos National Laboratory, which is under the auspices of the National Nuclear Security Administration of the U.S. Department of Energy under DOE Contracts W-7405-ENG-36 and LA-UR-10-04291.

We would like to thank the UNM Center for Advanced Research Computing, supported in part by the National Science Foundation, for providing high performance computing resources used in this work.

We would like to express our gratitude to the anonymous referees for their numerous suggestions that have greatly improved the manuscript.

Appendix A. The C-method for 2D ALE-Euler

We provide a brief review of the C-method for adding space-time smooth artificial viscosity to shocks and contacts [65]. The most important feature of the C-method is smooth tracking of shock/contact fronts and their geometries via so-called C-functions. The C-functions are space-time smoothed versions of localized solution gradients, and are found as the solutions to auxiliary scalar reaction-diffusion equations. Specifically, we use C to denote a smoothed shock tracking function, and τ to denote the vector-valued function $\tau = (\tau^1, \tau^2)$. The function $\vec{\tau}$ is a smoothed version of the tangent vector to an evolving contact discontinuity. These C-functions allow us to implement both directionally isotropic (for shock stabilization) and anisotropic (for contact stabilization) artificial viscosity schemes.

To summarize the method, it is convenient to introduce advection, artificial viscosity, and C-equation operators as follows.

A.0.1. ALE advection operator

For a scalar function $Q : \Omega_{\text{ref}} \to \mathbb{R}$, and a vector-valued function $v : \Omega_{\text{ref}} \to \mathbb{R}^2$, define

$$\mathcal{A}[Q\,;v] \coloneqq \partial_k \left(Q\, a_l^k v^l \right). \tag{60}$$

A.0.2. ALE isotropic artificial viscosity operator

For a scalar function $Q : \Omega_{\text{ref}} \to \mathbb{R}$, define

$$\mathcal{D}[Q\,;\beta] \coloneqq \partial_k \left(\tilde{\beta}\rho C\, a_i^k a_i^l\, \partial_l Q \right), \tag{61}$$

with

$$\tilde{\beta} = \frac{|\Delta x|^2}{\max C}\beta\,.$$

The constant β is an isotropic artificial viscosity parameter for shock stabilization.

A.0.3. ALE anisotropic artificial viscosity operator

For a scalar function $Q : \Omega_{\text{ref}} \to \mathbb{R}$, we define

$$\mathcal{D}^\tau[Q\,;\mu] \coloneqq \partial_k \left[\tilde{\mu}\rho\, \tau^i \tau^j a_i^k a_j^l\, \partial_l Q \right], \tag{62}$$

with

$$\tilde{\mu} = \frac{|\Delta x|^2}{\alpha^2}\mu\,. \tag{63}$$

Here, μ is the anisotropic artificial viscosity parameter for contact discontinuity stabilization and $\alpha = \max_x\{|\tau^1|, |\tau^2|\}$.

A.0.4. ALE C-equation operator

For a scalar function $H : \Omega_{\text{ref}} \to \mathbb{R}$ and scalar forcing function $Q : \Omega_{\text{ref}} \to \mathbb{R}$, let

$$\mathcal{L}[H\,;Q] \coloneqq \frac{\mathcal{S}}{\varepsilon|\Delta x|}(Q - H) + \kappa\mathcal{S}|\Delta x|\Delta H\,. \tag{64}$$

A.0.5. The complete ALE-Euler-C system

Now, we can write the full ALE-Euler-C system as

$$\partial_t(\mathcal{J}\rho) + \mathcal{A}[\rho\,;u-\psi_t] = 0\,, \tag{65a}$$

$$\partial_t(\mathcal{J}\rho u^r) + \mathcal{A}\left[\rho u^r\,;u-\psi_t\right] = \mathcal{D}^\tau[u^r\,;\mu] + \mathcal{D}[u^r\,;\beta_u] - \partial_j(a_r^j p)\,, \qquad \text{for } r=1,2, \tag{65b}$$

$$\partial_t(\mathcal{J}E) + \mathcal{A}[E\,;u-\psi_t] + \mathcal{A}[p\,;u] = \mathcal{D}[E/\rho\,;\beta_E]\,, \tag{65c}$$

$$\partial_t\mathcal{J} - \mathcal{A}[1\,;\psi_t] = 0\,, \tag{65d}$$

$$\partial_t C - \mathcal{L}[C\,;F] = 0\,, \tag{65e}$$

$$\partial_t\tau^r - \mathcal{L}\left[\tau^r\,;F^r\right] = 0\,, \qquad \text{for } r=1,2. \tag{65f}$$

The forcing functions for (65e) and (65f) are defined as follows. The shock C forcing function is given by

$$\hat{F} = \frac{|\frac{1}{\mathcal{J}} a_i^j \partial_j \rho|}{\max |\frac{1}{\mathcal{J}} a_i^j \partial_j \rho|}, \tag{66}$$

while the components of the forcing to the contact tangent vector τ equations are defined by

$$F^1 = -\frac{1}{\mathcal{J}} a_2^j \partial_j \rho \qquad \text{and} \qquad F^2 = \frac{1}{\mathcal{J}} a_1^j \partial_j \rho \,. \tag{67}$$

The initial conditions for C and τ are defined by solving the time-independent versions of (65e) and (65f).

Appendix B. Boundary smoothing for non-Neumann functions

Herein, we describe a simple boundary smoothing technique for non-Neumann functions. Let $x^r_{\text{mid}} = \frac{1}{2}\left(x^r_{\text{min}} + x^r_{\text{max}}\right)$, for $r = 1, 2$. Define smooth cutoff functions

$$\phi^1(\xi) = \frac{1}{2}\left[\tanh\left(\frac{\xi - (x^1_{\text{min}} + d_1)}{\varepsilon}\right) - \tanh\left(\frac{\xi - (x^1_{\text{max}} - d_1)}{\varepsilon}\right)\right],$$
$$\phi^2(\eta) = \frac{1}{2}\left[\tanh\left(\frac{\eta - (x^2_{\text{min}} + d_2)}{\varepsilon}\right) - \tanh\left(\frac{\eta - (x^2_{\text{max}} - d_2)}{\varepsilon}\right)\right],$$

where ε is a smoothing parameter, which we choose as $\varepsilon = 0.02$. The function ϕ^1 is equal to 1 in the interior of the domain, then smoothly decreases to 0 at a distance d_1 near the left and right boundaries. The function ϕ^2 behaves similarly. We set $d_r = 0.05(x^r_{\text{max}} - x^r_{\text{min}})$.

Given a non-Neumann function G, we first compute the derivatives $D_1\mathsf{G}$, $D_2\mathsf{G}$, and $D_{12}\mathsf{G}$. We then compute

$$\mathcal{I}^{(1)}(y^1) = \int_{x^1_{\text{mid}}}^{y^1} \phi^1(\xi) D_1\mathsf{G}(\xi, x^2_{\text{mid}})\, \mathrm{d}\xi \,,$$
$$\mathcal{I}^{(2)}(y^2) = \int_{x^2_{\text{mid}}}^{y^2} \phi^2(\eta) D_2\mathsf{G}(x^1_{\text{mid}}, \eta)\, \mathrm{d}\eta \,,$$
$$\mathcal{I}^{(3)}(y^1, y^2) = \int_{x^2_{\text{mid}}}^{y^2} \int_{x^1_{\text{mid}}}^{y^1} \phi^1(\xi) D_{12}\mathsf{G}(\xi, \eta)\, \mathrm{d}\xi \mathrm{d}\eta \,,$$

and define

$$\mathsf{G}^*(y^1, y^2) \coloneqq \mathsf{G}(x^1_{\text{mid}}, x^2_{\text{mid}}) + \mathcal{I}^{(1)}(y^1) + \mathcal{I}^{(2)}(y^2) + \mathcal{I}^{(3)}(y^1, y^2) \,.$$

The function G^* then satisfies $D\mathsf{G}^* \cdot \nu = 0$ on $\partial\Omega$.

Appendix C. The MK scheme

The MK scheme solves for the unique [8,16] diffeomorphism ψ satisfying (7) that minimizes the L^2 displacement $||\psi(x) - x||_{L^2}$. The MK formulation is developed by writing $\psi = x + \nabla\Psi$, where Ψ is a scalar potential. The equation governing Ψ is found by minimizing a functional consisting of the L^2 displacement and a local Lagrange multiplier, where the latter is used to enforce the Jacobian constraint (7). The resulting equation for Ψ is fully nonlinear, and the MK scheme uses an iterative Newton-Krylov solver with multigrid preconditioning to find an approximation to the solution Ψ, within some error tolerance ϵ.

C.1. Machine comparison

To reliably compare the runtimes of our static SAM Algorithm 1 with the MK scheme as listed in [22], we need to account for the different machines on which these codes were run. Therefore, we perform the following machine comparison experiment. In [22], the authors also report the CPU runtimes for a deformation method of Liao and Anderson [50], whose description is provided in the Appendix of [22]. We coded a numerical implementation of this method, which we refer to

Table 8
CPU runtimes for the LA scheme on the machine from [22] and the LA scheme on our machine. The data for the LA scheme in the top row is taken from Table 3 of [22].

Scheme		Cells				
		16×16	32×32	64×64	128×128	256×256
LA on [22] machine	T_{CPU}	0.2	0.9	3.4	13.6	55.0
LA on our machine	T_{CPU}	0.12	0.41	1.53	6.22	24.16
	speed-up factor	1.7	2.2	2.2	2.2	2.3

as LA, and ran the numerical experiments from [22] on our machine. The runtimes for LA on our machine, along with the LA runtimes from Table 3 of [22], are shown in Table 8. These data show that our machine runs approximately 2.2 times faster than the machine on which the MK simulations in [22] were performed.

References

[1] A.S. Almgren, V.E. Beckner, J.B. Bell, M.S. Day, L.H. Howell, C.C. Joggerst, M.J. Lijewski, A. Nonaka, M. Singer, M. Zingale, CASTRO: A new compressible astrophysical solver. I. Hydrodynamics and self-gravity, Astrophys. J. 715 (2) (may 2010) 1221–1238, https://doi.org/10.1088/0004-637x/715/2/1221.

[2] A. Averbuch, M. Israeli, L. Vozovoi, A fast Poisson solver of arbitrary order accuracy in rectangular regions, SIAM J. Sci. Comput. 19 (3) (1998) 933–952, https://doi.org/10.1137/S1064827595288589.

[3] B.N. Azarenok, S.A. Ivanenko, T. Tang, Adaptive mesh redistribution method based on Godunov's scheme, Commun. Math. Sci. 1 (1) (2003) 152–179.

[4] A.J. Barlow, P.-H. Maire, W.J. Rider, R.N. Rieben, M.J. Shashkov, Arbitrary Lagrangian Eulerian methods for modeling high-speed compressible multimaterial flows, J. Comput. Phys. (ISSN 0021-9991) 322 (2016) 603–665, https://doi.org/10.1016/j.jcp.2016.07.001, https://www.sciencedirect.com/science/article/pii/S0021999116302807.

[5] M. Berger, P. Colella, Local adaptive mesh refinement for shock hydrodynamics, J. Comput. Phys. (ISSN 0021-9991) 82 (1) (1989) 64–84, https://doi.org/10.1016/0021-9991(89)90035-1, https://www.sciencedirect.com/science/article/pii/0021999189900351.

[6] J. Brackbill, J. Saltzman, Adaptive zoning for singular problems in two dimensions, J. Comput. Phys. (ISSN 0021-9991) 46 (3) (1982) 342–368, https://doi.org/10.1016/0021-9991(82)90020-1, https://www.sciencedirect.com/science/article/pii/0021999182900201.

[7] J. Breil, Numerical Methods for Lagrangian and Arbitrary-Lagrangian-Eulerian Hydrodynamic Contribution to the simulation of High-Energy-Density-Physics Problems, Habilitation à diriger des recherches, Univresité de Bordeaux, June 2016, https://hal.archives-ouvertes.fr/tel-01467157.

[8] Y. Brenier, Polar factorization and monotone rearrangement of vector-valued functions, Commun. Pure Appl. Math. (ISSN 0010-3640) 44 (4) (1991) 375–417, https://doi.org/10.1002/cpa.3160440402.

[9] P. Browne, C. Budd, C. Piccolo, M. Cullen, Fast three dimensional r-adaptive mesh redistribution, J. Comput. Phys. (ISSN 0021-9991) 275 (2014) 174–196, https://doi.org/10.1016/j.jcp.2014.06.009, https://www.sciencedirect.com/science/article/pii/S0021999114004161.

[10] G.L. Bryan, M.L. Norman, B.W. O'hea, T. Abel, J.H. Wise, M.J. Turk, D.R. Reynolds, D.C. Collins, P. Wang, S.W. Skillman, B. Smith, R.P. Harkness, J. Bordner, J. hoon Kim, M. Kuhlen, H. Xu, N. Goldbaum, C. Hummels, A.G. Kritsuk, E. Tasker, S. Skory, C.M. Simpson, O. Hahn, J.S. Oishi, G.C. So, F. Zhao, R. Cen, Y. L., ENZO: an adaptive mesh refinement code for astrophysics, Astrophys. J. Suppl. Ser. 211 (2) (mar 2014) 19, https://doi.org/10.1088/0067-0049/211/2/19.

[11] C. Budd, B. Leimkuhler, M. Piggott, Scaling invariance and adaptivity, Appl. Numer. Math. (ISSN 0168-9274) 39 (3) (2001) 261–288, https://doi.org/10.1016/S0168-9274(00)00036-2, https://www.sciencedirect.com/science/article/pii/S0168927400000362.

[12] C. Budd, R. Russell, E. Walsh, The geometry of r-adaptive meshes generated using optimal transport methods, J. Comput. Phys. (ISSN 0021-9991) 282 (2015) 113–137, https://doi.org/10.1016/j.jcp.2014.11.007, https://www.sciencedirect.com/science/article/pii/S0021999114007591.

[13] C.J. Budd, W. Huang, R.D. Russell, Moving mesh methods for problems with blow-up, SIAM J. Sci. Comput. 17 (2) (1996) 305–327, https://doi.org/10.1137/S1064827594272025.

[14] C.J. Budd, W. Huang, R.D. Russell, Adaptivity with moving grids, Acta Numer. 18 (2009) 111–241, https://doi.org/10.1017/S0962492906400015.

[15] C.J. Budd, A.T. McRae, C.J. Cotter, The scaling and skewness of optimally transported meshes on the sphere, J. Comput. Phys. (ISSN 0021-9991) 375 (2018) 540–564, https://doi.org/10.1016/j.jcp.2018.08.028, https://www.sciencedirect.com/science/article/pii/S0021999118305515.

[16] L.A. Caffarelli, Interior $W^{2,p}$ estimates for solutions of the Monge-Ampère equation, Ann. Math. (2) (ISSN 0003-486X) 131 (1) (1990) 135–150, https://doi.org/10.2307/1971510.

[17] X. Cai, F. Ladeinde, Performance of weno scheme in generalized curvilinear coordinate systems, in: 46th AIAA Aerospace Sciences Meeting and Exhibit, 2008, p. 36.

[18] L. Chacón, G. Delzanno, J. Finn, Robust, multidimensional mesh-motion based on Monge-Kantorovich equidistribution, J. Comput. Phys. (ISSN 0021-9991) 230 (1) (2011) 87–103, https://doi.org/10.1016/j.jcp.2010.09.013, https://www.sciencedirect.com/science/article/pii/S0021999110005073.

[19] A.J. Christlieb, X. Feng, Y. Jiang, Q. Tang, A high-order finite difference WENO scheme for ideal magnetohydrodynamics on curvilinear meshes, SIAM J. Sci. Comput. 40 (4) (2018) A2631–A2666, https://doi.org/10.1137/17M115757X.

[20] A.W. Cook, M.S. Ulitsky, D.S. Miller, Hyperviscosity for unstructured ale meshes, Int. J. Comput. Fluid Dyn. 27 (1) (2013) 32–50, https://doi.org/10.1080/10618562.2012.756477.

[21] B. Dacorogna, J. Moser, On a partial differential equation involving the jacobian determinant, Ann. Inst. Henri Poincaré, Anal. Non Linéaire 7 (1) (1990) 1–26, http://eudml.org/doc/78211.

[22] G. Delzanno, L. Chacón, J. Finn, Y. Chung, G. Lapenta, An optimal robust equidistribution method for two-dimensional grid adaptation based on Monge-Kantorovich optimization, J. Comput. Phys. (ISSN 0021-9991) 227 (23) (2008) 9841–9864, https://doi.org/10.1016/j.jcp.2008.07.020, https://www.sciencedirect.com/science/article/pii/S0021999108004105.

[23] V. Dobrev, P. Knupp, T. Kolev, K. Mittal, V. Tomov, hr-adaptivity for nonconforming high-order meshes with the target matrix optimization paradigm, Eng. Comput. (2021) 1–17, https://doi.org/10.1007/s00366-021-01407-6.

[24] V.A. Dobrev, T.V. Kolev, R.N. Rieben, High-order curvilinear finite element methods for lagrangian hydrodynamics, SIAM J. Sci. Comput. 34 (5) (2012) B606–B641, https://doi.org/10.1137/120864672.

[25] J. Duan, H. Tang, Entropy stable adaptive moving mesh schemes for 2d and 3d special relativistic hydrodynamics, J. Comput. Phys. (ISSN 0021-9991) 426 (2021) 109949, https://doi.org/10.1016/j.jcp.2020.109949, https://www.sciencedirect.com/science/article/pii/S0021999120307233.

[26] J. Duan, H. Tang, High-order accurate entropy stable adaptive moving mesh finite difference schemes for special relativistic (magneto)hydrodynamics, J. Comput. Phys. (ISSN 0021-9991) 456 (2022) 111038, https://doi.org/10.1016/j.jcp.2022.111038, https://www.sciencedirect.com/science/article/pii/S0021999122001000.

[27] A.S. Dvinsky, Adaptive grid generation from harmonic maps on Riemannian manifolds, J. Comput. Phys. (ISSN 0021-9991) 95 (2) (1991) 450–476, https://doi.org/10.1016/0021-9991(91)90285-S, https://www.sciencedirect.com/science/article/pii/002199919190285S.

[28] H. Feng, S. Zhao, FFT-based high order central difference schemes for three-dimensional Poisson's equation with various types of boundary conditions, J. Comput. Phys. 410 (June 2020) 109391, https://doi.org/10.1016/j.jcp.2020.109391.
[29] B. Fryxell, K. Olson, P. Ricker, F.X. Timmes, M. Zingale, D.Q. Lamb, P. MacNeice, R. Rosner, J.W. Truran, H. Tufo. FLASH, An adaptive mesh hydrodynamics code for modeling astrophysical thermonuclear flashes, Astrophys. J. Suppl. Ser. 131 (1) (nov 2000) 273–334, https://doi.org/10.1086/317361.
[30] M. Gittings, R. Weaver, M. Clover, T. Betlach, N. Byrne, R. Coker, E. Dendy, R. Hueckstaedt, K. New, W.R. Oakes, D. Ranta, R. Stefan, The RAGE radiation-hydrodynamic code, Comput. Sci. Discov. 1 (1) (nov 2008) 015005, https://doi.org/10.1088/1749-4699/1/1/015005.
[31] M. Grajewski, M. Köster, S. Turek, Mathematical and numerical analysis of a robust and efficient grid deformation method in the finite element context, SIAM J. Sci. Comput. 31 (2) (2009) 1539–1557, https://doi.org/10.1137/050639387.
[32] M. Grajewski, M. Köster, S. Turek, Numerical analysis and implementational aspects of a new multilevel grid deformation method, Appl. Numer. Math. (ISSN 0168-9274) 60 (8) (2010) 767–781, https://doi.org/10.1016/j.apnum.2010.03.017, https://www.sciencedirect.com/science/article/pii/S0168927410000474.
[33] P. Grisvard, Elliptic Problems in Nonsmooth Domains, Society for Industrial and Applied Mathematics, 2011, https://epubs.siam.org/doi/abs/10.1137/1.9781611972030.
[34] J.-L. Guermond, B. Popov, L. Saavedra, Second-order invariant domain preserving ALE approximation of hyperbolic systems, J. Comput. Phys. (ISSN 0021-9991) 401 (2020) 108927, https://doi.org/10.1016/j.jcp.2019.108927, https://www.sciencedirect.com/science/article/pii/S0021999119306321.
[35] P. He, H. Tang, An adaptive moving mesh method for two-dimensional relativistic hydrodynamics, Commun. Comput. Phys. 11 (1) (2012) 114–146.
[36] P. He, H. Tang, An adaptive moving mesh method for two-dimensional relativistic magnetohydrodynamics, Comput. Fluids 60 (2012) 1–20.
[37] T. Hell, A. Ostermann, Compatibility conditions for Dirichlet and Neumann problems of Poisson's equation on a rectangle, J. Math. Anal. Appl. 420 (2014) 1005–1023, https://doi.org/10.1016/j.jmaa.2014.06.034.
[38] R.G. Hindman, Generalized coordinate forms of governing fluid equations and associated geometrically induced errors, AIAA J. 20 (10) (1982) 1359–1367, https://doi.org/10.2514/3.51196.
[39] W. Huang, Metric tensors for anisotropic mesh generation, J. Comput. Phys. (ISSN 0021-9991) 204 (2) (2005) 633–665, https://doi.org/10.1016/j.jcp.2004.10.024, https://www.sciencedirect.com/science/article/pii/S0021999104004310.
[40] W. Huang, R.D. Russell, Moving mesh strategy based on a gradient flow equation for two-dimensional problems, SIAM J. Sci. Comput. 20 (3) (1998) 998–1015, https://doi.org/10.1137/S1064827596315242.
[41] W. Huang, R.D. Russell, Adaptive Moving Mesh Methods, vol. 174, Springer Science & Business Media, 2010.
[42] W. Huang, W. Sun, Variational mesh adaptation II: Error estimates and monitor functions, J. Comput. Phys. (ISSN 0021-9991) 184 (2) (2003) 619–648, https://doi.org/10.1016/S0021-9991(02)00040-2, https://www.sciencedirect.com/science/article/pii/S0021999102000402.
[43] G.-S. Jiang, C.-W. Shu, Efficient implementation of weighted ENO schemes, J. Comput. Phys. (ISSN 0021-9991) 126 (1) (1996) 202–228, https://doi.org/10.1006/jcph.1996.0130.
[44] Y. Jiang, C.-W. Shu, M. Zhang, An alternative formulation of finite difference weighted ENO schemes with Lax–Wendroff time discretization for conservation laws, SIAM J. Sci. Comput. 35 (2) (2013) A1137–A1160, https://doi.org/10.1137/120889885.
[45] Y. Jiang, C.-W. Shu, M. Zhang, Free-stream preserving finite difference schemes on curvilinear meshes, Methods Appl. Anal. 21 (1) (2014) 1–30, https://doi.org/10.4310/MAA.2014.v21.n1.a1.
[46] P. Knupp, L.G. Margolin, M. Shashkov, Reference Jacobian optimization-based rezone strategies for arbitrary Lagrangian Eulerian methods, J. Comput. Phys. 176 (1) (Feb. 2002) 93–128, https://doi.org/10.1006/jcph.2001.6969.
[47] R. Li, T. Tang, P. Zhang, Moving mesh methods in multiple dimensions based on harmonic maps, J. Comput. Phys. (ISSN 0021-9991) 170 (2) (2001) 562–588, https://doi.org/10.1006/jcph.2001.6749, https://www.sciencedirect.com/science/article/pii/S002199910196749X.
[48] S. Li, L. Petzold, Moving mesh methods with upwinding schemes for time-dependent pdes, J. Comput. Phys. (ISSN 0021-9991) 131 (2) (1997) 368–377, https://doi.org/10.1006/jcph.1996.5611, https://www.sciencedirect.com/science/article/pii/S0021999196956119.
[49] S. Li, J. Duan, H. Tang, High-order accurate entropy stable adaptive moving mesh finite difference schemes for (multi-component) compressible Euler equations with the stiffened equation of state, https://arxiv.org/abs/2202.07989, 2022.
[50] G. Liao, D. Anderson, A new approach to grid generation, Appl. Anal. 44 (3–4) (1992) 285–298, https://doi.org/10.1080/00036819208840084.
[51] G. Liao, F. Liu, G.C. de la Pena, D. Peng, S. Osher, Level-set-based deformation methods for adaptive grids, J. Comput. Phys. (ISSN 0021-9991) 159 (1) (2000) 103–122, https://doi.org/10.1006/jcph.2000.6432, https://www.sciencedirect.com/science/article/pii/S0021999100964325.
[52] K. Lipnikov, M. Shashkov, A framework for developing a mimetic tensor artificial viscosity for Lagrangian hydrocodes on arbitrary polygonal meshes, J. Comput. Phys. (ISSN 0021-9991) 229 (20) (2010) 7911–7941, https://doi.org/10.1016/j.jcp.2010.06.045, https://www.sciencedirect.com/science/article/pii/S0021999110003694.
[53] R. Liska, B. Wendroff, Comparison of several difference schemes on 1D and 2D test problems for the Euler equations, SIAM J. Sci. Comput. (ISSN 1064-8275) 25 (3) (2003) 995–1017, https://doi.org/10.1137/S1064827502402120.
[54] F. Liu, S. Ji, G. Liao, An adaptive grid method and its application to steady Euler flow calculations, SIAM J. Sci. Comput. 20 (3) (1998) 811–825, https://doi.org/10.1137/S1064827596305738.
[55] Z. Liu, Y. Jiang, M. Zhang, Q. Liu, High order finite difference WENO methods for shallow water equations on curvilinear meshes, Commun. Appl. Math. Comput. (2022) 1–44, https://doi.org/10.1007/s42967-021-00183-w.
[56] D. Long, J. Thuburn, Numerical wave propagation on non-uniform one-dimensional staggered grids, J. Comput. Phys. (ISSN 0021-9991) 230 (7) (apr 2011) 2643–2659, https://doi.org/10.1016/j.jcp.2010.12.040.
[57] R. Loubère, P.-H. Maire, M. Shashkov, J. Breil, S. Galera, Reale: a reconnection-based arbitrary-Lagrangian-Eulerian method, J. Comput. Phys. (ISSN 0021-9991) 229 (12) (2010) 4724–4761, https://doi.org/10.1016/j.jcp.2010.03.011, https://www.sciencedirect.com/science/article/pii/S002199911000121X.
[58] D. Luo, W. Huang, J. Qiu, A quasi-lagrangian moving mesh discontinuous Galerkin method for hyperbolic conservation laws, J. Comput. Phys. (ISSN 0021-9991) 396 (2019) 544–578, https://doi.org/10.1016/j.jcp.2019.06.061, https://www.sciencedirect.com/science/article/pii/S0021999119304693.
[59] A.T.T. McRae, C.J. Cotter, C.J. Budd, Optimal-transport–based mesh adaptivity on the plane and sphere using finite elements, SIAM J. Sci. Comput. 40 (2) (2018) A1121–A1148, https://doi.org/10.1137/16M1109515.
[60] T. Nonomura, N. Iizuka, K. Fujii, Freestream and vortex preservation properties of high-order WENO and WCNS on curvilinear grids, Comput. Fluids 39 (2) (2010) 197–214, https://doi.org/10.1016/j.compfluid.2009.08.005.
[61] T. Nonomura, D. Terakado, Y. Abe, K. Fujii, A new technique for freestream preservation of finite-difference weno on curvilinear grid, Comput. Fluids (ISSN 0045-7930) 107 (2015) 242–255, https://doi.org/10.1016/j.compfluid.2014.09.025, https://www.sciencedirect.com/science/article/pii/S0045793014003624.
[62] H.S. Pathak, R.K. Shukla, Adaptive finite-volume weno schemes on dynamically redistributed grids for compressible Euler equations, J. Comput. Phys. 319 (2016) 200–230.
[63] R. Ramani, S. Shkoller, A multiscale model for Rayleigh-Taylor and Richtmyer-Meshkov instabilities, J. Comput. Phys. 405 (2020) 109177, https://doi.org/10.1016/j.jcp.2019.109177.
[64] R. Ramani, J. Reisner, S. Shkoller, A space-time smooth artificial viscosity method with wavelet noise indicator and shock collision scheme, Part 1: The 1-*D* case, J. Comput. Phys. (ISSN 0021-9991) 387 (2019) 81–116, https://doi.org/10.1016/j.jcp.2019.02.049, https://www.sciencedirect.com/science/article/pii/S0021999119301664.

[65] R. Ramani, J. Reisner, S. Shkoller, A space-time smooth artificial viscosity method with wavelet noise indicator and shock collision scheme, Part 2: The 2-*D* case, J. Comput. Phys. (ISSN 0021-9991) 387 (2019) 45–80, https://doi.org/10.1016/j.jcp.2019.02.048.
[66] F. Rathgeber, D.A. Ham, L. Mitchell, M. Lange, F. Luporini, A.T.T. Mcrae, G.-T. Bercea, G.R. Markall, P.H.J. Kelly, Firedrake: automating the finite element method by composing abstractions, ACM Trans. Math. Softw. (ISSN 0098-3500) 43 (3) (dec 2016), https://doi.org/10.1145/2998441.
[67] B. Semper, G. Liao, A moving grid finite-element method using grid deformation, Numer. Methods Partial Differ. Equ. 11 (1995) 603–615, https://doi.org/10.1002/num.1690110606.
[68] C.-W. Shu, Essentially non-oscillatory and weighted essentially non-oscillatory schemes for hyperbolic conservation laws, in: LNM, vol. 1697, Springer, Berlin, Heidelberg, ISBN 978-3-540-49804-9, 1998, pp. 325–432.
[69] C.-W. Shu, S. Osher, Efficient implementation of essentially non-oscillatory shock-capturing schemes, J. Comput. Phys. 77 (2) (Aug. 1988) 439–471, https://doi.org/10.1016/0021-9991(88)90177-5.
[70] J.M. Stone, T.A. Gardiner, P. Teuben, J.F. Hawley, J.B. Simon, Athena: a new code for astrophysical MHD, Astrophys. J. Suppl. Ser. 178 (1) (sep 2008) 137–177, https://doi.org/10.1086/588755.
[71] M. Sulman, J. Williams, R. Russell, Optimal mass transport for higher dimensional adaptive grid generation, J. Comput. Phys. (ISSN 0021-9991) 230 (9) (2011) 3302–3330, https://doi.org/10.1016/j.jcp.2011.01.025, https://www.sciencedirect.com/science/article/pii/S0021999111000507.
[72] M.M. Sulman, J. Williams, R.D. Russell, An efficient approach for the numerical solution of the Monge-Ampère equation, Appl. Numer. Math. (ISSN 0168-9274) 61 (3) (2011) 298–307, https://doi.org/10.1016/j.apnum.2010.10.006, https://www.sciencedirect.com/science/article/pii/S0168927410001819.
[73] H. Tang, T. Tang, Adaptive mesh methods for one- and two-dimensional hyperbolic conservation laws, SIAM J. Numer. Anal. 41 (2) (2003) 487–515, https://doi.org/10.1137/S003614290138437X.
[74] T. Tang, Moving mesh methods for computational fluid dynamics, in: Recent Advances in Adaptive Computation, in: Contemp. Math., vol. 383, Amer. Math. Soc., Providence, RI, 2005, pp. 141–173.
[75] P.D. Thomas, C.K. Lombard, Geometric conservation law and its application to flow computations on moving grids, AIAA J. 17 (10) (Oct. 1979) 1030–1037, https://doi.org/10.2514/3.61273.
[76] F.X. Timmes, G. Gisler, G.M. Hrbek, Automated analyses of the tri-lab verification test suite on uniform and adaptive grids for code project a, 2005.
[77] A. van Dam, P.A. Zegeling, et al., Balanced monitoring of flow phenomena in moving mesh methods, Commun. Comput. Phys. 7 (1) (2010) 138, https://doi.org/10.4208/cicp.2009.09.033.
[78] R. Vichnevetsky, L. Turner, Spurious scattering from discontinuously stretching grids in computational fluid dynamics, Appl. Numer. Math. (ISSN 0168-9274) 8 (3) (1991) 289–299, https://doi.org/10.1016/0168-9274(91)90058-8, https://www.sciencedirect.com/science/article/pii/0168927491900588.
[79] M.R. Visbal, D.V. Gaitonde, On the use of higher-order finite-difference schemes on curvilinear and deforming meshes, J. Comput. Phys. 181 (1) (2002) 155–185, https://doi.org/10.1006/jcph.2002.7117.
[80] R. Wang, H. Feng, R.J. Spiteri, Observations on the fifth-order weno method with non-uniform meshes, Appl. Math. Comput. (ISSN 0096-3003) 196 (1) (2008) 433–447, https://doi.org/10.1016/j.amc.2007.06.024, https://www.sciencedirect.com/science/article/pii/S0096300307006972.
[81] H. Weller, P. Browne, C. Budd, M. Cullen, Mesh adaptation on the sphere using optimal transport and the numerical solution of a Monge-Ampère type equation, J. Comput. Phys. (ISSN 0021-9991) 308 (2016) 102–123, https://doi.org/10.1016/j.jcp.2015.12.018, https://www.sciencedirect.com/science/article/pii/S0021999115008372.
[82] A.M. Winslow, Numerical solution of the quasilinear Poisson equation in a nonuniform triangle mesh, J. Comput. Phys. (ISSN 0021-9991) 1 (2) (1966) 149–172, https://doi.org/10.1016/0021-9991(66)90001-5, https://www.sciencedirect.com/science/article/pii/0021999166900015.
[83] A.M. Winslow, Adaptive-mesh zoning by the equipotential method, UCID-19062, Lawrence Livermore National Laboratory, 4 1981, https://www.osti.gov/biblio/6227449.
[84] X. Yang, W. Huang, J. Qiu, A moving mesh weno method for one-dimensional conservation laws, SIAM J. Sci. Comput. 34 (A2317–A2343) (01 2012), https://doi.org/10.1137/110856381.
[85] P.A. Zegeling, W.D. de Boer, H.Z. Tang, Robust and efficient adaptive moving mesh solution of the 2-D Euler equations, in: Recent Advances in Adaptive Computation, in: Contemp. Math., vol. 383, Amer. Math. Soc., Providence, RI, 2005, pp. 375–386.