WENO–WOMBAT: Scalable Fifth-order Constrained-transport Magnetohydrodynamics for Astrophysical Applications

J. M. F. Donnert[1,2,3], H. Jang[4], P. Mendygral[5], G. Brunetti[2], D. Ryu[4], and T. W. Jones[3,6]

[1] Leiden Observatory, P.O. Box 9513, NL-2300 RA Leiden, The Netherlands
[2] INAF Istituto di Radioastronomia, via P. Gobetti 101, I-40129 Bologna, Italy; brunetti@ira.inaf.it
[3] School of Physics and Astronomy, University of Minnesota, Minneapolis, MN 55455, USA; jdonnert@strw.leidenuniv.nl
[4] Department of Physics, School of Natural Sciences, Ulsan National Institute of Science and Technology, UNIST-gil 50, Ulsan, 44919, Republic of Korea
hanbyul@sirius.unist.ac.kr, ryu@sirius.unist.ac.kr
[5] Cray Inc., Bloomington, MN, USA; pjm@cray.com
[6] Minnesota Supercomputing Institute for Advanced Computational Research, USA; twj@umn.edu

Abstract

Owing to increases in computing power, high-order Eulerian schemes will likely become instrumental in simulations of turbulence and magnetic field amplification in astrophysical fluids in the next years. We present the implementation of a fifth-order weighted essentially non-oscillatory scheme for constrained-transport magnetohydrodynamics in the code WOMBAT. We establish the correctness of our implementation with extensive number tests. We find that the fifth-order scheme performs as accurately as a common second-order scheme at half the resolution. We argue that for a given solution quality, the new scheme is more computationally efficient than lower order schemes in three dimensions. We also establish the performance characteristics of the solver in the WOMBAT framework. Our implementation fully vectorizes using flattened arrays in thread-local memory. It performs at about 0.6 million zones per second per node on Intel Broadwell. We present scaling tests of the code on up to 98,000 cores on the Cray XC40 machine "Hazel Hen," with a sustained performance of about 5% of peak at scale.

Key words: magnetohydrodynamics (MHD) – methods: numerical

1. Introduction

Most of the baryonic matter in the universe is in the form of a thin ionized plasma. Hence, many numerical models of astrophysical interest require the solution of magnetohydro-dynamic (MHD) equations (e.g., Kulsrud & Ostriker 2006), from the solar corona, the intergalactic medium, to the jets of active galactic nuclei, the hot atmosphere of galaxy clusters, and the filaments of the cosmic web. Finite difference (FD) methods for hydrodynamics predate the earliest computers and of course transcend the field of astrophysics (Richardson 1922; LeVeque 2002). After seminal contributions by Lax (1954), Godunov (1959), Babuška & Zlámal (1973), van Leer (1979), and Roe (1981) and nearly a century of development, major progress has been made in ever more accurate and efficient finite volume (FV) and FD schemes to minimize spurious errors and capture shocks accurately (Harten 1983; Colella & Woodward 1984; Harten et al. 1987; Einfeldt 1988; Shu 1988; Liu et al. 1994). For MHD, the development of constrained-transport (CT) schemes marks the era of numerical magnetic fields with vanishing divergence to machine precision (Evans & Hawley 1988; Londrillo & Del Zanna 2000).

The plethora of available algorithms is only dwarfed by the vast amount of codes that use FV or FD methods today. In astrophysics, implementations would include AREPO (Springel 2010), ART (Kravtsov et al. 1997), ATHENA (Stone et al. 2008), CHOLLA (Schneider & Robertson 2015), DIS-PATCH (Nordlund et al. 2018), ENZO (Bryan et al. 2014), FLASH (Fryxell et al. 2000), GAMER (Schive et al. 2018), GIZMO (Hopkins & Raives 2016), Nyx (Almgren et al. 2013), RAMSES (Teyssier 2002), PLUTO (Mignone et al. 2007), ZEUS (Stone & Norman 1992), and many more.

With supercomputers now approaching the regime of 10^{18} floating point operations per second (flops), algorithms beyond the most commonly used second and third order for MHD are very feasible (see Balsara 2017 for a recent review). These methods offer increased accuracy over common codes, which is advantageous in the simulation of turbulence (Guillet et al. 2019) and when problems include supersonic advection of the fluid relative to the mesh.

However, improved accuracy comes at the expense of additional computational costs. It has been argued by Greenough & Rider (2004) that second-order codes are computationally more efficient in one dimension, in the sense of solution quality per computational cost. In this contribution, we will argue that this is not necessarily true in three dimensions for a fifth-order FD weighted essentially non-oscillatory (WENO) scheme (Shu 1998; Jiang & Shu 1996; Jiang & Wu 1999; Balsara & Shu 2000; Feng et al. 2004). WENO schemes use a weighted average of several stencils around a zone to achieve high accuracy and low truncation error in spatial interpolation, while avoiding spurious oscilla-tions near discontinuities (see Shu 2003 for an instructive exposition). Modern WENO schemes come in many flavors. Particularly relevant to this work are contributions by Henrick et al. (2005), who have shown that the classical scheme is only third-order accurate near critical points. Borges et al. (2008) have proposed a simple set of weights to restore full order (WENO-Z). Subsequent work has focused further improve-ments of the scheme and extended it to very high order; for reviews, see Shu (2009) and Balsara et al. (2016).

We will show that the classical FD method doubles the effective resolution of the grid compared to the popular piecewise parabolic method (PPM) or total variation

diminishing (TVD) schemes in all dimensions. This is in line with prior results that find WENO3, WENO5, and WENO9 double the effective resolution compared to the next lower order scheme (Shu 2003; Zhang et al. 2003); similar results have been obtained with other high-order schemes (Tóth et al. 2014). The excellent fidelity of these schemes allows in principle a problem to be simulated at half the resolution with WENO5 compared to PPM or WENO3, which more than makes up for the added computational cost by the fifth-order scheme. This is especially true for FD WENO schemes, which are an order of magnitude cheaper to compute than finite volume WENO schemes (Shu 2003) . WENO5 also reduces the accumulated round-off error over many time steps and reduces the data size by a factor of 8, which might be even more important than the reduction in computational cost.

However, due to the wide use of multicore computers, most fluid simulations today are not compute-bound. Thus, a WENO implementation will likely run at the same resolution as a lower order scheme to increase the fidelity of the simulation. Given the wide use of accelerators and the increasing diversity of CPU architectures, it is highly desirable to write a performance-aware implementation of an efficient scheme using open standards that can use many different architectures. The high compute intensity of the WENO FD scheme, i.e., the large byte per flop ratio in the main WENO loop, makes it very attractive for such an approach. As always in high performance computing (HPC), SIMD[7] vectorization is key to achieving good performance on CPUs and eventually GPUs by exposing instruction-level parallelism. The regular data layout of an Eulerian grid makes this significantly easier to achieve than in particle-based schemes.

Instead of presenting a fundamentally new WENO scheme, the aim of this work is to expose the efficiency of the FD WENO5 approach in terms of compute time per solution quality close to the petaflop scale on modern supercomputers. To this end, we present the implementation of a classical fifth-order FD WENO scheme (Jiang & Wu 1999) in the WOMBAT[8] code framework (Mendygral et al. 2017). We use CT to evolve the magnetic field with minimal divergence error with the "transport-flux" formulation by Ryu et al. (1998). Formally, spatial interpolation is fifth order, time interpolation is fourth order, and CT is second order. We deliberately chose a four-stage time integrator and a second-order CT scheme to keep the implementation computationally cheap. High-order finite volume schemes involving cross-dimensional averaging and their corresponding CT schemes are much more computationally expensive (e.g., Shu 2003; Londrillo & del Zanna 2004; Mignone et al. 2010; Verma et al. 2019). We will show that by sacrificing a little fidelity for computational speed, the FD approach leads to a very efficient and robust method. In particular, our simple second-order CT scheme affects only magnetic field dispersion; diffusion remains fifth-order accurate, at virtually no additional computational cost.

This work is structured as follows: we outline the scheme in Section 2. The implementation is heavily optimized to expose parallelism in the code at all levels; details are given in Section 3. Code tests with a special emphasis on errors relevant in cosmological simulations (advection error and angular momentum conservation) are presented in Section 4. Aside from WOMBAT's own TVD+CTU implementation, we frequently compare the code with published results from the order CTU+CT code ATHENA (Stone et al. 2008), as well as high-order finite volume codes. We test the code performance on modern HPC hardware in Section 5. Conclusions are drawn in Section 6. For reference purposes, we once again present the eigenvectors used in the calculation in the Appendix.

2. Numerical Method

The equations of ideal magnetohydrodynamics read (e.g., Ryu & Jones 1995):

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{v}) = 0, \tag{1}$$

$$\frac{\partial \boldsymbol{v}}{\partial t} + \boldsymbol{v} \cdot \nabla \boldsymbol{v} + \frac{1}{\rho}\nabla P - \frac{1}{\rho}(\nabla \times \boldsymbol{B}) \times \boldsymbol{B} = 0, \tag{2}$$

$$\frac{\partial P}{\partial t} + \boldsymbol{v} \cdot \nabla P + \gamma P \nabla \cdot \boldsymbol{v} = 0 \tag{3}$$

$$\frac{\partial \boldsymbol{B}}{\partial t} - \nabla \times (\boldsymbol{v} \times \boldsymbol{B}) = 0 \tag{4}$$

$$\nabla \cdot \boldsymbol{B} = 0 \tag{5}$$

where ρ is the density, $\boldsymbol{v} = (v_x, v_y, v_z)^{\mathrm{T}}$ is the velocity, and $\boldsymbol{B} = (B_x, B_y, B_z)^{\mathrm{T}}$ is the magnetic field. We have chosen a rationalized system of units, so factors of 4π do not appear. Further, we define $B = |\boldsymbol{B}|$, $v = |\boldsymbol{v}|$, the adiabatic index γ and the total pressure P^*, the total energy E, and the entropy S (Ryu et al. 1993) of the flow:

$$P^* = P + \frac{1}{2}B_x^2 + B_y^2 + B_z^2 \tag{6}$$

$$E = \frac{P}{\gamma - 1} + \frac{1}{2}(\rho v^2 + B^2), \tag{7}$$

$$S = \frac{P}{\rho^{\gamma - 1}}. \tag{8}$$

A state vector of conserved quantities $\boldsymbol{q}$ can be defined:

$$\boldsymbol{q} = (\rho, \rho v_x, \rho v_y, \rho v_z, B_x, B_y, B_z, E)^{\mathrm{T}} \tag{9}$$

and a vector of fluxes $\boldsymbol{F}$ in the x-direction:

$$\boldsymbol{F_x} = \begin{pmatrix} \rho v_x \\ \rho v_x^2 + P^* - B_x^2 \\ \rho v_x v_y - B_x B_y \\ \rho v_x v_z - B_x B_z \\ 0 \\ B_y v_x - B_x v_y \\ B_z v_x - B_x v_z \\ (E + P^*)v_x - B_x(B_x v_x + B_y v_y + B_z v_z) \end{pmatrix} \tag{10}$$

Then, Equations (1)–(4) can be written as a conservation law for[9] $\boldsymbol{q}$:

$$\frac{\partial \boldsymbol{q}}{\partial t} + \frac{\partial \boldsymbol{F_x}}{\partial x} = 0 \tag{11}$$

---

[7] Single instruction multiple data.
[8] wombatcode.org

---

[9] We do not consider source terms like gravity, cosmic rays, radiative cooling, etc.

This equation can be approximated in quasilinear form (Roe [1981], [1997]; Einfeldt et al. [1991]; van Leer [1979]):

$$\frac{\partial \boldsymbol{q}}{\partial t} + \mathsf{A}_x \frac{\partial \boldsymbol{q}}{\partial x} = 0 \qquad (12)$$

where we define the Jacobian matrix $\mathsf{A}_x = \partial \boldsymbol{F} / \partial \boldsymbol{q}$.

A complete set of left and right eigenvectors (see the Appendix), and real, albeit possibly degenerate, eigenvalues can be found for this Jacobian, thus the system (12) is hyperbolic (Brio & Wu [1988a]). The seven eigenvalues can be identified with the three MHD waves: the slow, fast, and Alfvèn modes, as well as an entropy mode. With the definition of the (hydrodynamic) sound speed $c_s = \sqrt{\gamma P / \rho}$, the MHD wave speeds are (Brio & Wu [1988a]; H. Jang et al. 2019, in preparation):

$$c_{\text{slow}} = \left(\frac{1}{2} c_s^2 + \frac{B^2}{\rho} - \sqrt{\left(\frac{B^2}{\rho} + c_s^2 \right)^2 - 4 \frac{B_x^2}{\rho} c_s^2} \right)^{1/2} \qquad (13)$$

$$c_{\text{fast}} = \left(\frac{1}{2} c_s^2 + \frac{B^2}{\rho} + \sqrt{\left(\frac{B^2}{\rho} + c_s^2 \right)^2 - 4 \frac{B_x^2}{\rho} c_s^2} \right)^{1/2} \qquad (14)$$

$$c_{\text{A}} = \frac{B_x}{\sqrt{\rho}}. \qquad (15)$$

Using the left- ($\boldsymbol{L}(m)$) and right-hand ($\boldsymbol{R}(m)$) eigenvectors of the linearized equations, the system can be decoupled into seven scalar advection equations by multiplying it with $\boldsymbol{L}$ from the left and using $\boldsymbol{L} \mathsf{A}_x \boldsymbol{R} = \Lambda$, where Λ is the diagonal matrix containing the eigenvalues $\lambda = (\lambda_{\text{fast}}^-, \lambda_{\text{A}}^-, \lambda_{\text{slow}}^-, \lambda_{\text{e}}, \lambda_{\text{slow}}^+, \lambda_{\text{A}}^+, \lambda_{\text{fast}}^+)$, where $\lambda^\pm = v_x \pm c_{\text{wave}}$ and c_{wave} is the corresponding wave speed. Thus, the linearized MHD equations encode the propagation of seven MHD waves in time along characteristics (e.g., LeVeque [2002]). The solution can then be obtained componentwise from the linearized scalar problem and be transformed back by multiplying it with $\boldsymbol{R}$.

In WOMBAT, we assume that space is discretized as $x_i = i \Delta x - L_{\text{Box}} / 2$, with $\Delta x = L_{\text{Box}} / N_x$, L_{Box} the size of the computational domain in the x-direction, and N_x the number of zones in the x-direction. One can enforce numerical conservation by writing Equation (12) with fluxes across the zone boundaries at $i \pm \frac{1}{2}$ in conservative form (Godunov [1959]; LeVeque [2002]):

$$\frac{d\boldsymbol{q}}{dt} + \frac{1}{\Delta x} (\boldsymbol{F}_{i+\frac{1}{2}}^n - \boldsymbol{F}_{i-\frac{1}{2}}^n) = 0. \qquad (16)$$

Given suitable initial conditions, the solution of Equation (11) then becomes an initial value problem and can be solved using the method of lines (e.g., Sarmin & Chudov [1967]).

Depending on the scheme, one chooses a suitable discretization in time to integrate forward and in space to interpolate the fluxes to the cell boundaries.

2.1. Time Discretization

Following Jiang & Shu ([1996]), we use a fourth-order, four-stage Runge–Kutta (RK4) time integrator. It has been shown that such a scheme cannot have the strong-stability-preserving property (e.g., Spiteri & Ruuth [2002]). In fact, the scheme used here is not even total variation diminishing (TVD; Shu [1988]; Shu & Osher [1988]). Nonetheless, it allows us to use

CFL = 0.8 everywhere, even in three dimensions, and is sufficiently robust in practice. Modern SSP RK schemes are better behaved mathematically, but have to involve five or more stages at fourth order and thus are computationally 25% more expensive than the fourth-order scheme at the same time-step size (e.g., Gottlieb et al. [2001]; Spiteri & Ruuth [2002]; Gottlieb et al. [2009]). As we aim for computational efficiency per solution quality, we leave tests of SSP integrators to future work.

In N_{d} dimensions, the time step is set as

$$\Delta t < \frac{\text{CFL} \Delta x}{\sum_{d=1}^{N_d} \max(|v_d| + c_{\text{fast}}^d))}. \qquad (17)$$

Omitting CT for now, the RK4 scheme from Jiang & Shu ([1996]) and Jiang & Wu ([1999]) can be written as

$$\boldsymbol{q} = \boldsymbol{q_0} = \boldsymbol{q}_{i,j,k}^n$$

$$\boldsymbol{q}_{\text{save}} = -\frac{4}{3} \boldsymbol{q_0}$$

$$\boldsymbol{q} = \boldsymbol{q_0} - \frac{1}{2} \frac{\Delta t}{\Delta x} (\boldsymbol{F}_{i+1/2,j,k}(\boldsymbol{q}) - \boldsymbol{F}_{i-1/2,j,k}(\boldsymbol{q}))$$

$$\boldsymbol{q}_{\text{save}} = \boldsymbol{q}_{\text{save}} + \frac{1}{3} \boldsymbol{q}$$

$$\boldsymbol{q} = \boldsymbol{q_0} - \frac{1}{2} \frac{\Delta t}{\Delta x} (\boldsymbol{F}_{i+1/2,j,k}(\boldsymbol{q}) - \boldsymbol{F}_{i-1/2,j,k}(\boldsymbol{q}))$$

$$\boldsymbol{q}_{\text{save}} = \boldsymbol{q}_{\text{save}} + \frac{2}{3} \boldsymbol{q}$$

$$\boldsymbol{q} = \boldsymbol{q_0} - \frac{\Delta t}{\Delta x} (\boldsymbol{F}_{i+1/2,j,k}(\boldsymbol{q}) - \boldsymbol{F}_{i-1/2,j,k}(\boldsymbol{q}))$$

$$\boldsymbol{q}_{\text{save}} = \boldsymbol{q}_{\text{save}} + \frac{1}{3} \boldsymbol{q}$$

$$\boldsymbol{q} = \boldsymbol{q_0} - \frac{1}{6} \frac{\Delta t}{\Delta x} (\boldsymbol{F}_{i+1/2,j,k}(\boldsymbol{q}) - \boldsymbol{F}_{i-1/2,j,k}(\boldsymbol{q}))$$

$$\boldsymbol{q}_{\text{save}} = \boldsymbol{q}_{\text{save}} + \boldsymbol{q}$$

$$\boldsymbol{q}_{i,j,k}^{n+1} = \boldsymbol{q}_{\text{save}}.$$

The CT time integration scheme is the same, but updates the face-centered magnetic fields after the WENO step (see Section 2.3). In three dimensions, it adds the three-component magnetic field on the boundary and the three-component corner flux to global storage.

2.2. WENO Spatial Discretization

For reference, we outline the WENO5 spatial discretization to obtain fluxes at the zone boundaries in Equation (16). The classical scheme from Jiang & Wu ([1999]) is an FD approach that efficiently computes boundary fluxes from point-valued cell-centered states and fluxes. This is in contrast to modern schemes that often use a more flexible, but also more computationally expensive, FV approach. For more details, we kindly ask the reader to refer to the extensive literature on the subject (e.g., Jiang & Shu [1996]; Shu [1998]; Jiang & Wu [1999]; Shu [2009]; H. Jang et al. 2019, in preparation).

The fifth-order WENO scheme uses a weighted average of three third-order stencils to approximate the solution at the boundary. Thus, the WENO averaging itself requires two boundary zones. Including an additional zone from the flux

difference in Equation (16) gives a total of three boundary zones per RK substep. To avoid Gibbs phenomena (ringing) near shocks, the weights "switch off" one of the three polynomials adjacent to discontinuities and inside a discontinuity, where the method becomes first order.

We note that in two or three dimensions, all fluxes are applied at the same time, starting from the same initial state vector. Strang splitting is not used (Shu 2003). For the sake of efficiency, the scheme does not include dimensional cross-terms found in many recent FV schemes (see, e.g., Balsara et al. 2009; Verma & Müller 2018). This strategy is common in the FD WENO literature that commonly aims at cheap high-order implementations (e.g., Shu 2003). The general argument is that the time-integration scheme implicitly averages over all three directions four times and thus maintains the high order as long as cross-dimensional fluxes are not too large. The rotated three-dimensional linear wave advection test (Section 4.1) confirms that the WENO5 scheme is effectively unsplit at fifth order in the linear regime. Cross-terms might be more relevant for strongly nonlinear problems, where the influence of the grid can be seen, e.g., in the Sedov–Taylor blast wave test, Section 4.16. As we will show, the algorithmic fidelity of the FD WENO5 scheme is then comparable to a third-order discontinuous Galerkin scheme, however still at a fraction of the computational cost.

Following Jiang & Wu (1999), we denote quantities in the decoupled system with superscript s, denote the individual components with $m \in [1, 7]$, and drop the vector notation. Thus, the fluxes, state vectors, and their differences in the decoupled system are

$$F_k^s(m) = \boldsymbol{L}_{i+\frac{1}{2}}(m) \cdot \boldsymbol{F}_k, \tag{18}$$

$$q_k^s(m) = \boldsymbol{L}_{i+\frac{1}{2}}(m) \cdot \boldsymbol{q}_k, \tag{19}$$

$$\Delta F_{k+\frac{1}{2}}^s(m) = F_{k+1}^s(m) - F_k^s(m), \tag{20}$$

$$\Delta q_{k+\frac{1}{2}}^s(m) = q_{k+1}^s(m) - q_k^s(m), \tag{21}$$

respectively. Note that the left-hand eigenvector is taken at the zone boundary using simple arithmetic averaging. In the decoupled system, WENO uses Lax–Friedrichs-type flux splitting to upwind the fluxes (Lax 1954). Jiang & Wu (1999) and Shu (2003) have argued that this usually very diffusive approach is sufficiently accurate and computationally cheap in the FD WENO context:

$$F_i^{s\pm}(m) = \frac{1}{2}(F_i^s(m) \pm \alpha(m)q_i^s(m)) \tag{22}$$

$$\Delta F_{k+\frac{1}{2}}^{s\pm}(m) = \frac{1}{2}(\Delta F_{k+\frac{1}{2}}^s(m) \pm \alpha(m)\Delta q_{k+\frac{1}{2}}^s(m)), \tag{23}$$

where $\alpha(m) = \max(|\lambda_i(m)|, |\lambda_{i+1}(m)|)$ is the larger of the two eigenvalues $\lambda(m)$ between zone i and $i + 1$.

The WENO-interpolated fluxes at the zone boundary for Equation (16) are given by $\boldsymbol{F}_{i+\frac{1}{2}} = F_{i+\frac{1}{2}}^s(m) \cdot \boldsymbol{R}_{i+\frac{1}{2}}(m)$, where again the right-hand eigenvector is taken at the zone boundary. In the decoupled system, the flux of each component m is

(Jiang & Wu 1999)

$$\begin{aligned} F_{i+\frac{1}{2}}^s = &\frac{1}{12}(-F_{i-1}^s + 7F_i^s + 7F_{i+1}^s - F_{i+2}^s) \\ &- \varphi_N(\Delta F_{i-\frac{3}{2}}^{s+}, \Delta F_{i-\frac{1}{2}}^{s+}, \Delta F_{i+\frac{1}{2}}^{s+}, \Delta F_{i+\frac{3}{2}}^{s+}) \\ &+ \varphi_N(\Delta F_{i+\frac{5}{2}}^{s-}, \Delta F_{i+\frac{3}{2}}^{s-}, \Delta F_{i+\frac{1}{2}}^{s-}, \Delta F_{i-\frac{1}{2}}^{s-}), \end{aligned} \tag{24}$$

where we have dropped the (m) notation for clarity. The WENO interpolant $\varphi_N(a, b, c, d)$ is defined as

$$\varphi_N = \frac{1}{3}\omega_0(a - 2b + c) + \frac{1}{6}\left(\omega_2 - \frac{1}{2}\right)(b - 2c + d). \tag{25}$$

The nonlinear weights are

$$\omega_0 = \frac{\alpha_0}{\alpha_0 + \alpha_1 + \alpha_2}, \quad \omega_2 = \frac{\alpha_2}{\alpha_0 + \alpha_1 + \alpha_2} \tag{26}$$

$$\alpha_0 = \frac{1}{(\epsilon + IC_0)^2}, \quad \alpha_1 = \frac{6}{(\epsilon + IC_1)^2}, \tag{27}$$

$$\alpha_2 = \frac{3}{(\epsilon + IC_2)^2}, \tag{28}$$

with $\epsilon = 10^{-6}$ and

$$IS_0 = 13(a - b)^2 + 3(a - 3b)^2, \tag{29}$$

$$IS_1 = 13(b - c)^2 + 3(b + c)^2, \tag{30}$$

$$IS_2 = 13(c - d)^2 + 3(3c - d)^2. \tag{31}$$

This description of the weights is generally fifth-order accurate in smooth flows, but only third-order accurate near critical points (extrema and saddle points). Borges et al. (2008) proposed the WENO-Z weights to make the scheme truly fifth order everywhere:

$$\tau_5 = |IS_0 - IS_2| \tag{32}$$

$$\alpha_0 = 1 + \frac{\tau_5}{(\epsilon_Z + IS_0)^2}, \tag{33}$$

$$\alpha_1 = 6\left(1 + \frac{\tau_5}{(\epsilon_Z + IS_1)^2}\right), \tag{34}$$

$$\alpha_2 = 3\left(1 + \frac{\tau_5}{(\epsilon_Z + IS_2)^2}\right), \tag{35}$$

and $\epsilon_Z = 10^{-40}$. As we will see, this is useful in advection-dominated problems, but reduces the general robustness of the scheme. For complex problems, WENO-Z will fall back into protection fluxes more often, which might not aid the solution for all problems.

2.3. Constrained-transport MHD

Equation (5) requires maintenance of a zero magnetic field divergence at all times. In CT, the magnetic field is defined on a staggered mesh, i.e., on the faces of the computational zones (Evans & Hawley 1988; Dai & Woodward 1998; Balsara & Spicer 1999b; Tóth 2000). This way, magnetic field divergence can be maintained to machine precision. For computational efficiency, we follow (Ryu et al. 1998) in our implementation of CT, which uses second-order accurate averages to obtain the corner fluxes. As we will see, this choice introduces additional dispersion to the solution, but not additional diffusion.

We denote the magnetic field on the forward zone faces (i.e., $i + \frac{1}{2}$ for the x-component) as $\boldsymbol{b} = (b_x, b_y, b_z)^T$. From the

Riemann solver, we obtain the fifth-order accurate magnetic field fluxes across the zone faces f_y, f_z during the sweep in the x-direction (components six and seven); g_z, g_x for the y-direction; and h_x, h_y for the z-direction:

$$f_y^* = f_y + \frac{1}{2}(B_{x,i,j,k}v_{y,i,j,k} + B_{x,i+1,j,k}v_{y,i+1,j,k}) \qquad (36)$$

$$f_z^* = f_z + \frac{1}{2}(B_{x,i,j,k}v_{z,i,j,k} + B_{x,i+1,j,k}v_{z,i+1,j,k}) \qquad (37)$$

$$g_x^* = g_x + \frac{1}{2}(B_{y,i,j,k}v_{z,i,j,k} + B_{y,i,j+1,k}v_{z,i,j+1,k}) \qquad (38)$$

$$g_z^* = g_z + \frac{1}{2}(B_{y,i,j,k}v_{x,i,j,k} + B_{y,i,j+1,k}v_{x,i,j+1,k}) \qquad (39)$$

$$h_x^* = h_x + \frac{1}{2}(B_{z,i,j,k}v_{x,i,j,k} + B_{z,i,j,k+1}v_{x,i,j,k+1}) \qquad (40)$$

$$h_y^* = h_y + \frac{1}{2}(B_{z,i,j,k}v_{y,i,j,k} + B_{z,i,j,k+1}v_{u,i,j,k+1}). \qquad (41)$$

Due to our implementation involving the rotation of the grid, the components actually remain the same for every direction. However, the fluxes have to be rotated backward (g_x, g_z) or forward (h_x, h_y) into the original frame. The corner fluxes Ω_x, Ω_y, Ω_z are then

$$\Omega_x = \frac{1}{2}(g_{x,i,j,k}^* + g_{x,i+1,j,k}^*) - \frac{1}{2}(f_{y,i,j,k}^* + f_{y,i,j+1,k}^*) \qquad (42)$$

$$\Omega_y = \frac{1}{2}(h_{y,i,j,k}^* + h_{y,i,j+1,k}^*) - \frac{1}{2}(g_{z,i,j,k}^* + g_{z,i,j,k+1}^*) \qquad (43)$$

$$\Omega_z = \frac{1}{2}(f_{z,i,j,k}^* + f_{z,i,j,k+1}^*) - \frac{1}{2}(h_{x,i,j,k}^* + h_{x,i+1,j,k}^*). \qquad (44)$$

The face-centered magnetic fields are then updated analogously to Equation (16):

$$\frac{db_x}{dt} + \frac{1}{\Delta x}(\Omega_{x,i,j,k} - \Omega_{x,i,j-1,k})$$
$$- \frac{1}{\Delta x}(\Omega_{z,i,j,k} - \Omega_{z,i,j,k-1}) = 0 \qquad (45)$$

$$\frac{db_y}{dt} + \frac{1}{\Delta x}(\Omega_{y,i,j,k} - \Omega_{y,i,j,k-1})$$
$$- \frac{1}{\Delta x}(\Omega_{x,i,j,k} - \Omega_{x,i-1,j,k}) = 0 \qquad (46)$$

$$\frac{db_z}{dt} + \frac{1}{\Delta x}(\Omega_{z,i,j,k} - \Omega_{z,i-1,j,k})$$
$$- \frac{1}{\Delta x}(\Omega_{y,i,j,k} - \Omega_{y,i,j-1,k}) = 0. \qquad (47)$$

The zone-centered magnetic field is interpolated at fourth order from face values:

$$B_{x,i,j,k} = \frac{1}{16}(-b_{x,i-2,j,k} + 9b_{x,i-1,j,k}$$
$$+ 9b_{x,i,j,k} - b_{x,i+1,j,k}), \qquad (48)$$

and the other components follow accordingly along the j and k indices.

We use the same fourth-order Runge–Kutta scheme presented in Section 2.1. The CT update is interleaved with the WENO update, with boundary communication of Ω

between a WENO and a CT step. As we will see, the CT scheme converges to second order. However, it can be shown that magnetic field dissipation converges to fifth order, so the additional error is in the shape of the field, not its energy (see also H. Jang et al. 2019, in preparation).

3. Implementation

We implement the scheme in the numerical code WOMBAT (see footnote 8). For a detailed description of the code infrastructure and its performance, readers may consider Mendygral et al. (2017). The code is written in object-oriented Fortran 2008 and hybrid-parallelized with MPI and OpenMP. It decomposes the computational domain into patches, blocks of independent work of variable size that can be cache-blocked (16–32 zones per dimension depending on architecture). This way, communication is reduced to a boundary problem, i.e., the boundary (ghost) zones between patches need to be communicated (resolved). Patches are implemented as Fortran objects and store boundary zones alongside all necessary information about neighboring patches and MPI ranks. Thus, there is no global data structure keeping track of patch distribution that may grow with an increasing number of MPI ranks, and communication and computation are intrinsically separated, i.e., the program is highly modular.

Initially, each MPI rank carries a cubic portion of the world grid (in 3D), a domain, containing a large number of patches. Patches are load-balanced via tunable virtual domains, where neighboring MPI ranks are included into a ranks domain. Patches are offloaded to other MPI ranks by exporting them into the virtual domain region and are then communicated analogous to the boundary exchange.

OpenMP threads are implemented using a single parallel region and combined with MPI_THREAD_MULTIPLE, so that threads independently and asynchronously carry out MPI communication of boundaries and resolution of patches (computational work). WOMBAT currently uses one-sided MPI-RMA with calls to MPI_Put to place each of the 27 patch boundary regions in a neighbor's mailbox of user-definable size. A signal per mailbox of 8 bytes (the "heartbeat") is used to notify the neighboring MPI rank of completed communication. The signal is updated at every loop iteration even if no data was communicated to achieve a form of weak synchronization among ranks. Once every boundary is communicated (i.e., all the signals are set as completed), a thread on the other rank unpacks all boundaries from the mailbox into the corresponding patch object. The patch is then "resolved" by any OpenMP thread on the rank. This way, communication can react to load-imbalance and also network contention on large multiuser machines. This represents fine-grained communication–computation overlap on the thread level. This implementation shifts the communication pattern from few large MPI messages requiring large buffers to many small messages with accordingly smaller buffers. Thus, this approach requires the MPI library to support lock-free OpenMP and lowest overhead, which is currently the case for Cray's MPI library and also OpenMPI, where WOMBAT is part of the regression testing. The communication scheme is actively researched for exascale systems by performance engineers at Cray Inc. and thus subject to continuous improvements.

3.1. The WENO Solver

In WOMBAT, every solver works on a single boundary communicated patch, where the grid data reside in rank global memory. At this point, no communication is required any longer; work and communication are completely separate in the implementation. This eases maintenance of the code considerably.

In the WENO solver, we first copy the grid data into OpenMP thread private buffers to minimize Non-Uniform Memory Access effects. This introduces memory overhead of about 25 MB per thread for patches with 18^3 zones. In multiple dimensions, the grid is explicitly flattened as well, i.e., the number of indices of all arrays is reduced to one along spatial dimensions and all loops run over the flattened array, ignoring the boundaries between dimensions. Thus, memory and code-level layout of the data are identical. This increases vector length and decreases cache-blocked patch size, which in turn eases MPI load balancing. It also improves code readability, as all lower level routines (fluxes, eigenvectors) remain inherently one-dimensional. This layout naturally leads to an explicit separation of computation and data movement in the code. In two and three dimensions, the sweeps in the y-direction and z-direction are implemented as rotations of the plane/cube in flattened memory. The resulting WENO fluxes are later rotated back into the original coordinate system to compute the state-vector updates (Equation (16) and Section 2.1). Using the Fortran CONTIGUOUS keyword and implied array sizes, all relevant loops in the implementation auto-vectorize with current Cray and Intel compilers. This is a necessary prerequisite to achieve a significant fraction of peak performance on current CPUs. We anticipate that this implementation will be very convenient to port to accelerators (GPUs) as it naturally exposes long vectors and makes memory movement explicit.

Due to WOMBAT's parallelization strategy, the RK4 scheme presented in Section 2.1 is implemented using a running state vector q, a state vector buffer q_{save} to accumulate intermediate results, and a copy of the initial state $q_0 = q_i^n$. Note that we communicate boundaries twice (once for WENO, once for CT update) per RK4 substep. This allows us to retain three boundary zones on the patch, which is optimal regarding flops and memory consumption given that communication in WOMBAT is local, asynchronous, and thus extremely cheap.

3.1.1. WENO–WOMBAT in One Dimension

The program proceeds as follows:

1. Copy-in q. If executing the first substep, initialize q_0 and set q_{save}.
2. Compute cell-centered pressure P (Equation (7)), eigenvalues c_A, c_{fast}, c_{slow} (Equations (13)–(15)), and fluxes F (Equation (10)).
3. Compute eigenvectors L and R on the cell boundaries, compute WENO fluxes, Equation (24).
4. Update q using WENO fluxes, update q_{save} using new q.
5. Move q, q_0, q_{save} on the patch, communicate patch boundaries.

Repeat four times with corresponding Runge–Kutta factors so that $q^{n+1} = q_{\mathrm{save}}$ at the last substep.

3.1.2. WENO–WOMBAT in Two Dimensions

In two dimensions, we have to interleave the CT step into the WENO5 RK4 update. The program proceeds as follows:

1. Copy-in and flatten q. If executing first substep, initialize q_0 and set q_{save}, else copy-in and flatten them. Set a buffer $q_{\mathrm{buf}} = q_0$.
2. Compute from q the cell-centered pressure P (Equation (7)), eigenvalues c_A, c_{fast}, c_{slow} (Equations (13)–(15)), and fluxes F (Equation (10)).
3. Compute from q the eigenvectors L and R on the cell boundaries, compute WENO fluxes (Equation (24)), and compute 2D CT fluxes (Equations (36) and (37)).
4. Update q_{buf} using WENO fluxes.
5. Rotate q forward so the flattened index is along the y-direction.
6. Compute from q the cell-centered pressure P (Equation (7)), eigenvalues c_A, c_{fast}, c_{slow} (Equations (13)–(15)), and fluxes (Equation (10)).
7. Compute from q the eigenvectors L and R on the cell boundaries, compute WENO fluxes (Equation (24)), and compute 2D CT fluxes (Equations (38) and (39)).
8. Rotate q and CT fluxes backwards.
9. Update q_{buf} using WENO fluxes, copy q_{buf} into q, and compute CT corner flux Ω_x (Equation (42)).
10. Update q_{save} using new q, but not the magnetic field.
11. Move q, q_0, q_{save}, Ω_x on the patch; communicate patch boundaries, face-centered magnetic fields, and corner flux.
12. Copy-in and flatten q, b, Ω_x.
13. Update face-centered magnetic field b from Ω_x.
14. Interpolate face-centered magnetic field b to zone-centered magnetic field B at fourth order. Update q_{save} with new magnetic field.
15. Move q, b, q_{save} to the patch object.

Repeat four times with corresponding Runge–Kutta factors so that $q^{n+1} = q_{\mathrm{save}}$ at the last substep. Perform another CT update on q_{save} before moving it into q.

3.1.3. WENO–WOMBAT in Three Dimensions

In three dimensions, the program proceeds as follows:

1. Copy-in and flatten q. If executing first substep, initialize q_0 and set q_{save}, else copy-in and flatten them. Set a buffer $q_{\mathrm{buf}} = q_0$.
2. Compute from q the cell-centered pressure P (Equation (7)), eigenvalues c_A, c_{fast}, c_{slow} (Equations (13)–(15)), and fluxes F (Equation (10)).
3. Compute from q the eigenvectors L and R on the cell boundaries, compute WENO fluxes (Equation (24)), and compute 3D CT fluxes (Equations (36) and (37)).
4. Update q_{buf} using WENO fluxes
5. Rotate q and q_{buf} forward so the flattened index is along the y-direction.
6. Compute from q the cell-centered pressure P (Equation (7)), eigenvalues c_A, c_{fast}, c_{slow} (Equations (13)–(15)), and fluxes (Equation (10)).
7. Compute from q the eigenvectors L and R on the cell boundaries, compute WENO fluxes (Equation (24)), and compute 3D CT fluxes (Equations (38) and (39)).
8. Update q_{buf} using WENO fluxes.

9. Rotate q and $q_{\rm buf}$ forward so that the flattened index is along the z-direction. Rotate CT fluxes backwards.
10. Compute from q the cell-centered pressure P (Equation (7)), eigenvalues c_A, $c_{\rm fast}$, $c_{\rm slow}$ (Equations (13)–(15)), and fluxes (Equation (10)).
11. Compute from q the eigenvectors L and R on the cell boundaries, compute WENO fluxes (Equation (24)), and compute 3D CT fluxes using (Equations (40) and (41)).
12. Update $q_{\rm buf}$ using WENO fluxes.
13. Rotate forward q and $q_{\rm buf}$ and CT fluxes, so that the first index runs along the x-direction again.
14. Update $q_{\rm buf}$ using WENO fluxes, copy $q_{\rm buf}$ into q, compute CT corner flux (Equations (42)–(44)) from rotated CT fluxes.
15. Update $q_{\rm save}$ using new q, but not the magnetic field.
16. Move q, q_0, $q_{\rm save}$, Ω on the patch, communicate patch boundaries, face-centered magnetic fields b, and corner flux.
17. Copy-in and flatten q, b, Ω.
18. Update face-centered magnetic field b from Ω.
19. Interpolate face-centered magnetic field b to zone-centered magnetic field B at fourth order. Update $q_{\rm save}$ with the new magnetic field.
20. Move q, b, $q_{\rm save}$ to the patch object.

Repeat four times with corresponding Runge–Kutta factors so that $q^{n+1} = q_{\rm save}$ at the last substep. Perform another CT update on $q_{\rm save}$ before moving it into q.

4. Test Calculations

All simulations in this section are run with CFL $= 0.8$ in 8 byte precision. We also do not use protection fluxes or density/pressure floors, unless noted otherwise. State vectors are defined as primitive variables $U = (\rho, v_x, v_y, v_z, B_x, B_y, B_z, P)^T$ or as conserved variables $Q = (\rho, \rho v_x, \rho v_y, \rho v_z, B_x, B_y, B_z, E)^T$, where $E(U)$ can be obtained from Equation (7). We compare our WENO5 or WENO5-Z results with WOMBAT's second-order TVD+CTU implementation presented in Mendygral et al. (2017). The L_1 error norm is given by

$$L_1(q) = \sqrt{\sum_s \left(\sum_{i=1}^{N_{\rm Zones}} \frac{|q_i(t) - q_i(0)|}{N} \right)^2}, \qquad (49)$$

where s denotes the components of the state vector. We are using the geometrical norm of the L_1 error to keep the results comparable to those of Gardiner & Stone (2005, 2008), Stone et al. (2008), and Felker & Stone (2018). Other definitions are in use as well, e.g., the mean of L_1.

4.1. Linear Wave Convergence

The convergence order of the scheme can be exposed by advecting a perturbation corresponding to one of the (M)HD eigenvectors through a periodic box with $2N \times N \times N$ zones and a domain of $3 \times 1.5 \times 1.5$ in three dimensions. This test evaluates the code in the linear regime and is very valuable for debugging. As the setup is not trivial, especially in three dimensions, we describe it in greater detail here, but note that the WOMBAT or ATHENA source code is likely indispensable to fully reproduce the results. A very instructive discussion of advection comparing WENO5 with TVD in one dimension can be found in the introduction of Shu (2003).
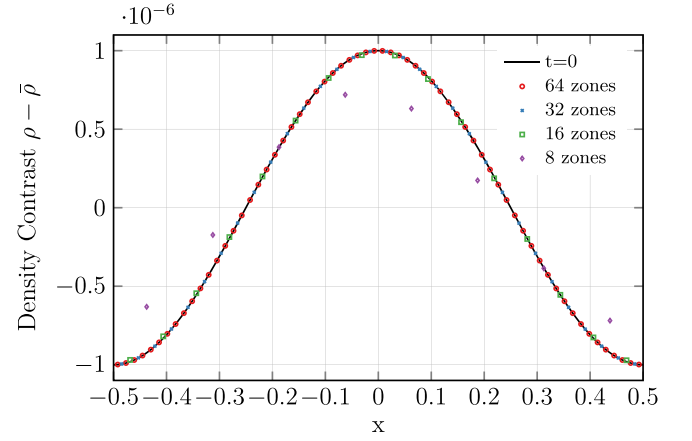

Figure 1. Density contrast in the one-dimensional sound wave advection test. Initial conditions with 1024 zones are shown by the black line. Simulation at $t = 10$ with 8, 16, 32, and 64 zones shown as violet diamonds, green square, blue crosses, and red circles, respectively.

Following Gardiner & Stone (2005, 2008) and Stone et al. (2008), we set $\gamma = 5/3$ and $\bar{U} = (1, v_x, 0, 0, B_x, B_y, B_z, 1/\gamma)^T$, where $v_x = 1$ for shear and entropy modes and $v_x = 0$ otherwise. For hydrodynamic waves, $B = 0$; for MHD waves, $B = (1, \sqrt{2}, 1/2)^T$. We add a perturbation on the conserved variables: $Q = \bar{Q} + A_0 R_k \sin(2\pi x)$. Here, R_k is the right-hand eigenvector of mode k and $A_0 = 10^{-6}$. We note that the perturbation is applied only once per component, i.e., the momentum fluctuation uses the unperturbed background density. The right eigenvectors for the (M)HD waves are

$$R_{\rm Alfven} = \frac{1}{6\sqrt{5}}(0, \ 0, \ 1, \ -2\sqrt{2}, \ 0, \ -1, \ 2\sqrt{2}, \ 0)^T \qquad (50)$$

$$R_{\rm fast} = \frac{1}{6\sqrt{5}}(6, 12, \ -4\sqrt{2}, \ -2, 0, \ 8\sqrt{2}, \ 4, 27)^T \qquad (51)$$

$$R_{\rm slow} = \frac{1}{6\sqrt{5}}(12, 6, 8\sqrt{2}, \ 4, 0, \ -4\sqrt{2}, \ -2, 9)^T \qquad (52)$$

$$R_{\rm entropy} = \left(1, 1, \ 0, 0, \ 0, 0, \ 0, \ \frac{1}{2}\right)^T. \qquad (53)$$

$$R_{\rm sound} = (1, 1, \ 0, 0, \ 0, 0, \ 0, 1/(\gamma - 1))^T \qquad (54)$$

$$R_{\rm shear,y} = (0, 0, \ 1, \ 0, \ 0, 0, \ 0, 0)^T. \qquad (55)$$

In three dimensions, we rotate the zone positions parallel to the wave vector following Gardiner & Stone (2008), with the rotation matrix

$$M(\alpha, \beta) = \begin{bmatrix} \cos\alpha\cos\beta & -\sin\beta & \sin\alpha\cos\beta \\ \cos\alpha\sin\beta & \cos\beta & -\sin\alpha\sin\beta \\ \sin\alpha & 0 & \cos\alpha \end{bmatrix}, \qquad (56)$$

where $\sin\alpha = 2/3$ and $\cos\beta = 2/\sqrt{5}$. We use its inverse/transpose to rotate back into the laboratory frame. We set a magnetic vector potential:

$$A_x = 0 \qquad (57)$$

$$A_y = \frac{A_0}{2\pi}R_{k,7}\sin(2\pi x) + B_z x_k \qquad (58)$$

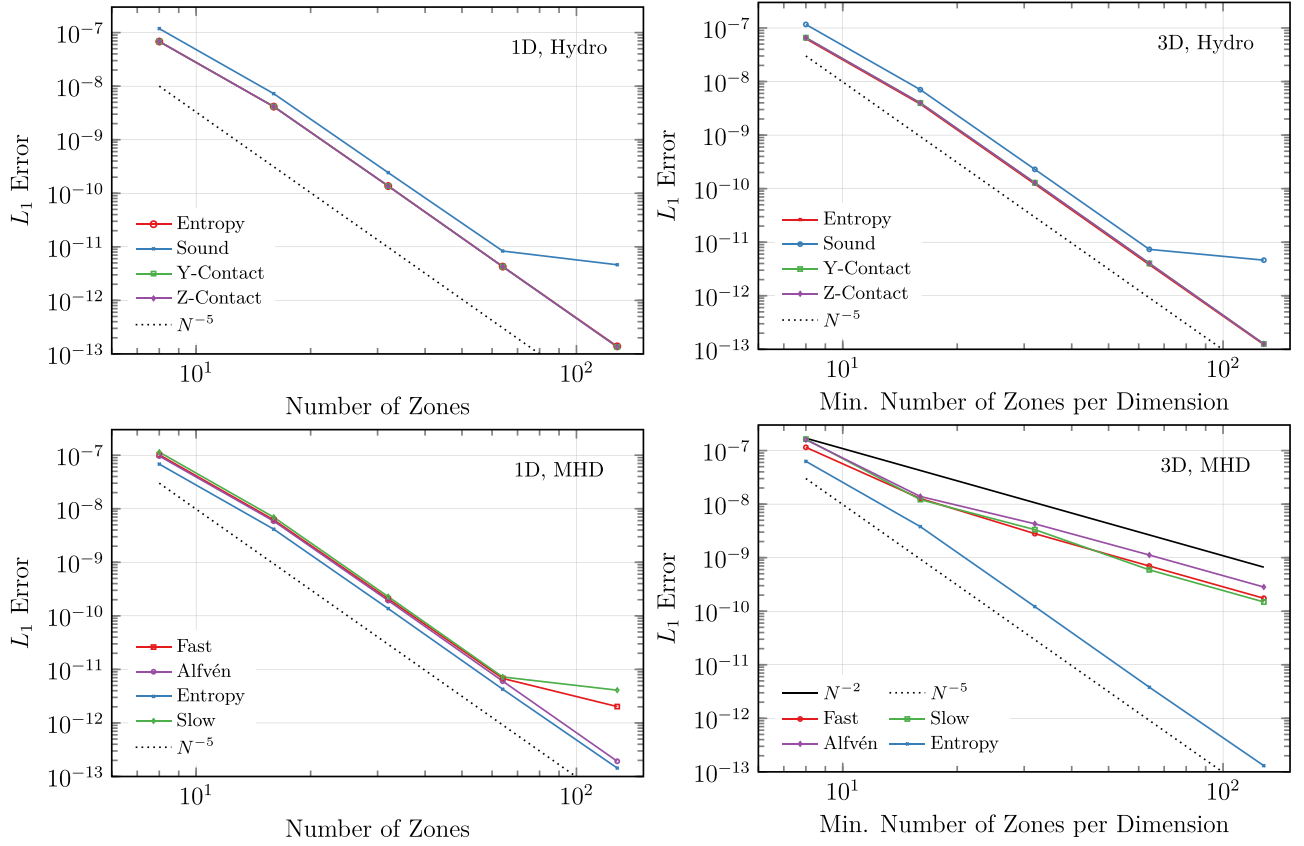$$A_z = -\frac{A_0}{2\pi}R_{k,6}\sin(2\pi x) - B_y x_k + B_x y_k, \qquad (59)$$

7

Figure 2. L_1 error over resolution for advecting hydrodynamic (top) and MHD waves (bottom) in one dimension (left) and in three dimensions (right) with WENO5-Z across a unit computational domain for one wavelength following (Gardiner & Stone 2008). Fifth order is marked as dotted black line, second order as black line.

where x_k and y_k are the x and y components of the zone position rotated with the inverse rotation matrix. We note that the vector potential has to be evaluated on the edges of the zone in three dimensions, so that the resulting magnetic field is field-centered. An example in our case Ax is defined at $\left(i, j + \frac{1}{2}, k + \frac{1}{2}\right)$. The vector potential is then multiplied with the forward rotation matrix, and the magnetic field on the interfaces is obtained using a standard first-order FD rotation operator. We observe $\nabla \cdot \boldsymbol{B} < 10^{-12}$ in all tests at all times.

In Figure 1, we compare the profile of the density contrast at $t = 10$ in one dimension with the analytic solution. Simulations are done with 8, 16, 32, and 64 zones and shown as violet diamonds, green square, blue crosses, and red circles, respectively. Dispersion is visible only at the lowest resolution, eight zones. This compares well with results shown in Wongwathanarat et al. (2016). In Figure 2, we plot the resulting L_1 error for resolutions between 8 and 128 zones run with WENO5-Z. From top left to bottom right, we show 1D hydrodynamic waves, 3D hydrodynamic waves, 1D MHD waves, and 3D MHD waves. In the 1D case, the simulation converges as N^{-5}, as expected, down to 10^{-11}, where the effect of wave steepening begins to limit further convergence of the compressive modes. We note that for $N > 128$, the round-off error from the 8 byte variables leads to an increase in L_1 again (see also Donnert et al. 2018). This suggests that the scheme needs to be run with at least 8 byte precision to take advantage of the high-order convergence properties.
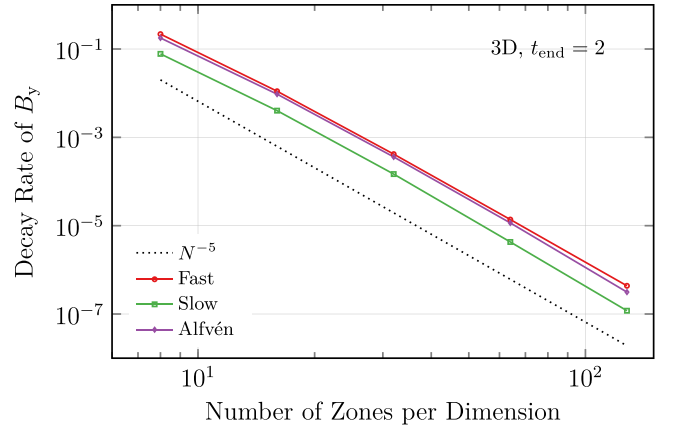
Figure 3. Decay rate (Equation (60)) of the y-component of the magnetic field between $t = 0$ and $t = 2$ for the fast mode (red), the slow mode (green), and the Alfvén mode (purple) with WENO5-Z.

In three dimensions, all hydrodynamic waves converge to fifth order. The entropy MHD wave converges to fifth order in L_1 as well. This mode does not feature a fluctuation in the magnetic field (Equation (53)); the background magnetic field is only advected. All other MHD waves converge to second order in L_1, because our CT scheme is geometrically only second order. However, the fluxes used are of course fifth-order accurate, thus this error should be dominated by dispersion, not dissipation (LeVeque 2002). The L_1 norm measures both errors, dispersion and diffusion.
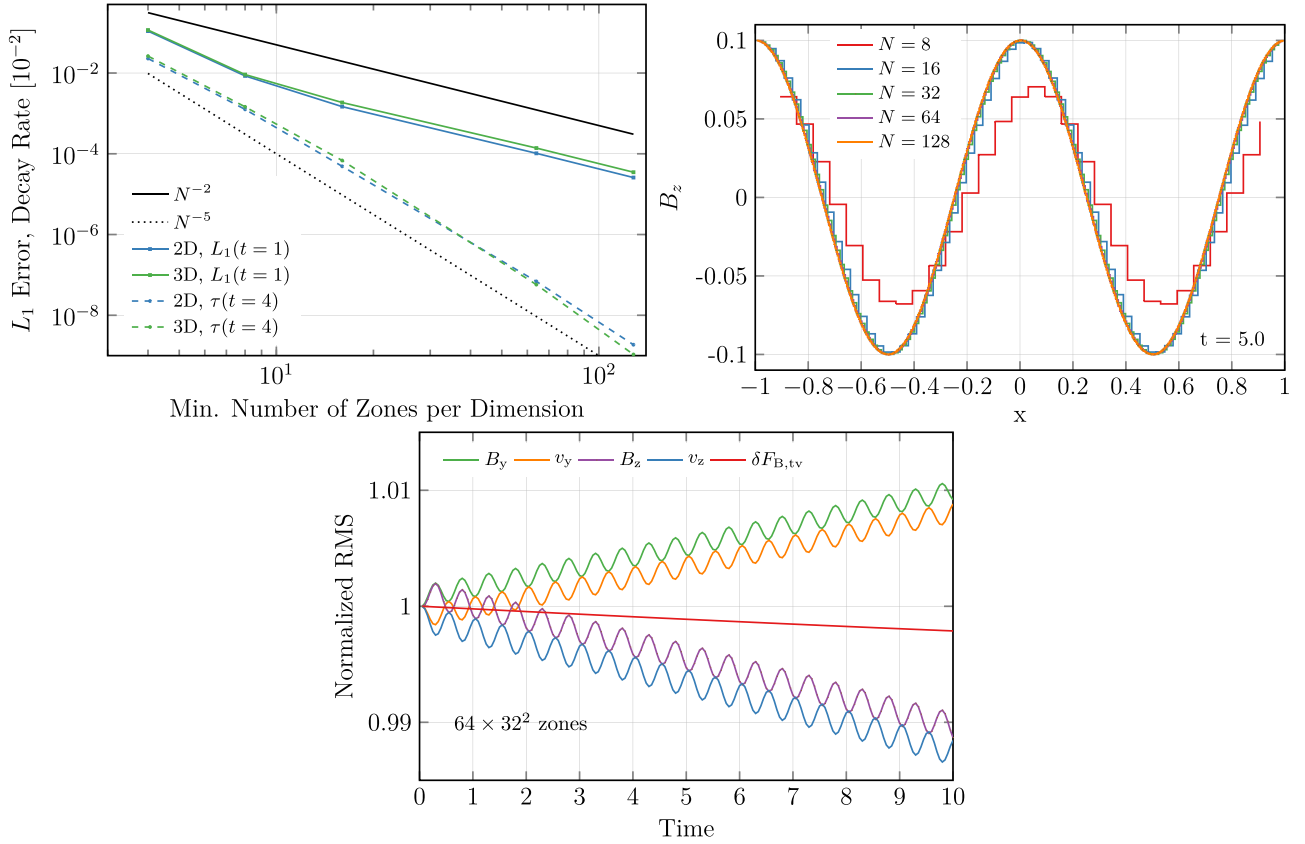
Figure 4. Top left: L_1 error of the circularly polarized Afvén wave test over resolution in 2D (blue) and 3D (green). Decay rate of the transverse magnetic flux is shown by the dashed lines in units of 100. We mark second order as solid and fifth order as dotted black lines. Top right: B_z for the circularly polarized Alfvén wave test in two dimensions at $t = 5$ with $N \in [8, 16, 32, 64, 128]$ in red, blue, green, purple, and orange, respectively. Right: time evolution of root-mean-square fluctuations around the background state in the Alfvén wave test with 64×32^2 zones for the state vector components that carry a fluctuation.

To further investigate this, we consider the decay rate of the magnetic field in the wave that is sensitive only to the dissipation error of the scheme (numerical diffusion), not the dispersion of the wave. The decay rate $\tau_{\rm D}$ of B_y is defined as

$$\tau_{\rm D}(B_y) = -\frac{1}{t_{\rm end}} \log\left(\frac{\delta B_y(t = t_{\rm end})}{\delta B_y(t = 0)}\right), \qquad (60)$$

where $\delta B_y(t)$ is the root-mean-square magnetic field fluctuation in the frame parallel to the wave vector. We plot the decay rate of the fast, slow, and Alfvén waves in Figure 3 between the initial conditions and time $t = 2$. The time was chosen so that all wave families have been advected through the domain at least once. It is important to evaluate the decay time at a late enough time, as the rms values fluctuate with time, indicative of additional waves being present in the test setup. Fifth-order convergence of the decay rate is observed for all three remaining MHD waves.

This confirms that the error introduced by the second-order CT scheme is indeed in the form of wave dispersion only, not dissipation, i.e., the price paid for the computationally cheap CT method from Ryu et al. (1998) is in the form of the magnetic field shape, but not magnetic energy conservation. This view is supported by results in the Alfvén wave and magnetic field loop advection tests shown later in the paper. Nonetheless, a comparison of the absolute value of the L_1 error with Stone et al. (2008 their Figure 32) yields that CT-WENO5 increases the effective resolution compared to ATHENA in three

dimensions by roughly a factor of 2, even for linear MHD waves propagating with an oblique angle to the grid in three dimensions. The L_1 error is significantly smaller even at $N = 16$, so the statement is true despite the second-order convergence for MHD waves.

4.2. Circularly Polarized Alfvén Waves

Polarized Alfvén waves are important in heating the solar corona due to instabilities and have been studied for many decades (e.g., Goldstein 1978; Del Zanna et al. 2001; Ruderman & Simpson 2005; De Pontieu et al. 2007). As a numerical test, polarized Alfvén waves can be used to evaluate the fidelity of the CT scheme (Tóth 2000).

Following Gardiner & Stone (2005), we set $\boldsymbol{U} = (1, 0, v_y, v_z, 1, B_y, B_z, 0.1)^T$, with $v_y = B_y = 0.1 \sin(2\pi x)$, $v_z = B_z = 0.1 \cos(2\pi x)$. In two dimensions, we rotate $\boldsymbol{U}$ by the second Euler angle $\theta = \tan^{-1}(2) \approx 63°5$. The computational domain has $2N \times N$ zones and covers $L_{{\rm Box},x}$, $L_{{\rm Box},y} = \sqrt{5}$, $\sqrt{5}/2$. In three dimensions, the wave is rotated similarly to Section 4.1. The computational domain of $2N \times N \times N$ zones covers $(L_x, L_y, L_z) = (3, 3/2, 3/2)$. Gardiner & Stone (2005) did not find an instability using these parameters. We note that the analytical rms at $t = 0$ is given as $0.1/\sqrt{2}$ for fluctuating quantities, while for all other quantities the initial rms is zero. Thus, the former errors dominate the simulation, while the latter quantities are subject to truncation errors from the rotation operation.
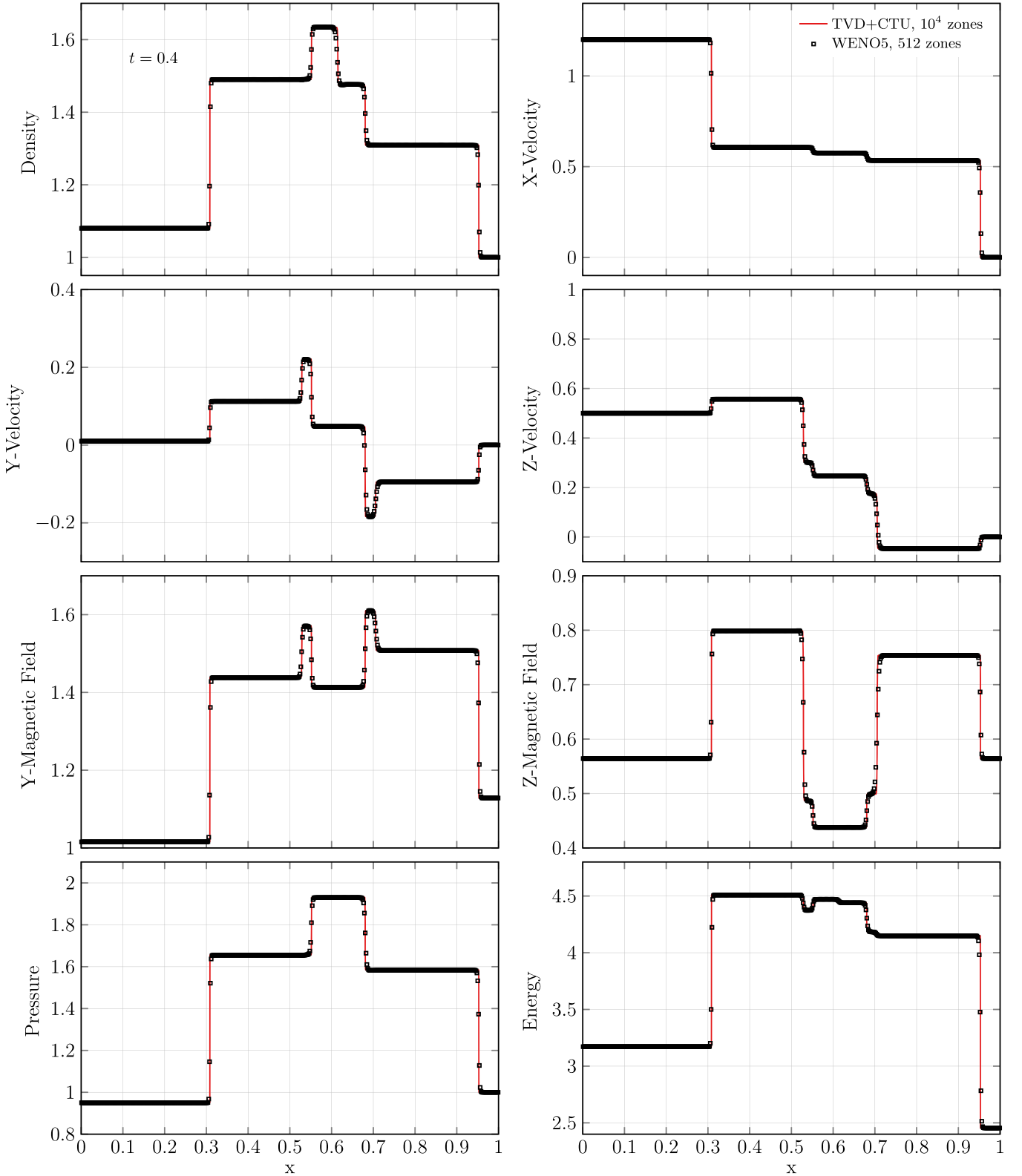
Figure 5. Shock tube test 2a from Ryu & Jones (1995) at time $t = 0.2$ in one dimension. The TVD+CTU result with 10,000 zones is shown by the red line, WENO5 result with 512 zones as black squares. Density, velocity components, magnetic field components, pressure, and energy are shown from top left to bottom right.

On the top left of Figure 4, we show the L_1 error norm in 2D (blue) and 3D (green) over the number of zones after the wave has traveled for a five-wave period ($c_A = 1$) in a frame parallel to the wave vector. The scheme converges at second order in L_1. This is in line with the result from the previous linear wave convergence test.

We also show the decay rate convergence as dashed lines in Figure 4. However, this time, oscillations in the rms of the state
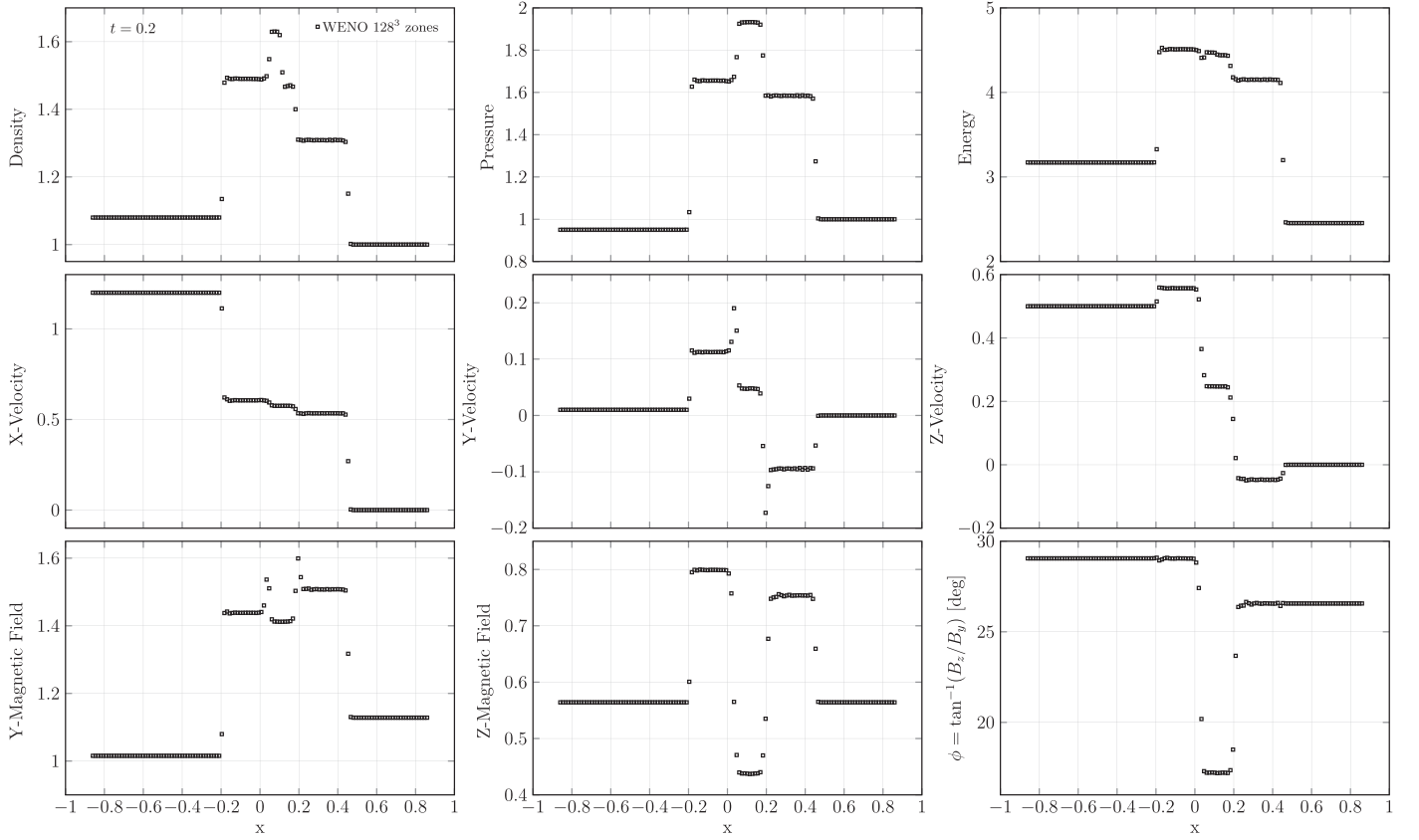
Figure 6. Shock tube test 2a from Ryu & Jones (1995) at time $t = 0.2$ in three dimensions along the zone diagonal of the computational domain. The WENO5 result with 128^3 zones is shown by the black squares. Density, pressure, energy, velocity components, magnetic field components, and magnetic field angle ϕ in degrees are shown from top left to bottom right.

vector components can lead to negative decay rates even at rather late times. In Figure 4, bottom, we show these rms fluctuations over time for 64×32^2 zones in three dimensions for the components that carry a fluctuation in the initial conditions. Dissipation in the solution is observed as asymmetry in the rms at late times. To obtain a robust measure for the wave decay, one may consider the rms of the transverse magnetic flux in the wave (red line):

$$\delta F_{\mathrm{B,tv}} = \mathrm{rms}(\sqrt{(B_y v_y)^2 + (B_z v_z)^2}) \qquad (61)$$

For this quantity, fifth-order convergence is observed at all times in Figure 4, top left.

In Figure 4, top right, we show the z-component of the magnetic field in two dimensions after advecting for five wave periods for resolutions of $N \in [8, 16, 32, 64, 128]$ in red, blue, green, purple, and orange, respectively. Only the lowest resolution shows some dissipation; dispersion is observed for the two lowest resolutions. This is a significant improvement over the results shown in Tóth (2000) for the same CT scheme. It also compares favorably with WOMBAT's TVD+CTU implementation (see Mendygral et al. 2017) and matches results from the ATHENA solver (Gardiner & Stone 2008, their Figure 4), despite the simpler CT scheme.

4.3. RJ95 2a Shock Tube

We compute shock tube number 2a from Ryu & Jones (1995) with WOMBAT's TVD+CTU and WENO5 solver. The shock tube features all four MHD modes. The left-hand state vector

is $\boldsymbol{U}_{\mathrm{L}} = (1.08, 1.2, 0.01, 2/\sqrt{4\pi}, 3.6/\sqrt{4\pi}, 2/\sqrt{4\pi}, 0.95)^T$. The right-hand state vector is $\boldsymbol{U}_{\mathrm{R}} = (1, 0, 0, 0, 2/\sqrt{4\pi}, 4/\sqrt{4\pi}, 2/\sqrt{4\pi}, 1)^T$. We plot the TVD+CTU result with 10^4 zones using the red line and the WENO5 result with 256 zones as black circles. The evolved state vector components at $t = 0.2$ for the one-dimensional computation are found in Figure 5; those in three dimensions are in Figure 6. For three dimensions, we rotated the shock normal along $\boldsymbol{k} = (1, 1, 1)^T$ in a 128^3 zone simulation. The primitive variables alongside the magnetic field angle $\phi = \arctan^{-1}(B_y/B_z)$ in degrees are shown in Figure 6.

4.4. RJ95 4d Shock Tube

We compute shock tube number 4d from Ryu & Jones (1995) with WOMBAT's TVD+CTU and WENO5 solver. The left-hand state vector is $\boldsymbol{U}_{\mathrm{L}} = (1, 0, 0, 0, 0.7, 0, 0, 1)^T$. The right-hand state vector is $\boldsymbol{U}_{\mathrm{R}} = (0.3, 0.01, 0.7, 1, 0, 0.2)^T$. We plot the TVD+CTU result with 10^4 zones using the red line and the WENO5 result with 256 zones as black circles. The evolved state vector components at $t = 0.16$ are shown in Figure 7.

4.5. Brio & Wu Shock Tube

The MHD shock tube from Brio & Wu (1988a) has the initial conditions $\boldsymbol{U}_{\mathrm{L}} = (0.125, 0, 0, 0, 0.75, 1, 0, 1)^T$, $\boldsymbol{U}_{\mathrm{R}} = (0.125, 0, 0, 0, 0.75, -1, 0, 0.1)^T$, with $\gamma = 2$. In Figure 8, we show the solution at $t = 0.08$ with 400 zones as black squares and the solution from TVD+CTU with 10^4 zones as the red line.
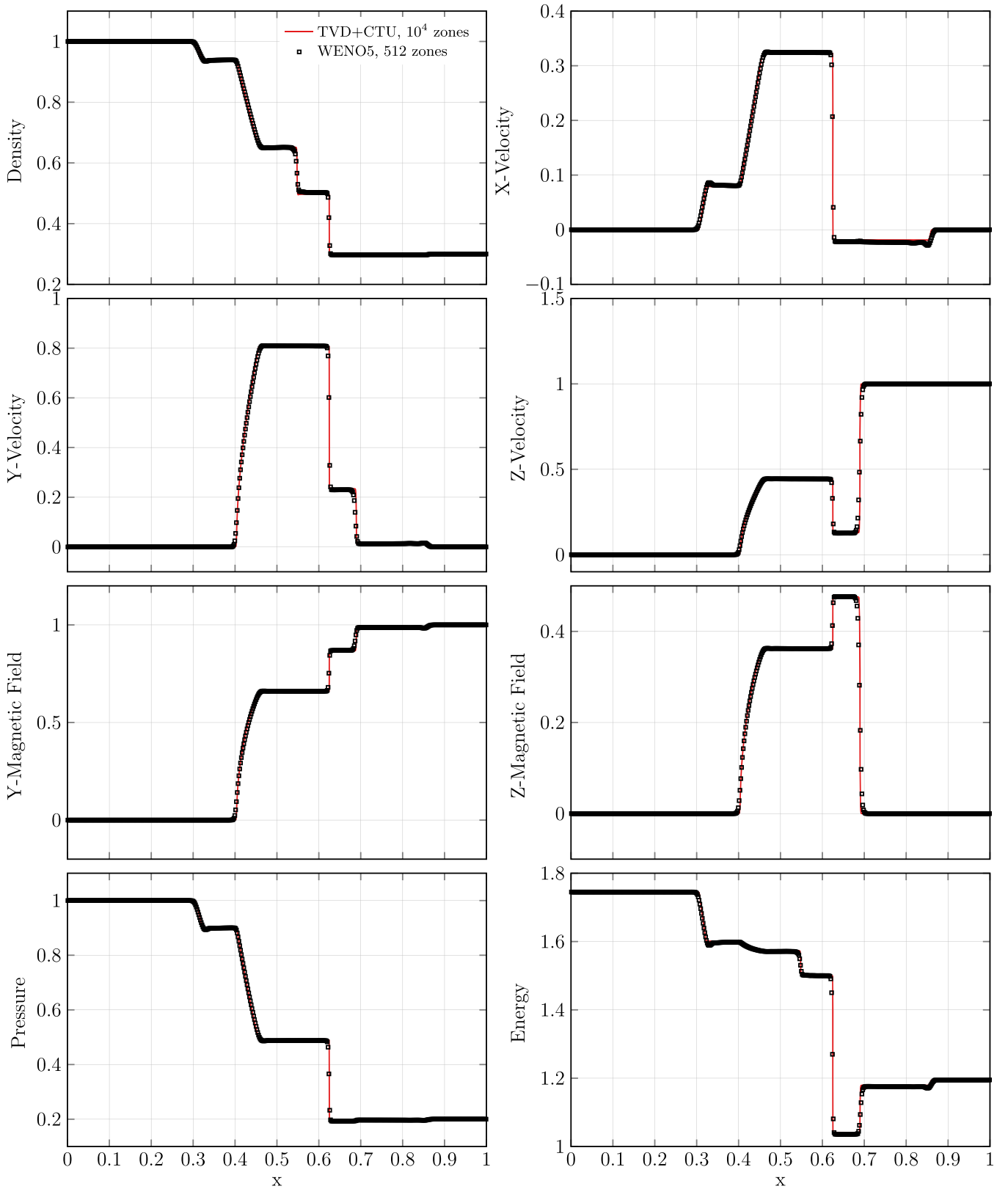
Figure 7. Shock tube test 4d from Ryu & Jones (1995) at time $t = 0.2$ in one dimension. The TVD+CTU result with 10,000 zones is shown by the red line, WENO5 result with 256 zones as black squares. Density, velocity components, magnetic field components, pressure, and energy are shown from top left to bottom right.

We note that this problem starts with a degeneracy in the magnetic field when computing the eigenvectors of the initial conditions in the single zone at $x = 0.5$. Following Brio & Wu (1988a), we resolve this degeneracy by setting $B_y = B_z = 1/\sqrt{2}$.

However, in this particular problem, $B_z = 0$ at all times. Thus, the test exposes this choice in the eigenvectors. Setting $B_z = 0$ for this problem only would remove the oscillations in Figure 8. Balbás & Tadmor (2006) have shown that the issue can also be fixed with
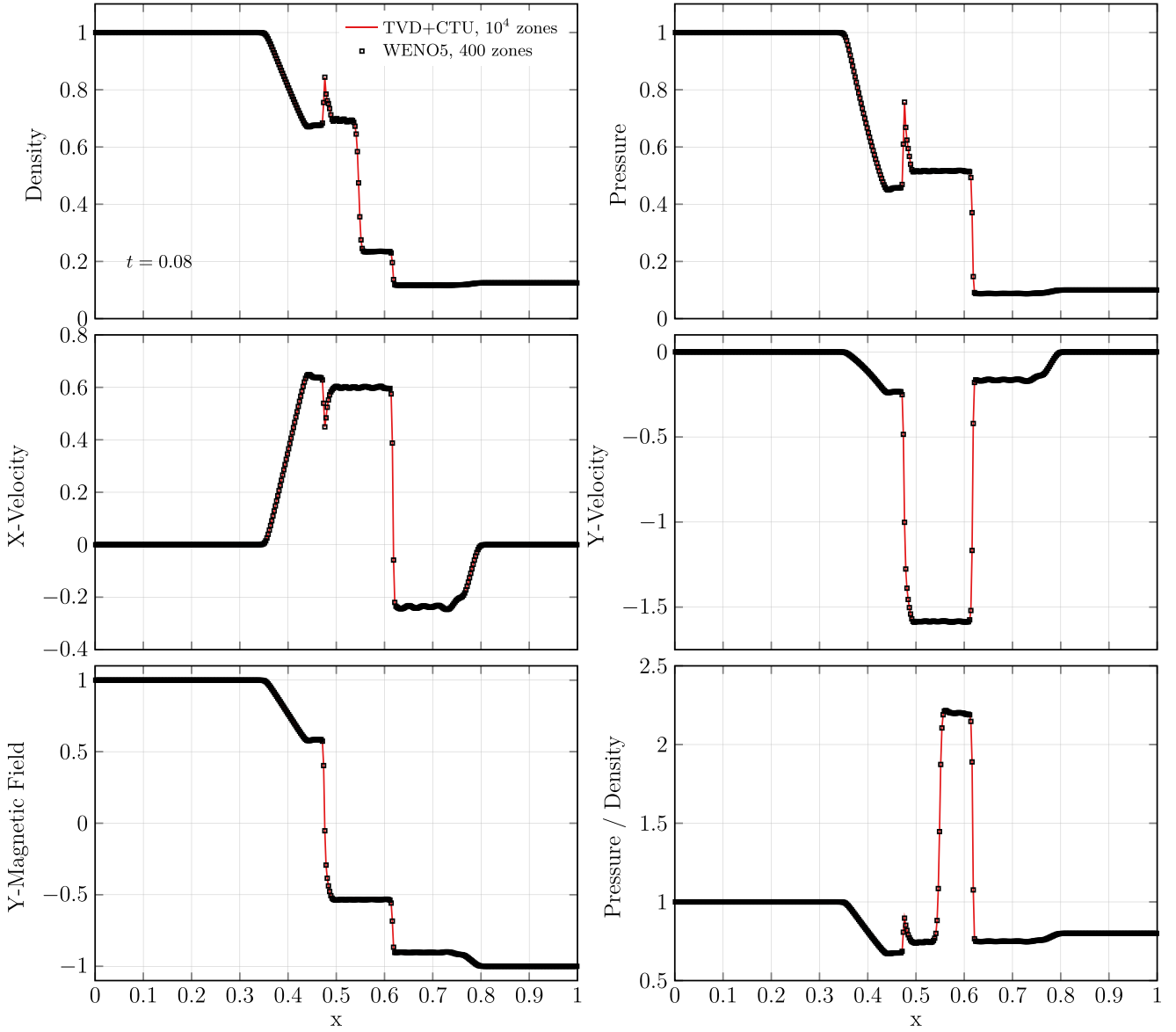
Figure 8. Shock tube test from Brio & Wu (1988b) at time $t = 0.08$. We show the density, x and y components of the velocity, y component of the magnetic field, pressure, and energy from top left to bottom right. Results from WENO5 with 400 zones are shown by the black squares, and those from TVD+CTU with 10,000 zones as the red line.

global smoothness indicators. In real world applications in two or more dimensions, this is not an issue. H. Jang et al. (2019, in preparation) found that the oscillations disappear with WENO3, i.e., in lower order codes, increased dissipation leads to a constant solution.

4.6. Shock-density Wave Interaction

The Shu–Osher shock tube simulates the interaction of a shock with a smooth flow (Shu & Osher 1989). Greenough & Rider (2004) used it to argue that second-order PPM methods give results comparable to third-order WENO methods and are thus computationally more efficient. It exposes that the standard WENO5 weights are at best third-order accurate in critical points (Borges et al. 2008).

The initial conditions contain a shock tube with left and right states $U_L = (3.857143, 2.629369, 0, 0, 0, 0, 0, 31/3)^T$ and

$U_R = (1 + 0.2\sin(5x), 0, 0, 0, 0, 0, 0, 1)^T$ on a domain of size $L_x = 10$. The shock is located at $x = -4$. The shock tube is evolved until $t = 2$.

In Figure 9, we show the result from the WENO5 (green circles) and WENO5-Z simulations (black squares) with 300 zones and TVD+CTU with 10,000 zones (red line). Clearly, the WENO-Z result traces the complex flow behind the shock more accurately than the WENO5 result.

Another variant of the test, run on a domain of $[-1, 1]$ and with a factor π in the sine function of the initial conditions, is shown in Figure 9, bottom. The result can be directly compared to Felker & Stone (2018), their Figure 5. WENO5 roughly performs as well as their PPM+RK4 scheme, while WENO5-Z traces the complex shock wave structure better than PPM +RK4. This is in line with expectations from the WENO literature (e.g., Borges et al. 2008).
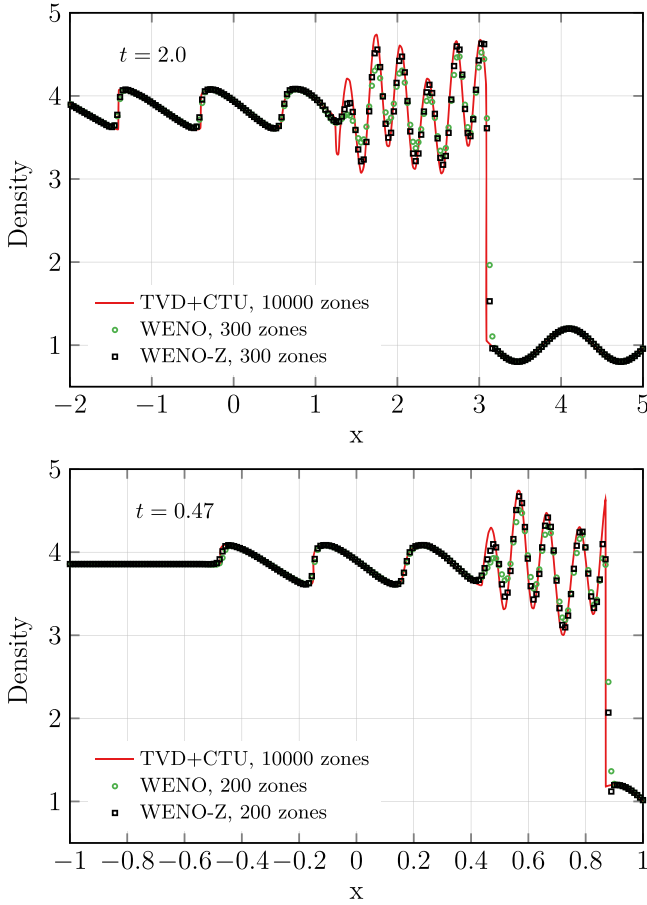
Figure 9. Top: density of the shock-density wave interaction test at time $t = 1$. Bottom: Shu–Osher test at $t = 0.47$ with 200 zones for WENO and WENO-Z, and 10,000 zones for TVD.

4.7. Gaussian Pulse

Following McCorquodale & Colella (2011), we simulate the time evolution of a Gaussian acoustic pulse in two dimensions. In a periodic domain of size one, we set

$$\rho(r) = \begin{cases} \rho 0 + (\delta \rho_0) e^{-16r^2} \cos(\pi r) & \text{if } r \leqslant \frac{1}{2} \\ \rho 0 & \text{otherwise} \end{cases} \quad (62)$$

$$p = \left(\frac{\rho}{\rho 0}\right)^{\gamma} \quad (63)$$

with $\rho_0 = 1.4$, $\delta\rho_0 = 0.14$, $\gamma = 1.4$. The simulation ran until $t = 0.24$.

We show the resulting density in Figure 10 at $t = 0$ (left) and at $t = 0.24$ (right). The color scale ranges from 0.136 to 1.54; we add contours at 1.375, 1.4, 1.425, 1.45, 1.475, 1.5, and 1.525.

To compare directly with McCorquodale & Colella (2011), we run a convergence study of the density differences at time 0.24. To this end, we compute the density error using cubic spline interpolation to obtain a reference solution from a run with 1024^2 zones. We obtain second-order convergence.

4.8. Implosion

The noise inherent in an implementation can be exposed in the two-dimensional implosion test (Liska & Wendroff 2003).

The setup leads a series of shock waves, which are intersecting many times in the simulation. The problem is highly nonlinear and thus not very useful to compare algorithmic fidelity. There is simply no convergence to a universal solution that results could be compared to. However, the nonlinearity ensures that computational noise gets amplified very quickly and solutions diverge strongly from correct, less noisy results. Using this test, we found noise injected by the compiler at the $\sqrt{10^{-16}}$ level in the eigenvector calculation (see the Appendix) by comparing the solution to TVD+CTU. This may be a word of warning for running Eulerian methods with 4 byte floating point precision, where numerical noise itself resides at 10^{-8} and thus will influence this test significantly. Many real world applications are strongly nonlinear and thus may not be correct with 4 byte variables.

We set $\boldsymbol{U} = (1, 0, 0, 0, 0, 0, 0, 1)^T$ everywhere in a periodic domain with size $L_x = L_y = 0.3$ and 200^2 zones. When $x + y < 0.15$, we set $\boldsymbol{U} = (0.125, 0, 0, 0, 0, 0, 0, 0.14)^T$. We show the resulting density in Figure 11 at $t = 0.2, 0.3, 0.5,$ and 0.7 (top left to bottom right) on a color scale ranging from 0.4 to 1.2. We notice that the symmetry in the simulation is not visibly broken, and the results are reasonably similar to the tests shown in Liska & Wendroff (2003). We show the vorticity of the test at $t = 0.3$ in Figure 12, where no noise is visible. In particular, the result is symmetric along the $x = y$ axis down to single zones.

The implosion test was used by Sijacki et al. (2012) to compare results between a traditional SPH and a Lagrangian finite volume method. Both show more computational noise than our Eulerian method due to particle motion. The same seems to be true for the low-order DG scheme presented in Mocz et al. (2014). There are sizable differences in the shape of the "bird" in the lower left corner between all these codes and our results. In particular, interfaces in the results of the Lagrangian FV and DG schemes differ from our WENO5 simulation. Numerical noise seems to trigger Raleigh–Taylor instabilities in the Lagrangian simulation that are absent in the Eulerian simulations. This noise is clearly visible in the vorticity maps (their Figure 3). Runs at much higher resolution show that indeed these interfaces do become unstable eventually. However, due to the nonlinearity of the flow, the high-resolution Eulerian result differs significantly from both low-resolution runs in that all instabilities grow faster in the high-resolution Eulerian runs. As mentioned above it is unclear which result is closer to the "true solution" in this test, and we conclude that numerical noise affects the growth of instabilities in highly nonlinear solutions of the hydrodynamic equations, which is of course not a new result at all (e.g., McNally et al. 2012).

4.9. Orszag–Tang Vortex

The Orszag–Tang vortex is a standard MHD test that mimics the transition of a smooth flow into 2D turbulence (Orzang & Tang 1979; Tóth 2000; Stone et al. 2008). The initial conditions in a unit domain are $\boldsymbol{U} = (25/(36\pi), -\sin(2\pi y), \sin(2\pi y), 0, B_x, B_y, B_z 5/(12\pi))^T$, where the magnetic field components are obtained from a vector potential with the z-component $A_z = (B_0/4\pi)\cos(4\pi x) + (B_0/2\pi)\cos(2\pi y)$, with $B_0 = 1/\sqrt{4\pi}$.

We show in Figure 13, top left to bottom right, the density, pressure, velocity magnitude, and magnetic field strength at time $t = 0.5$ at a resolution of 192^2 zones with WENO5-Z. In
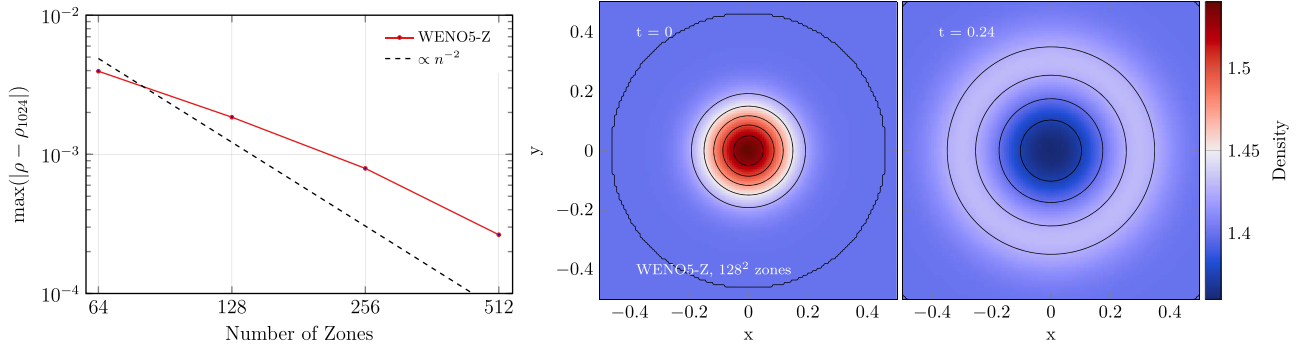
Figure 10. Left: convergence study of the Gaussian pulse test. Right: density of the initial conditions and at $t = 0.24$ in the Gaussian pulse test, run with WENO5-Z and 128^2 zones. Black contours are at 1.375, 1.4, 1.425, 1.45, 1.475, 1.5, and 1.525.
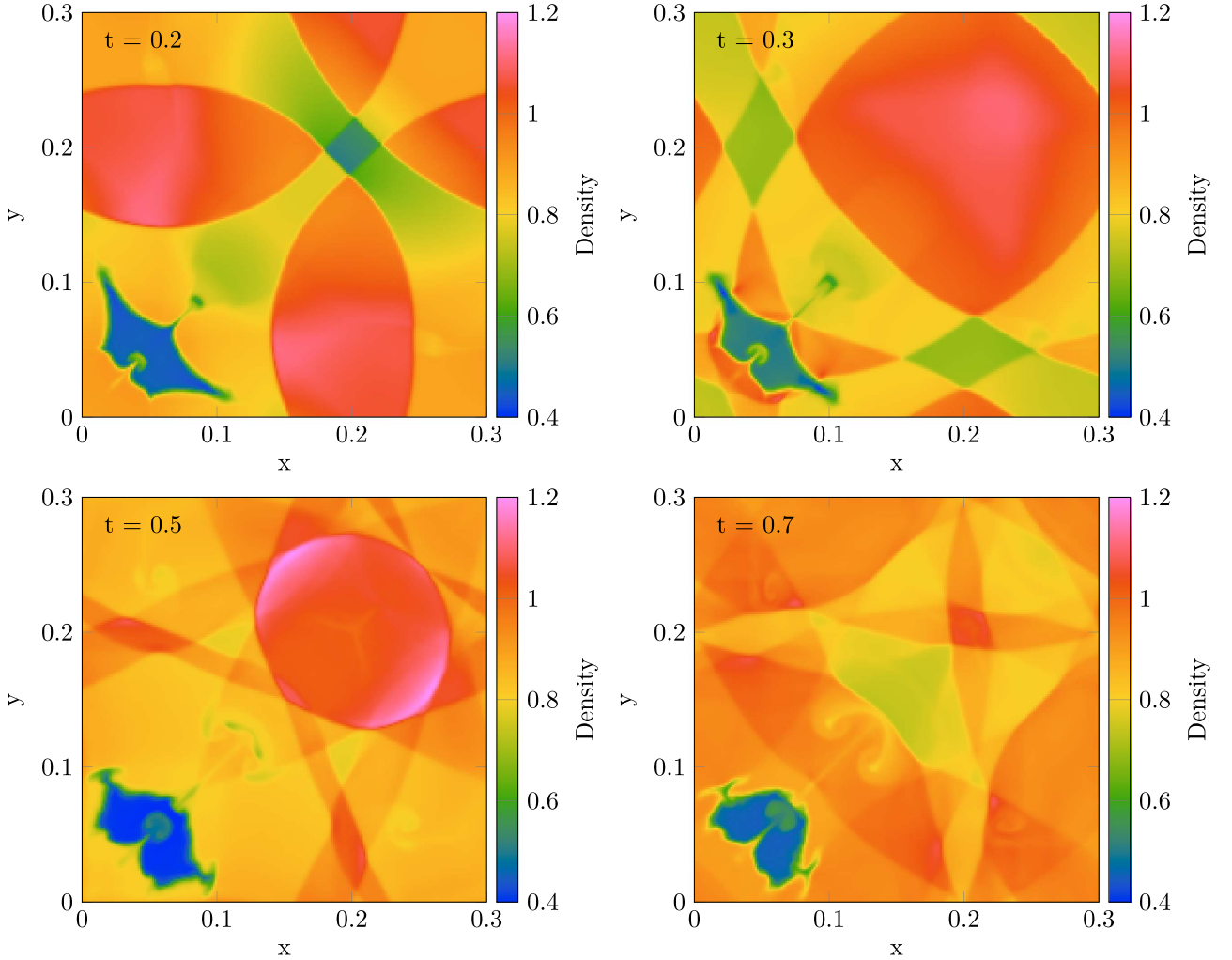

Figure 11. Density of the implosion test at times $t = 0.2$, $t = 0.3$, $t = 0.5$, and $t = 0.7$ (top left to bottom right). The color bar ranges from 0.4 to 1.2; the resolution was 200^2 zones.

Figure 14, we show the vortex at $t = 1$ run with 500^2 zones. Some schemes produce negative pressures at these late times. The results compare very well to the high-order code shown in Felker & Stone (2018). In particular, we see more structure in the central vortex and no break in symmetry. In Figure 15, we show the pressure in two slices through the domain at $t = 0.5$, $y = -0.1875$ (top) and $t = 0.5$, $y = 0.073$ (bottom). A high-resolution run with TVD+CTU is included as a reference solution.

In Figure 16, we compare the pressure at $y = -0.1875$, $-0.5 < x < 0$, and $t = 0.5$ of WENO5-Z with 128^2 zones (green squares) with TVD+CTU (blue circles) with 256^2 zones. The reference solution with 1024^2 zones is plotted by the red line. Apparently, WENO5-Z has fidelity comparable to or better than that of TVD+CTU at double the resolution in this test. In particular, the blip at $x = -0.45$ is better resolved in the WENO5-Z solution. A comparison with Figure 15 shows that the blip becomes fully resolved at 192^2.
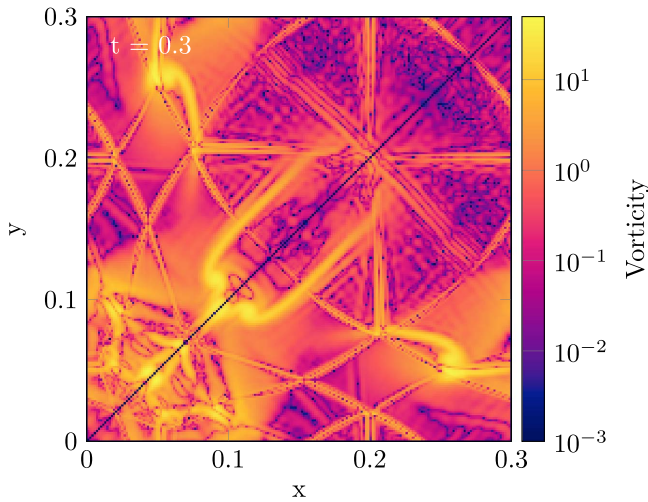
Figure 12. Logarithm of the vorticity of the implosion test at time $t = 0.3$. The color bar ranges from 0.001 to 50, the resolution was 200^2 zones.

In Figure 17, we compare Schlieren images (Hooke 1665; Helzel et al. 2011) of the density in TVD+CTU runs at 512^2 and 256^2 zones (left, middle), with the WENO5-Z run at 128^2. Following Guo & Zhang (2004), we obtained a synthetic brightness value by adding the red and yellow channels with a Gladstone–Dale constant of $K_{GD} = 1$, $L = 1$, and a contrast parameter of $C = 0.075$. Schlieren images visualize the gradient of the solution, which makes the Orszag–Tang vortex test more discriminating between schemes. We observe that all main features of the low-resolution TVD+CTU simulation are reproduced in the WENO5-Z simulation at half that resolution. We conclude that WENO5-Z doubles the effective resolution compared to TVD+CTU in this test.

4.10. MHD Rotor

The MHD rotor test simulates a rapidly rotating disk in two dimensions, which produces shocks and rarefaction waves as the disk expands due to centrifugal forces (Stone & Norman 1992; Balsara & Spicer 1999a; Stone et al. 2008). The solution needs to remain symmetric, and no spurious oscillations may occur. We set initial conditions with $\gamma = 5/3$ in a unit domain of $P = 1$, $\boldsymbol{B} = (5/\sqrt{4\pi}, 0)^T$, $v_0 = 2$, $r_0 = 0.1$, $r_1 = 0.115$, $f(r) = \frac{r_1 - r}{r_1 - r_0}$, and (Tóth 2000)

$$\rho = \begin{cases} 10 & \Leftrightarrow r < r_0 \\ 1 + 9f(r) & \Leftrightarrow r_0 \leqslant r \leqslant r_1 \\ 1 & \text{otherwise.} \end{cases} \quad (64)$$

The velocity components are

$$v_x = \begin{cases} -y\frac{v_0}{r_0} & \Leftrightarrow r < r_0 \\ -f(r)y\frac{v_0}{r_0} & \Leftrightarrow r_0 \leqslant r \leqslant r_1 \\ 0 & \text{otherwise} \end{cases} \quad (65)$$

$$v_y = \begin{cases} -x\frac{v_0}{r_0} & \Leftrightarrow r < r_0 \\ -f(r)x\frac{v_0}{r_0} & \Leftrightarrow r_0 \leqslant r \leqslant r_1 \\ 0 & \text{otherwise,} \end{cases} \quad (66)$$

We show the test result with WENO5 and 200^2 zones at time $t = 0.15$ in Figure 18. A slice through the rotor at $t = 0.5$ and $y = -0.1875$ (top) and $y = 0.073$ are shown in Figure 19. We show the WENO5 solution with black squares, and TVD +CTU with 1024^2 zones with the red line and with 400^2 zones with blue dots. No asymmetries or oscillations are visible; the WENO5 solution at 200^2 resolves the small features comparably to or better than TVD+CTU. A by-eye comparison with published results from ATHENA (Stone et al. 2008, their Figure 26), yields that WENO5 at half resolution is not quite as resolved as ATHENA at full resolution, likely because our CT scheme is only second order in space. There is some ringing visible in the density maps, which could likely be alleviated using global weights. We leave the improvement to future work.

4.11. Double Mach Reflection Test

The double Mach reflection test simulates the evolution of a Mach 10 shock at an oblique $60°$ angle to the grid with reflecting boundaries on the bottom x-axis, continuous upper x-boundary, and open y-boundaries (Woodward & Colella 1984; Stone et al. 2008). Here, the adiabatic index $\gamma = 1.4$ for air, the state ahead of the shock is $U = (1.4, 0, 0, 0, 0, 0, 0, 1)^T$, and the domain size is 3.25×1. Because the shock is reflected off the bottom wall, a second weak shock and a jet form. In the upper boundary, the evolution of the shock is followed analytically, i.e., $x_{shock} = 1/6 + (1 + 20t)$. As the shock in the setup is only one cell wide, the initial conditions contain an error, which is also continuously injected at the upper boundary. A one-cell-wide shock is not a "natural" solution to the numerical scheme, similar to the common shock tube setup used in SPH code tests. The test foremost exposes the robustness of the numerical scheme that has to handle these errors by injecting numerical diffusion. This differs from SPH shock tube tests, where dissipation and conduction has to be explicitly added. This test produces negative pressures when run with the WENO5-Z scheme without protection floors or protection fluxes.

In Figure 20, we show the density from the test run with the standard WENO5 scheme at times $t = 0.05$, 0.1, and 0.2 (top to bottom). We also overlay 30 contours between 1.73 and 20.92, which can be directly compared to Woodward & Colella (1984), their Figure 4. In the top panel, the error of the initial shock is visible in the color scheme as two stripes with slightly lower density left of and parallel to the shock. Over time, another such dip forms, starting at the location of the shock at the upper y-boundary traveling toward the lower left.

A zoom into a simulation with 1024×256 zones at $t = 2$ is shown in Figure 21 and can be directly compared to McCorquodale & Colella (2011), their Figure 7. In some codes, the reflection of the shock off the bottom y-boundary causes a carbuncle instability, because the Mach number is very high here. We do observe this instability with the standard WENO5 scheme, although only at high resolution.
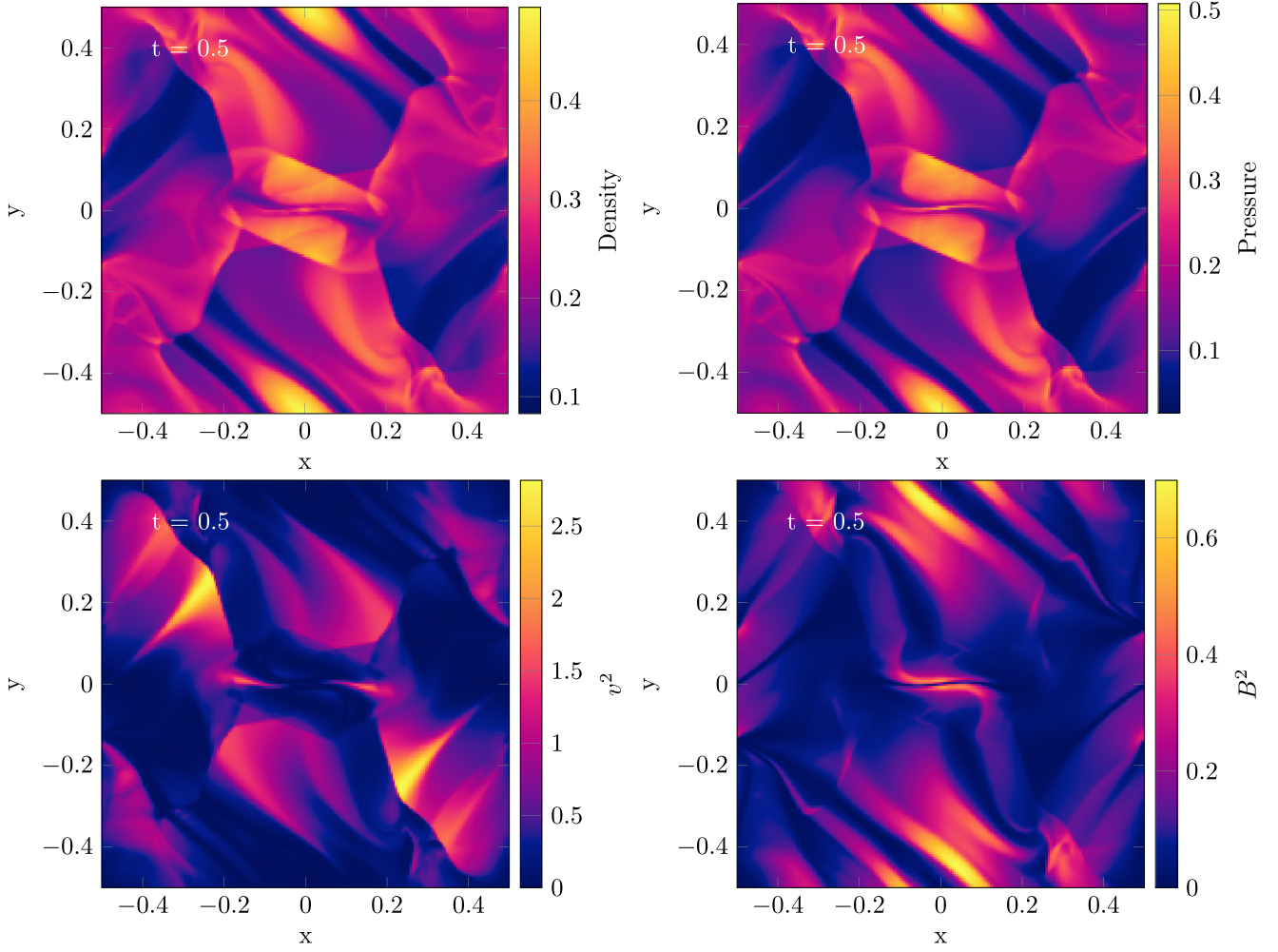
Figure 13. Images of the Orszag–Tang vortex test at time $t = 0.5$ with 192^2 zones. Top left to bottom right: density, pressure, v^2, and B^2.
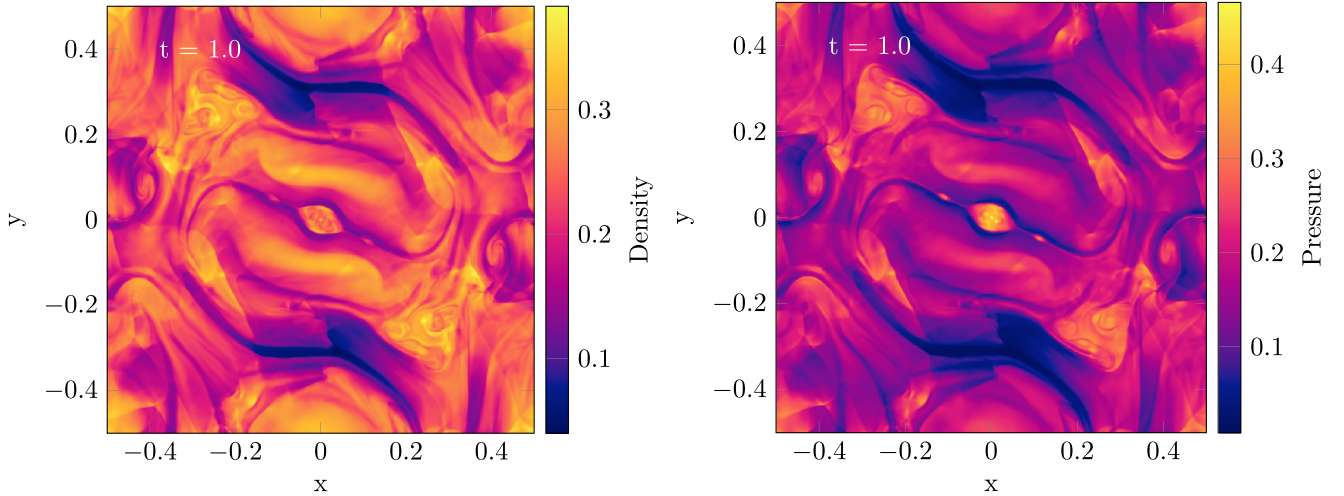

Figure 14. Density (left) and pressure (right) of the Orszag–Tang vortex as $t = 1$ with 500^2 zones.

4.12. Kelvin–Helmholtz Instability

The growth of instabilities plays a crucial role for mixing and the injection of vorticity in astrophysical fluid flows (e.g., Sheardown et al. 2018). We evaluate the performance of the code using the Kelvin–Helmholtz (KH) test, where the kinetic energy of a slightly perturbed shear flow generates vorticity at the interface (e.g., Parker 1958; Miura & Pritchett 1982; Frank et al. 1996). This test has been used extensively in the literature to test hydrodynamic codes (e.g., Springel 2010; McNally et al. 2012; Hopkins 2015; Schaal et al. 2015; Beck et al. 2016). In particular, the KH problem was used to expose the artificial surface tension of traditional SPH methods (e.g., Hopkins 2013). It also exposes
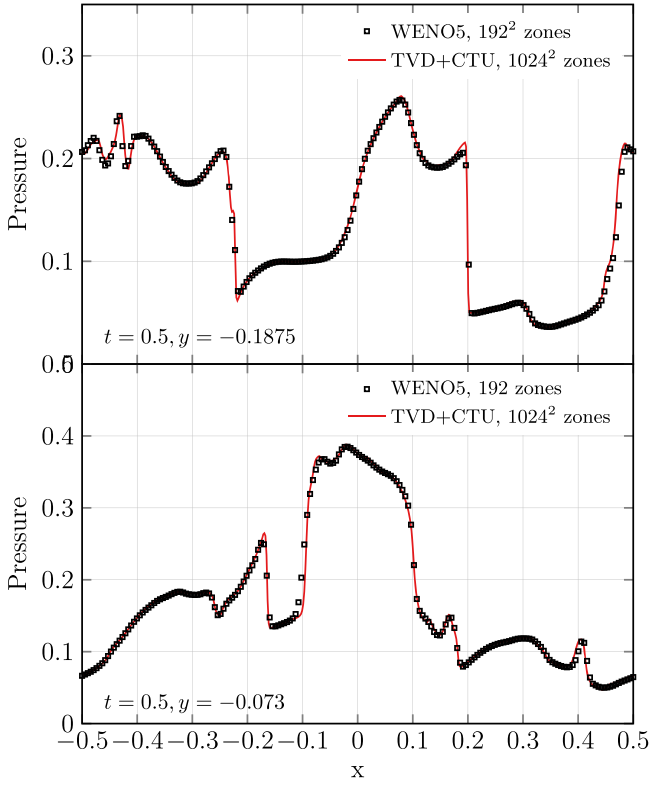
Figure 15. Pressure in the Orszag–Tang vortex test at $y = -0.1875$ (top) and $y = 0.073$ (bottom) and at time $t = 0.5$. WENO5-Z with 192^2 zones is shown by the black squares, TVD+CTU with 1024^2 zones by the red line.

the velocity-dependent truncation error in Eulerian grid methods (Robertson et al. 2010). Our setup follows Lecoanet et al. (2016):

$$\rho = 1 + \frac{\Delta\rho}{\rho_0}\frac{1}{2}\left[\tanh\left(\frac{z - z_1}{a}\right) - \tanh\left(\frac{z - z_2}{a}\right)\right] \quad (67)$$

$$v_x = v_{\text{flow}}\left[\tanh\left(\frac{z - z_1}{a}\right) - \tanh\left(\frac{z - z_2}{a}\right) - 1\right] \quad (68)$$

$$v_z = A\sin(2\pi x)\left[\exp\left(-\frac{(z - z_1)^2}{\sigma^2}\right) + \exp\left(-\frac{(z - z_2)^2}{\sigma^2}\right)\right] \quad (69)$$

$$P = P_0, \quad (70)$$

with the parameters $A = 0.01$, $P_0 = 10$, $a = 0.05$, $\sigma = 0.4$, $z_1 = 0.5$, $z_2 = 1.5$, $u_{\text{flow}} = 1$, and $\Delta\rho/\rho_0 = \rho_0 = 1$. We note that the instability is not resolved in our runs with this choice of parameters, i.e., the instability grows too slowly due to numerical effects. Simulations are run in a periodic domain with $L_y = 2$, $L_x = 1$.

In Figure 22, we show the density of four simulations at $t = 2$. The top-left to bottom-right panels show WENO5-Z without boost with 512×1024 zone resolution, WENO5 with 32×64 zones without a velocity boost, WENO5 with 32×64 zones boosted by $v_x = 50$, and TVD+CTU with 128×256 zones without boost.
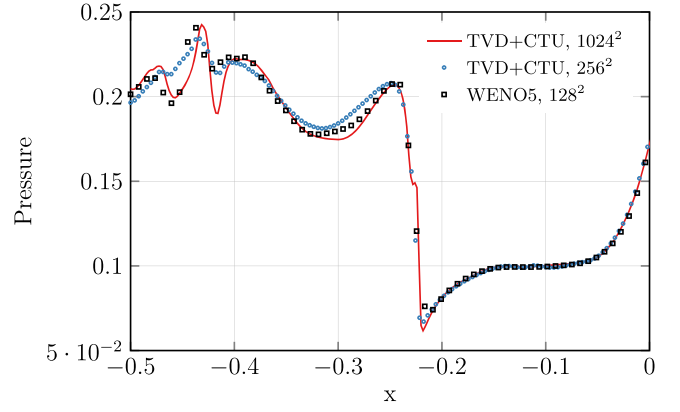
Figure 16. Pressure in the Orszag–Tang vortex test at $y = -0.1875$, $-0.5 < x < 0$, and at time $t = 0.5$. WENO5-Z with 128^2 zones is shown by the black squares, TVD+CTU with 1024^2 zones by the red line, and TVD+CTU with 256^2 zones by the blue circles.

The TVD+CTU run does not develop a vortex roll due to numerical diffusivity, even at twice the resolution from the WENO5 runs. In contrast, the low-resolution WENO5 runs develop a vortex similar to the high-resolution simulation (see Lecoanet et al. 2016). This shows that WENO5 resolves instabilities much better than the second-order TVD+CTU, even at half the grid resolution. WENO5 is also much less susceptible to advection errors. The boost of $v_x = 50$ (!) barely changes the result. Some differences are visible at the tip of the roll, because the truncation error is not Galilean invariant. Nonetheless, we expect significant improvements in the growth of instabilities in cosmological simulations, where advection is commonplace and may be a major source of diffusivity for second-order Eulerian methods (Springel 2010).

4.13. Gresho Vortex

The Gresho vortex (Gresho & Chan 1990; Liska & Wendroff 2003; Springel 2010) is a two-dimensional rotating structure in hydrodynamic equilibrium that evaluates the angular momentum and vorticity conservation of the code. The test becomes very challenging for Eulerian codes, when the vortex is advected to the grid. It exposes how the increase in truncation error affects angular momentum conservation. Instructive discussions can be found in Miczek et al. (2015) and Wongwathanarat et al. (2016).

In a two-dimensional unit domain, we set

$$\boldsymbol{U} = (1, v_x(r), v_y(r), 0, 0, 0, 0, P(r))^T, \quad (71)$$

with the radius r and[10]

$$v_x(r) = -v(r)\cos(\phi) + v_{\text{boost}}, \quad (72)$$

$$v_y(r) = -v(r)\sin(\phi), \quad (73)$$

$$\phi = \tan^{-1}\left(\frac{y}{x}\right) \quad (74)$$

$$v(r) = 0.4\pi\begin{cases} 5r & \Leftrightarrow r \leqslant 0.2 \\ 2 - 5r & \Leftrightarrow r < 0.4 \\ 0 & \text{otherwise} \end{cases} \quad (75)$$

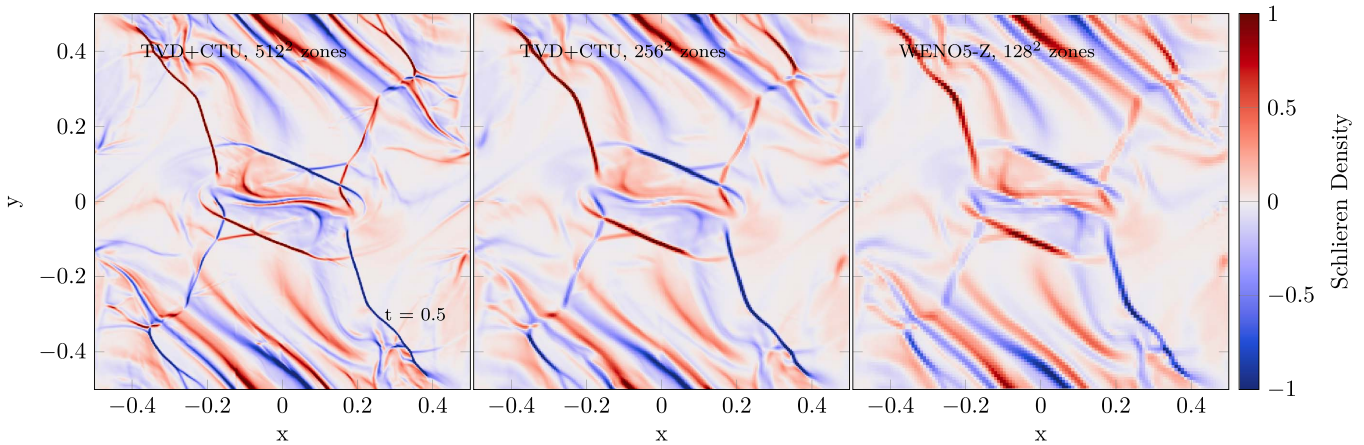[10] We use the Fortran ATAN2 function here.

Figure 17. Schlieren image of the density in the Orszag–Tang vortex test at $t = 0.5$ for TVD+CTU with 512^2 zones (left) and 256^2 zones (middle), and WENO5-Z with 128^2 zones (right).

$$P(r) = \begin{cases} p_0 + \frac{25}{2}r^2 & \Leftrightarrow r \leqslant 0.2 \\ p_0 + \frac{25}{2}r^2 + 4\left(1 - 5r + \log\left(\frac{r}{0.2}\right)\right) & \Leftrightarrow r < 0.4 \\ p_0 - 2 + 4\log(2) & \text{otherwise} \end{cases}$$

$$(76)$$

$$p_0 = \frac{(0.4\pi)^2}{\gamma M_{\max}^2} - \frac{1}{2}. \tag{77}$$

The tests shown in Springel (2010) correspond to $p_0 = 5$, a relatively high maximum Mach number of $M_{\max} \approx 0.41$.

We run a convergence study of the test with the WENO5-Z algorithm using three different boost velocities in the x-direction: 0, 1, 3. The results are shown in Figure 23 where we show the L_1 error over resolution. Blue, green, and purple lines correspond to boost velocities of 0, 1, 3, respectively. We add a dotted line scaling with $N^{-1.4}$, which corresponds to the scaling obtained with AREPO in Springel (2010). In this regime, our results compare quite favorably to the results obtained with lower order FV codes. In particular, the error remains small even with the velocity boost at most resolutions and is smaller than the errors observed with ATHENA at all times.

The boost mostly affects resolutions below 100^2, where the vortex becomes underresolved. Below 20^2, the L_1 error with velocity boost is reduced again. We note that the error changes depending on how the center of the vortex is placed on the grid, likely because the initial conditions contain a discontinuity at $r = 0.2$. As discussed in Muñoz et al. (2013), the sampling of the discontinuity affects the convergence properties of the simulations.

To illustrate this further, we show in Figure 24 (left to right) the angular velocity v_ϕ, pressure, and vorticity $\omega(r) = \nabla \times \boldsymbol{v}$ over radius. We compute

$$v(\phi) = r\frac{xv_y - yv_x}{x^2 + y^2}, \tag{78}$$

and the vorticity from the standard FD operator. Analytic profiles are shown as red lines, the initial conditions as red dots, the simulation at $t = 3$ without boost as black dots, and the simulation with velocity boost of $v_x = 3$ as blue dots. From top to bottom, we show resolutions of 16^2, 32^2 and 64^2 zones. The results for the static compare well with the results of the

low-order DG scheme on a rectangular grid shown in Mocz et al. (2014), their Figure 5. Low-order DG codes on polar grids, however, show better conservation properties in vorticity (e.g., Velasco Romero et al. 2018, their Figure 3). The run with 32^2 zones and velocity boost shows the largest scatter, in line with the increase in L_1 error observed before. At 16^2 zones, the profiles show less scatter, but are only sampled with 15 unique values below $r = 0.4$ in the ICs. At this low resolution, the problem is underresolved, and it stands to reason that the reduced scatter in the evolved boosted profiles is a result of the poor sampling. In particular, the discontinuous peak of the v_ϕ profile is not sampled directly, which should lead to additional error (Springel 2010; Muñoz et al. 2013). We speculate that the pronounced increase in L_1 error around 32^3 zones is a feature of the FD WENO5 reconstruction without cross-terms. Indeed, Liska & Wendroff (2003) found a relatively high L_1 error in total kinetic energy for their WENO5 scheme at low resolutions.

To fully expose the fidelity in momentum conservation, we show the normalized velocity at $t = 1$ with 40^2 zones in Figure 25. From left to right, we show runs with decreasing $M_{\max} = 10^{-1}, 10^{-2}, 10^{-3}$. The vortex clearly loses integrity at low Mach numbers. This result can be compared directly to Wongwathanarat et al. (2016), their Figure 3. Following the discussion in Miczek et al. (2015), this result is likely due to the scaling of the Roe fluxes, and the missing cross-terms in our FD WENO scheme.

We conclude that the stationary solution with WENO5 is competitive with the AREPO and ATHENA solutions presented in Springel (2010), but not with high-order DG codes (Velasco Romero et al. 2018) or FV codes on polar grids (Wongwathanarat et al. 2016). At relatively high Mach numbers, the error in the boosted solutions is comparable with second-order static Cartesian mesh solutions at intermediate resolutions and approaches the Lagrangian error at resolutions above 100^2 zones.

4.14. Advection of a Density Square

The velocity-dependent truncation error of Eulerian grid methods can be clearly exposed in the density advection test (Robertson et al. 2010; Hopkins 2015). In two dimensions, a density square in hydrodynamic equilibrium with the
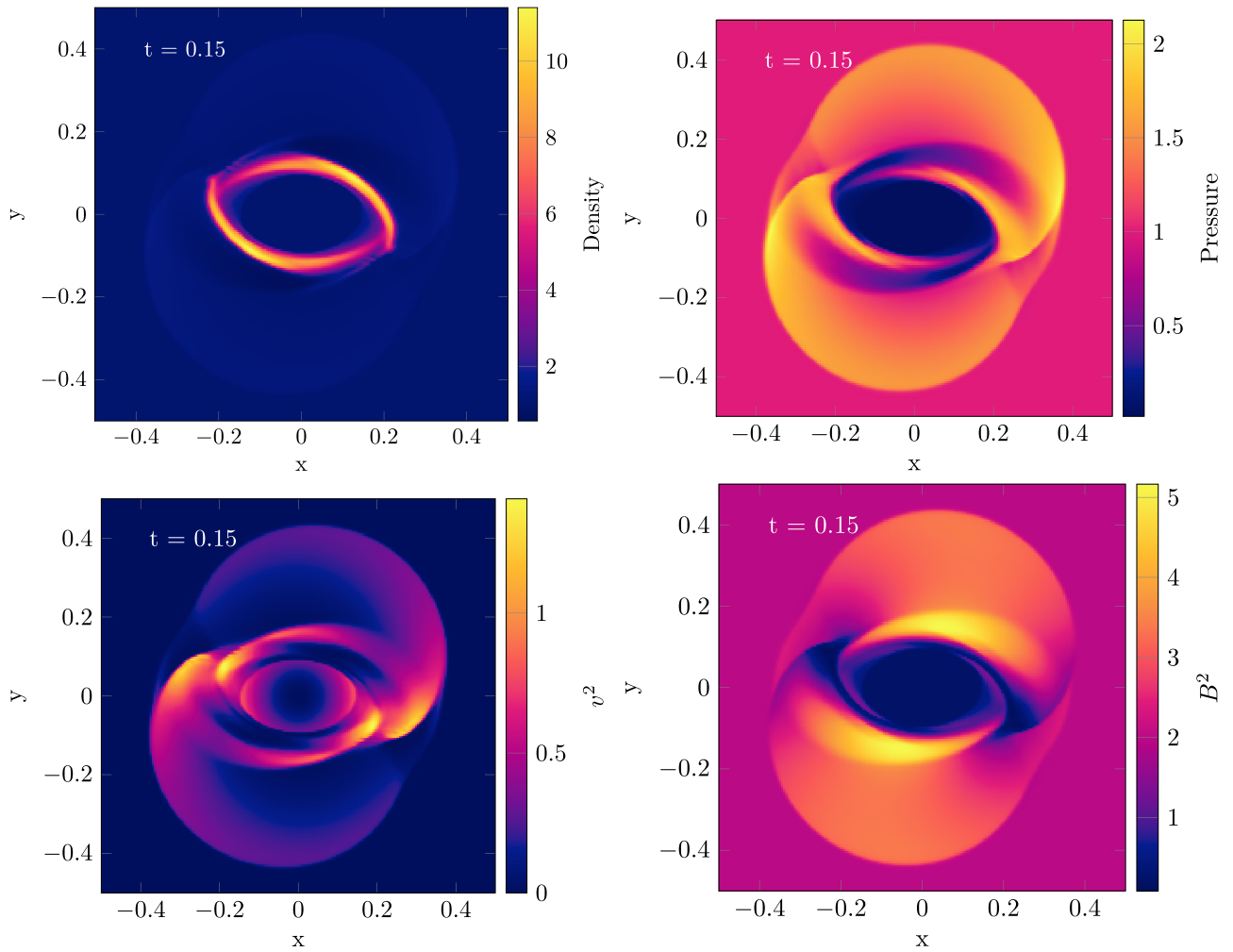
Figure 18. Images of the MHD rotor test at time $t = 0.15$ with 400^2 zones. Top left to bottom right: density, pressure, v^2, and B^2.

surrounding medium is advected supersonically through a unit domain: $\boldsymbol{U} = (1, 100, 50, 0, 0, 2.4)^T$, except $\rho = 4$, when $-0.25 < x < 0.25$ and $-0.25 < y < 0.25$. Here, $\gamma = 5/3$, thus $c_s = 2$, so the advection is supersonic ($M > 100$) with respect to the reference frame of the mesh. Although this test is trivial with Lagrangian schemes such as SPH, meshless finite volume, and moving finite volume schemes, it presents a very serious challenge to Eulerian grid methods.

We show the result of the test run with TVD+CTU, WENO5, and WENO5-Z in Figure 26 at time $t = 10$. We also include a WENO5-Z run with CFL = 0.2. The convergence rate is shown in Figure 27.

Due to the high velocity relative to the grid, the solution is smeared out significantly by the TVD+CTU algorithm, as expected from a second-order method. WENO5 performs a bit better, although still with significant dispersion. This is because the edges of the density square are critical points in the flow, where the classical WENO5 weights are only third-order accurate. WENO5-Z performs significantly better, with the density square being preserved, but strongly distorted. Quantitatively, WENO5-Z shows an improvement in L_1 error by roughly a factor of 2 at late times.

These results can be directly compared to the results shown with the discontinuous Galerkin (DG) code TENET in Schaal et al. (2015). In terms of L_1 error, WENO5 performs roughly

like a second-order DG code. WENO5-Z performs better than second-order DG, but worse than third-order DG. We note that increasing the convergence order in a DG scheme reduces the time step by a factor of 2. Thus, the time step of the DG3 scheme is 20 times smaller than in our WENO5-Z run (compare their Equation (24), CFL = 0.2, with our Equation (17), CFL = 0.8, in three dimensions). Thus, these DG schemes are more computationally expensive than WENO5, where the time step does not change with the order of the scheme.[11] DG schemes also require more memory, because the full polynomial has to be stored in every zone in one way or another. We ran the WENO5-Z simulation with a reduced CFL number of 0.2 to approach the computational cost of a DG3 algorithm. The distortion of the square is reduced further, but the L_1 error remains roughly the same.

This result suggests running strongly advection-dominated problems with WENO5-Z, not with the classical WENO5 method. As the method is significantly less robust, this results in a trade-off between more protection fluxes and reduced advection error for complex problems. It will depend on the problem under consideration on which approach delivers better results.

---

[11] This result is independent of architecture if both codes are properly vectorized and cache-blocked, and compilers are mature.
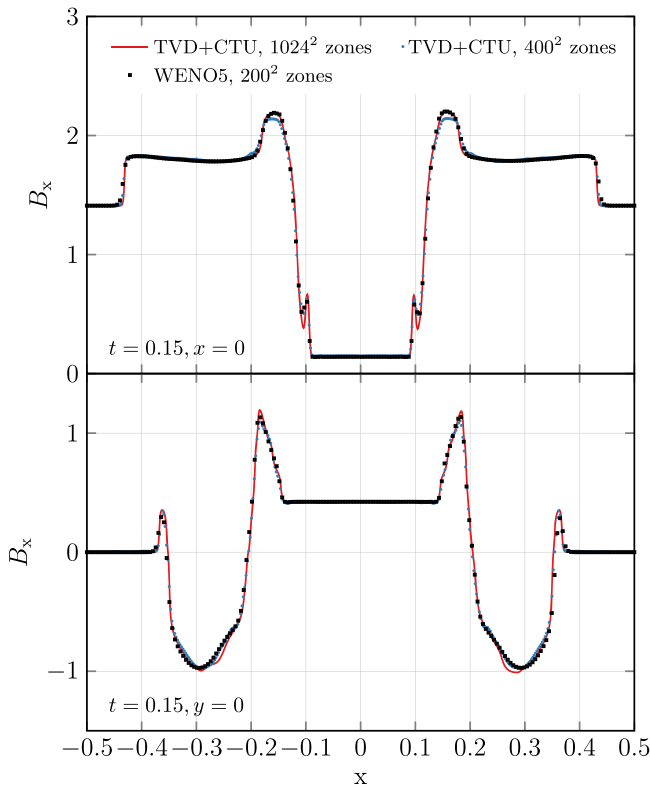
Figure 19. *x*-component of the magnetic field (top) and *y*-component of the magnetic field (bottom) in the MHD rotor test at $y = -0.1875$ (top) and $y = 0.073$ (bottom), at time $t = 0.5$. WENO with 200^2 zones is shown by the black squares, TVD+CTU with 400^2 zones by blue dots, and TVD+CTU with 1024^2 zones by the red line for reference.

4.15. Advection of a Magnetic Field Loop

The field loop advection test is instrumental for the development and testing of CT schemes for the magnetic field evolution in Eulerian codes (Gardiner & Stone 2008). This is a difficult test for the CT scheme of the code, as only the electric field and magnetic field are moving through the domain. In applications, this will rarely be the case.

In two dimensions, we set up a periodic domain of size $L_x \times L_y = 2 \times 1$ with $\boldsymbol{U} = (1, 2, 1, 0, B_x, B_y, 0, 1)^T$ and a vector potential on the interfaces with zero x and y components, but

$$A_z = A_0(r_c - r), \tag{79}$$

where $A_0 = 10^{-3}$, $r_c = 0.3$. The magnetic field on the interfaces is then obtained as the curl of the vector potential on the faces. In Figure 28, we show the resulting magnetic field energy density of the initial conditions (left) and the simulation at time $t = 2$ with TVD+CTU (middle) and WENO5 (right) with 128×64 zones. The WENO5 scheme keeps the shape of the loop well while it is advected through the box. The shape of the loop is comparable to second-order CT schemes (Stone et al. 2008; Lee 2013). In contrast, the TVD+CTU loop is barely visible; most of the magnetic energy has dissipated into heat. To quantify this, we show the time evolution of the normalized magnetic energy in three runs with 32, 64, and 128 zone base resolution in Figure 29. We mark 98% by the dotted horizontal line.

WENO5 retains more than 90% of the magnetic energy of the field loop, even at the lowest resolution run, while TVD+CTU loses more than half of the magnetic energy until that time, even at the highest resolution. The conservation of magnetic energy also compares favorably with the results shown in Lee (2013) and Felker & Stone (2018). Even though our CT scheme is much less elaborate than theirs, the magnetic field fluxes are fifth-order accurate and thus improve energy conservation. Again, WENO5 roughly doubles the effective resolution of the simulation, i.e., our run with 64^2 zones conserves magnetic energy roughly like ATHENA with 128^2 zones. These results are in line with our findings from the wave convergence tests, where wave diffusion is fifth-order accurate, but wave dispersion is second-order accurate. The structural integrity of our field loop is comparable to the results in Felker & Stone (2018), which is third order in space and fourth order in time. As expected, WENO5 with CT conserves magnetic energy better than Felker & Stone (2018), due to the fifth-order fluxes. The scheme also compares well with the FDs schemes shown in Mignone et al. (2010). This shows the advantage of the CT approach over divergence-cleaning methods, despite the second-order convergence in the linear wave advection tests. The fourth-order FV CWENO code presented in Verma et al. (2019) shows even better conservation properties. The CWENO code retains 98% of magnetic energy at $t = 100$ for 128^2 zones. These results place our FD WENO5 scheme in between high-order FV WENO codes and low-order FV codes or high-order codes without CT. Considering the usual trade-off in efficiency and fidelity, our simple but fast CT approach sacrifices some magnetic field structure for computational efficiency, but not magnetic energy conservation. In three dimensions, the loop is rotated around the *y*-axis by $\phi = \arctan(0.5)$ (Gardiner & Stone 2008). We set up the magnetic vector potential as

$$A_x = -A_0 \frac{(r_c - r^\star)}{\sqrt{5}} \tag{80}$$

$$A_y = 0 \tag{81}$$

$$A_z = -A_0 \frac{2(r_c - r^\star)}{\sqrt{5}}, \tag{82}$$

where $r^\star$ is the radius obtained from the coordinates

$$x^\star = \begin{cases} \frac{(2x+z)+2}{\sqrt{5}} & \Leftrightarrow x < -\sqrt{5} \\ \frac{(2x+z)-2}{\sqrt{5}} & \Leftrightarrow x > -\sqrt{5} \end{cases} \tag{83}$$

$$y^\star = y \tag{84}$$

$$z^\star = \frac{-x + 2z}{\sqrt{5}}. \tag{85}$$

A rendering of the resulting magnetic energy at time $t = 2$ is shown in Figure 30, i.e., after advecting the loop twice through the grid. Following Gardiner & Stone (2008), we impose a threshold in the projection at $B^2 = 10^{-7}$ to show the shape of the loop edge. Again, the structure of the loop is comparable with the rendering shown in Gardiner & Stone (2008).

4.16. Sedov–Taylor Blast Wave

This test models the self-similar evolution of a point-like thermal detonation (von Neumann 1942; Sedov 1946; Taylor 1950). Despite its unpleasant historical context, it also is a
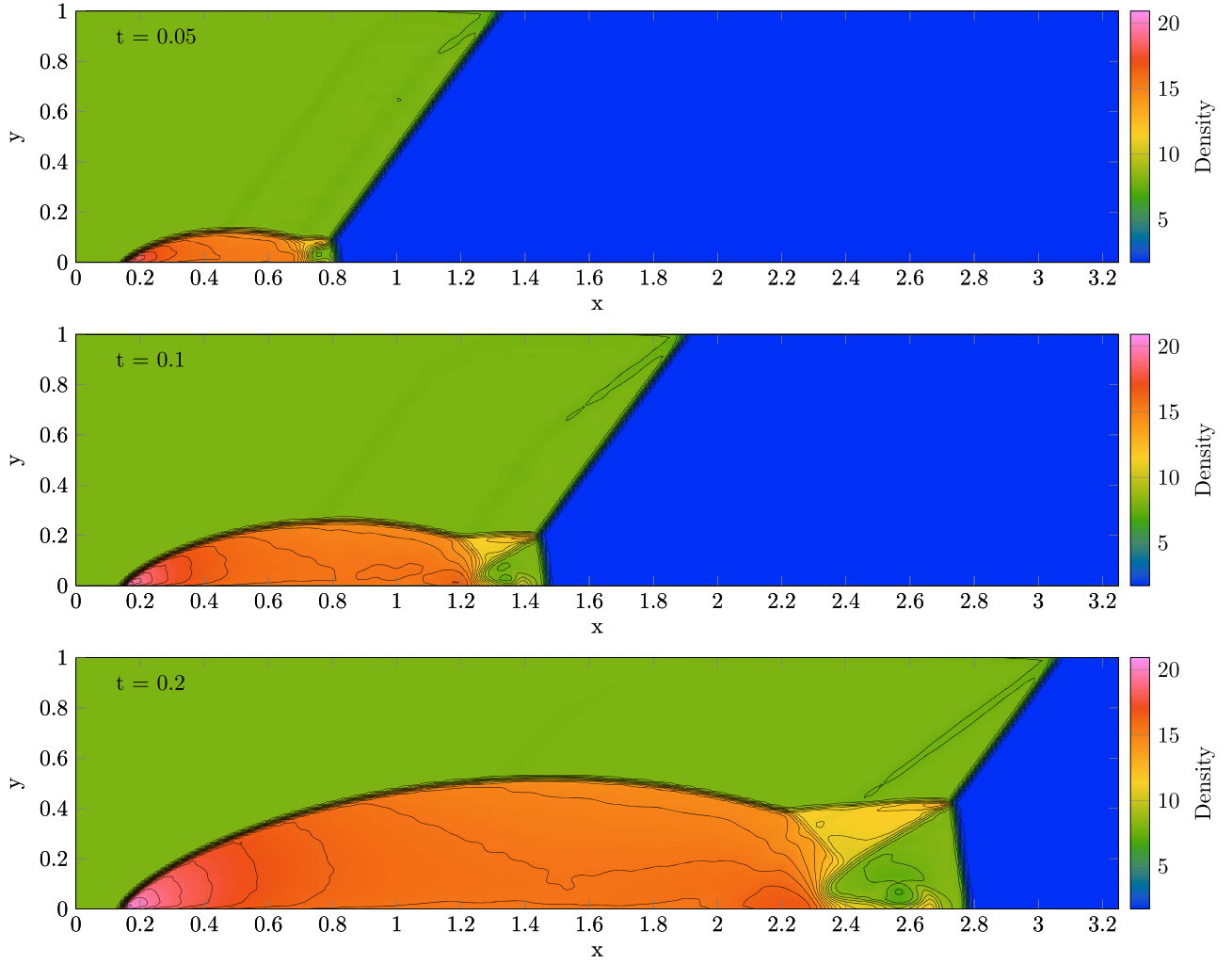
Figure 20. Density of the 2D double Mach reflection test with WENO5 at times 0.05, 0.1, and 0.2 (top to bottom) and resolution 260×80 zones. We overplot 30 contours between 1.73 and 20.92, following Woodward & Colella (1984).
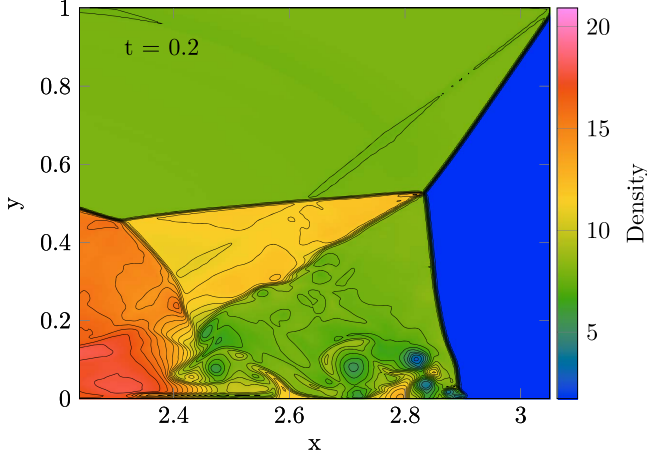

Figure 21. Density of the 2D double Mach reflection test with WENO5 at $t = 0.2$ and resolution 1024×256 zones. We overplot 30 contours between 1.73 and 20.92, following Woodward & Colella (1984).

model for early supernova explosions and is useful to expose the effects of the regular grid on the evolution of a strong spherical shock wave propagating into a thin medium with

negligible pressure. We set $\boldsymbol{U} = (1, 0, 0, 0, 10^{-8}, 0, 0, 10^{-5})$ everywhere in a three-dimensional computational domain with $L_x = L_y = L_z = 1$. We inject a unit energy $E_1 = 1$ in the center of the box and distribute it using a Gaussian with FWHM $\sigma = 0.5 \, dx$, where dx is the zone size. Our simulation uses $\gamma = 5/3$ and $N = 64^3$ zones. The self-similar analytic solution of the problem can be found in Landau & Lifshitz (1966), with $\beta = 1.15$.

In the left panel of Figure 31, we plot the radial profile of the shock at $t = 0.06$: the analytic solution in the red line and every 20th zone of the simulation by a black dot. On the right of Figure 31, we show a slice through the simulation at $z = 0$, $t = 0.06$ with colors on a log scale from 10^{-3} to 2 and contours at 0.01, 0.05, 0.1, 0.2, 0.5, and 2. The imprint of the grid on the explosion is clearly visible at radii below 0.3 and comparable to the DG3 method reported in Schaal et al. (2015), but better than the lower order WENO method presented in Feng et al. (2004), likely due to our higher order time integrator. Although these results seem disappointing at first, it is worth remembering that the DG3 scheme results used a 20 times smaller time step than ours and are an order of magnitude more expensive to compute.
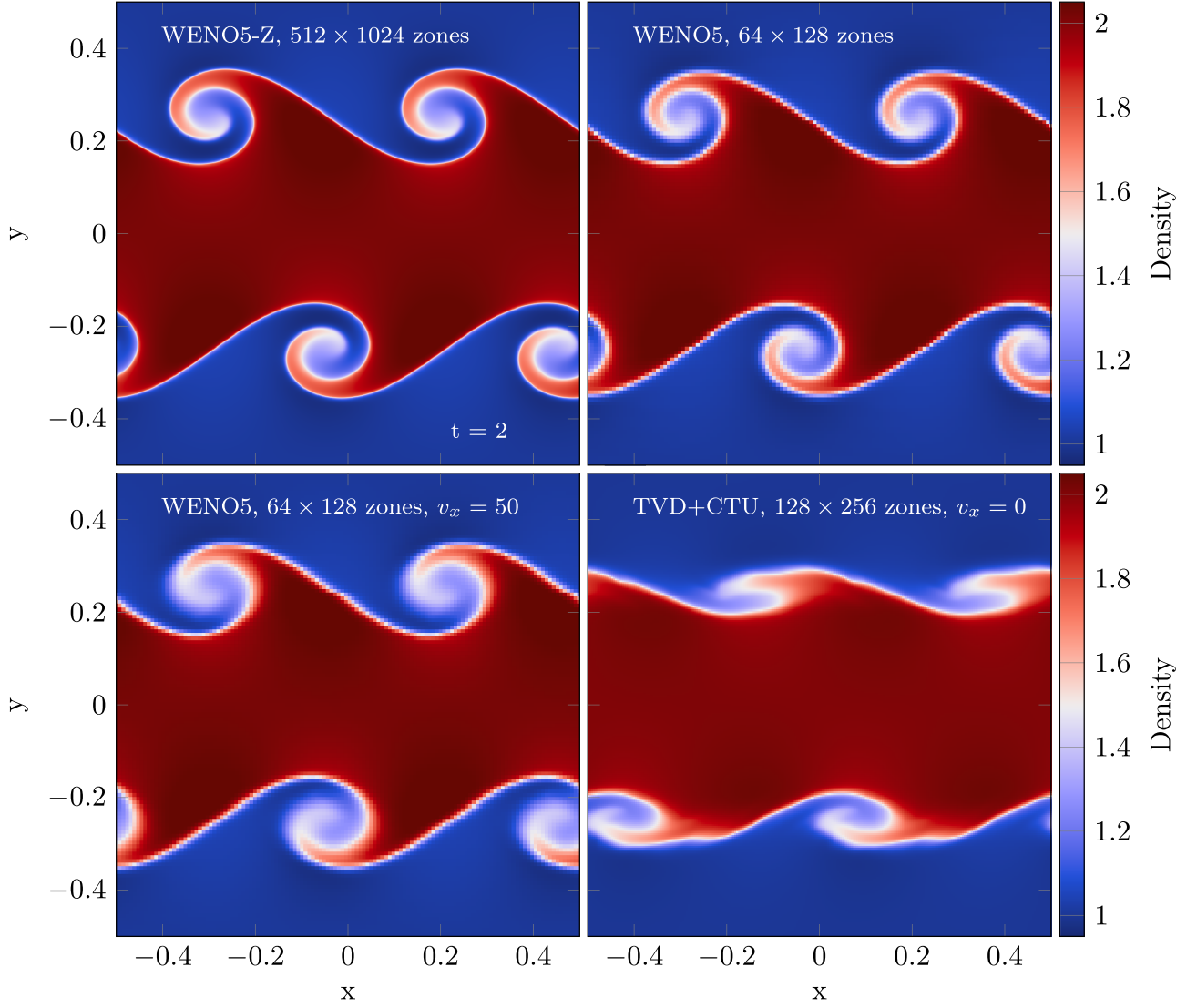
Figure 22. Density in Kelvin–Helmholtz simulations at $t = 2$. The color bar ranges from 0.95 to 2.05. Top left to bottom right: WENO5-Z with 512×1024 zones, WENO5 with 32×64 zones, WENO5 with 32×64 zones and a velocity boost of $v_x = 10$, and TVD+CTU with 128×256 zones and $v_x = 0$. We have duplicated the domain once along the x-direction to ease comparison with other publications.

In this test, Lagrangian schemes perform much better than our Eulerian scheme, due to the adaptive sampling (e.g., Springel 2010).

4.17. MHD Blast Wave

The magnetized blast wave simulation tests code performance in the low-β regime, i.e., when shocks are evolved in the presence of strong magnetic fields. Following Londrillo & Del Zanna (2000), we set $U = (1, 0, 0, 0, B_0/\sqrt{2}, B_0/\sqrt{2}, 0, P)^T$, with $P(r < r_c) = 1$, $P(r > r_c) = 100$, $B_0 = 10$, and $r_c = 0.125$. It follows that $\beta = 0.02$ in the medium ahead of the shock. The domain is $L_x \times L_y \times L_z = 1 \times 1.5 \times 1$. Slices through density, pressure, v^2, and B^2 from a WENO5 simulation at time $t = 0.2$ and $z = 0$ with resolution of $200 \times 300 \times 200$ zones are shown in Figure 32. The imprint of the magnetic field that is oriented along $(1, 1, 0)^T$ on the shock is clearly visible. We note that this test evaluates the robustness of the algorithm, and indeed, WENO5 required protection floors to complete the simulation.
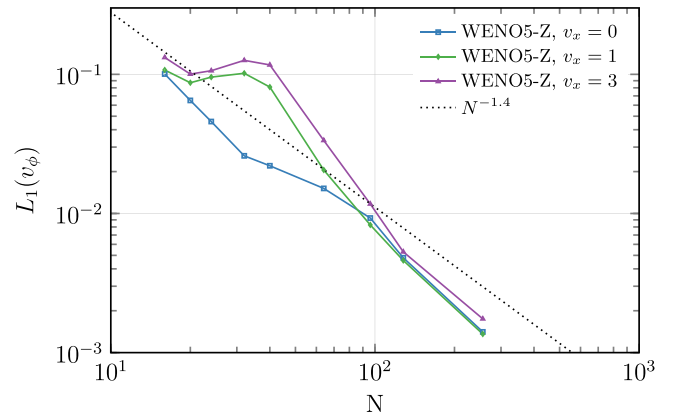
Figure 23. L_1 error norm of the angular velocity v_ϕ in the Gresho vortex test at $t = 3$ for three WENO5 simulations with $v_x = 0$ (blue), $v_x = 1$ (green), and $v_x = 3$ (purple). The dotted line corresponds to a scaling of $N^{-1.4}$ and is normalized to approximately correspond to the results shown in Springel (2010), their Figure 29.
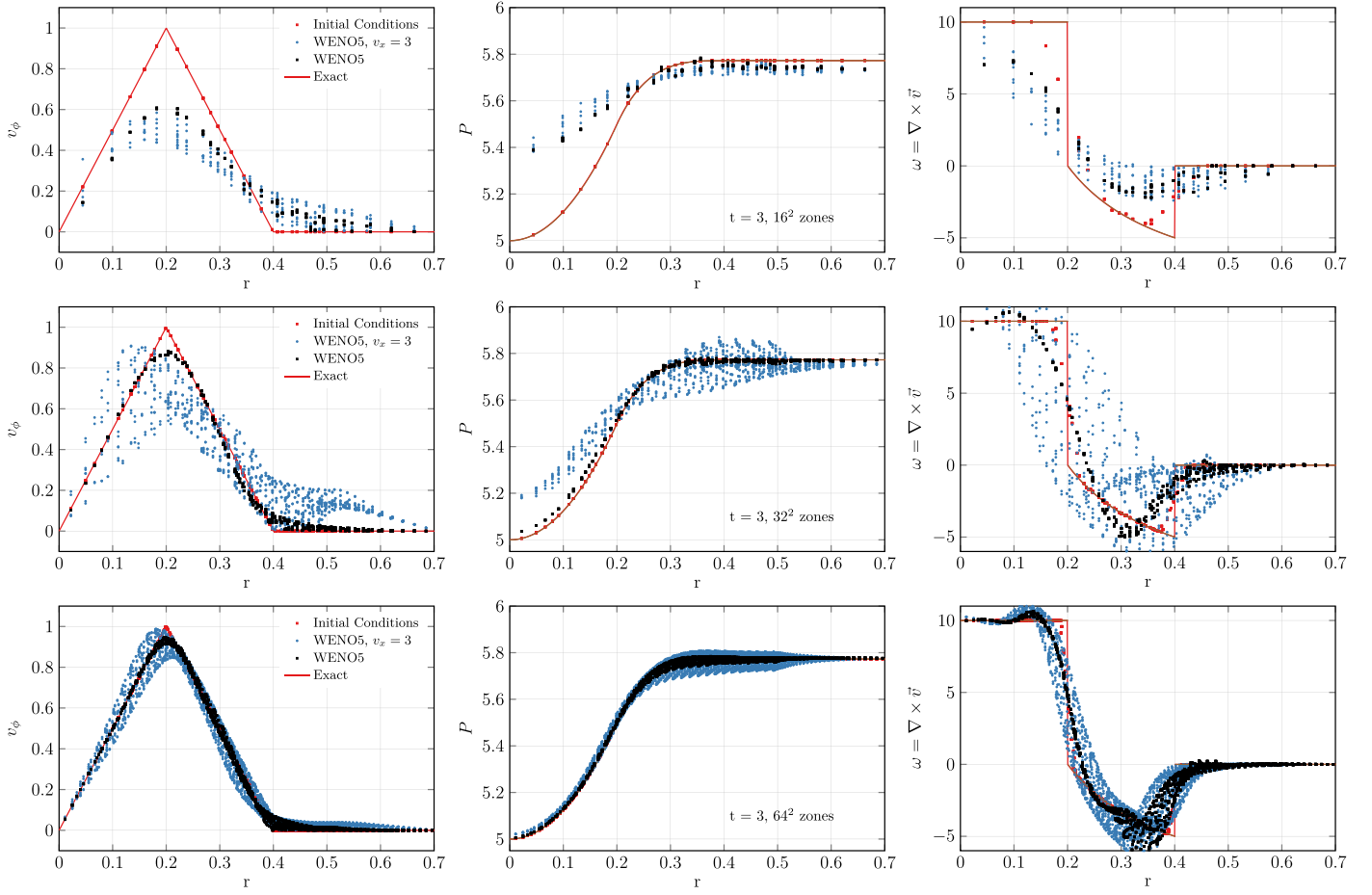
Figure 24. Left to right: angular velocity, pressure, and vorticity profile of the Gresho vortex simulations at (top to bottom) 16^2, 32^2, 64^2 zones. In each panel, we show the exact solution as the red line, the initial conditions as the red dots, and at $t = 3$ the WENO5 solution without boost as the black dots alongside the WENO5 solution with $v_x = 3$ as the blue dots.
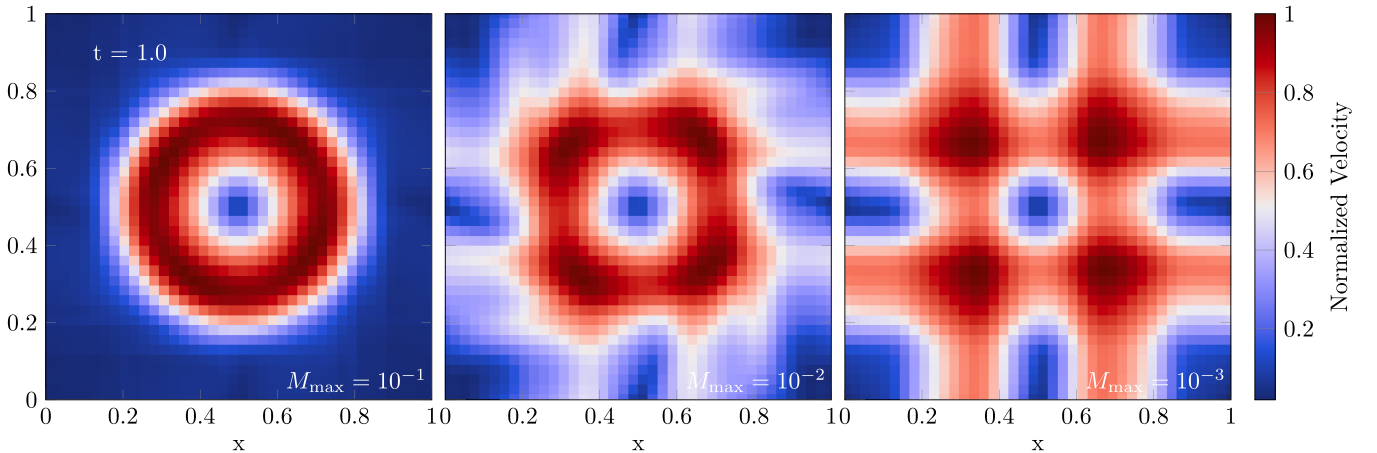

Figure 25. Normalized velocity of the Gresho vortex run with WENO5-Z at $t = 1$ with 40^2 zones. Left to right: $M_{max} = 0.1, 0.01, 0.001$.

We leave a more elaborate implementation of protection fluxes to future work.

5. Code Performance

5.1. Cache Blocking

All modern HPC systems feature a cache hierarchy to increase effective memory bandwidth to the CPUs (level 1, level 2, ...), with the fastest cache usually being the smallest due to silicon cost. To achieve good performance, it is desirable to divide the computational problem into subproblems that fit into the cache hierarchy, so that memory bandwidth is increased. This technique is called cache blocking. The optimal size will depend on the architecture of choice and the overhead in the algorithm, thus it has to be determined empirically. In WOMBAT, cache blocking is mitigated by tuning the patch size for a given architecture. Small patch sizes are desirable, because they enable more fine-grained load balancing.
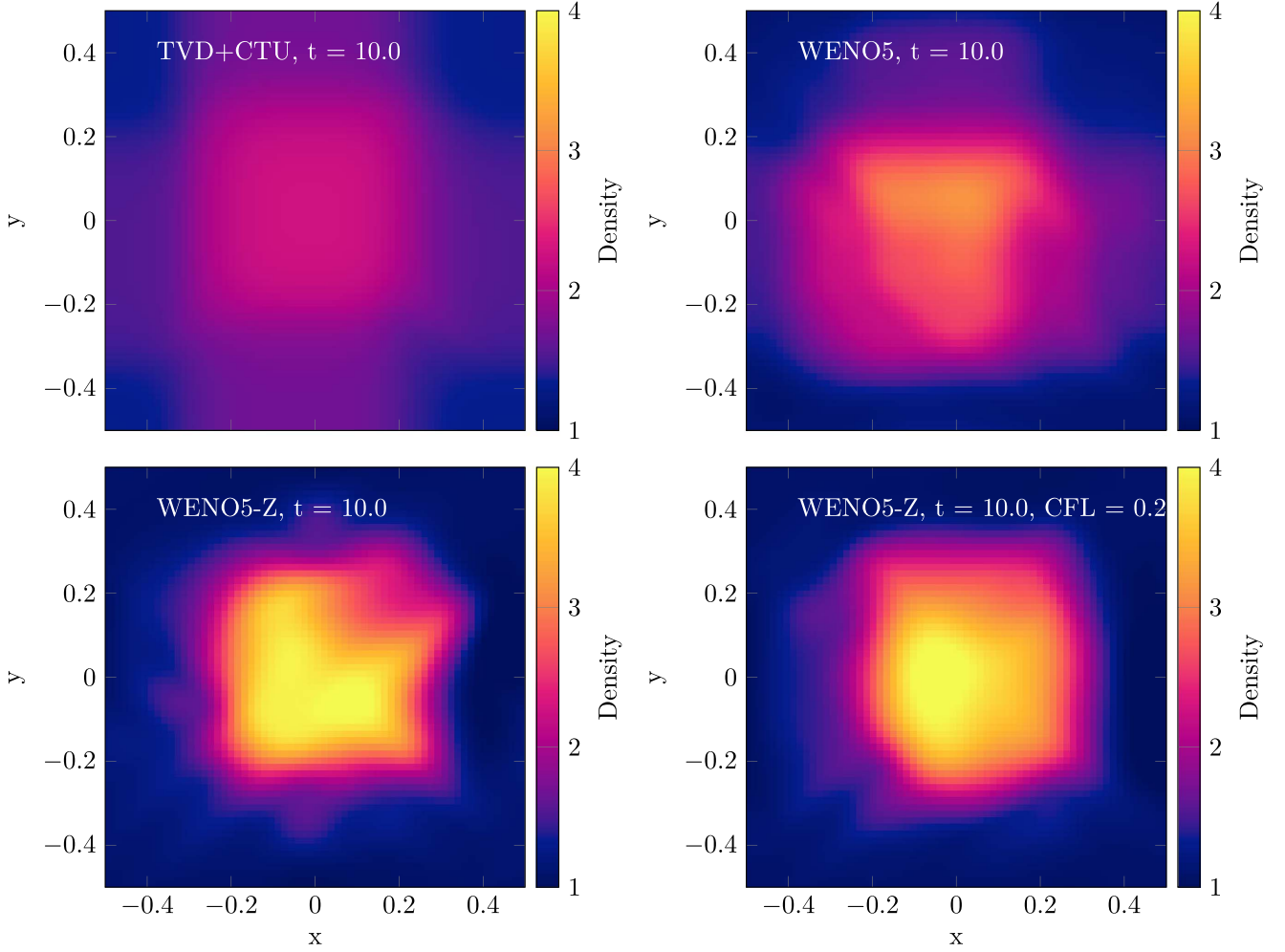
Figure 26. Density in the square advection test at $t = 10$ with 64^2 zones. Top left to bottom right: TVD+CTU, WENO5, WENO-Z, and WENO-Z with CFL $= 0.2$.
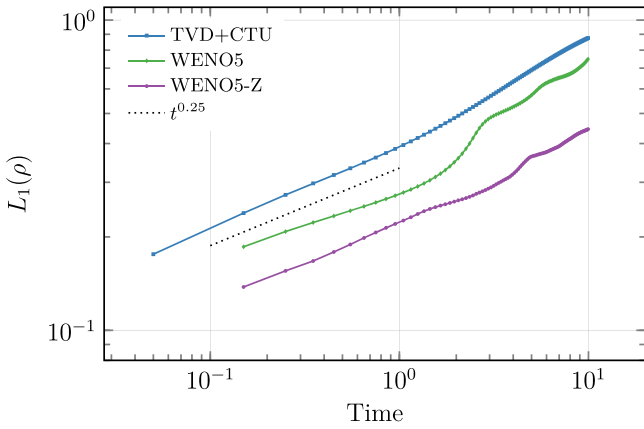
Figure 27. L_1 error norm over time in square advection test with 64^2 zones. We show TVD+CTU (blue), WENO5 (green), and WENO-Z (purple).

We run a patch size optimization study with 27 nodes to saturate MPI communication on the Aries interconnect. In Figure 33, we show the performance of the WENO5 algorithm on the Broadwell architecture with 2×18 cores per node in million zones per second per node as a function of number of zones per dimension per patch. The problem size is at least 512^3 zones, but varies by about 30% between runs, as the number of patches must be evenly divisible by the number of threads to not induce load imbalance in the threading. Colors correspond to runs with a varying number of

MPI processes per node—blue: 2 MPI ranks/12 OpenMP threads; green: 4 ranks/9 threads; purple: 12 ranks/3 threads; orange: 18 ranks/2 threads; yellow: 36 ranks/1 thread.

For all decompositions, the performance is maximized at 18^3 zone patches, then drops slightly and increases again toward 70^3 zone patches. This behavior differs from the TVD+CTU solver that peaks at 32^3 and shows a strong drop in performance for a larger patch size (see green dashed line from Mendygral et al. (2017). The flattening of the data arrays increases the vector length of all loops in the algorithm and shifts the optimal cache block to smaller patch sizes in the WENO5 solver. It also leads to effective use of the hardware prefetcher at large patch sizes, so the cache blocking is effectively done in hardware. This is not possible in the TVD +CTU solver that is not flattened. With the side constraint of small patch sizes, we conclude that WENO5 performs at 0.6 million zones per node with 18^3 zones per patch on Broadwell, with four MPI ranks per node. WENO5 is thus a factor of 7.5 slower than the TVD+CTU solver at the same resolution, but doubles the effective resolution. It follows that the WENO5 solver is more efficient than the second-order scheme, because increasing the resolution by a factor of 2 per dimension increases the runtime of the TVD+CTU solver by roughly a factor of 16: $2\times$ per dimension and roughly $2\times$ because the time step halves due to the factor 2 in Δx in Equation (17) (TVD+CTU uses a 1D criterion, but with CFL $= 0.4$ in 3D).
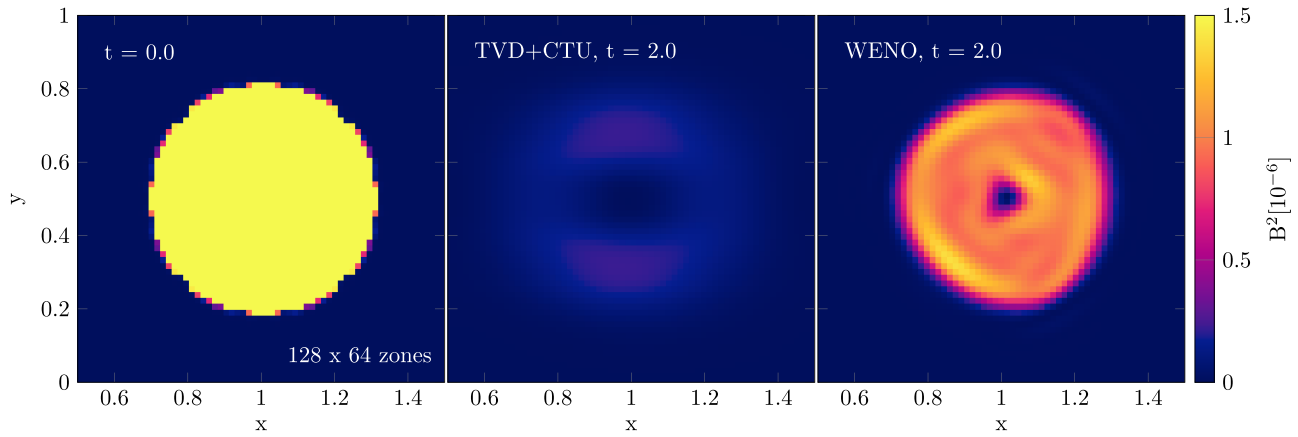
Figure 28. Magnetic field energy density B^2 at zone center in units of 10^{-6} of the magnetic field loop advection test in two dimensions with 128×64 zones. Left to right: initial conditions, TVD+CTU at time $t = 2$, and WENO5 at time $t = 2$.
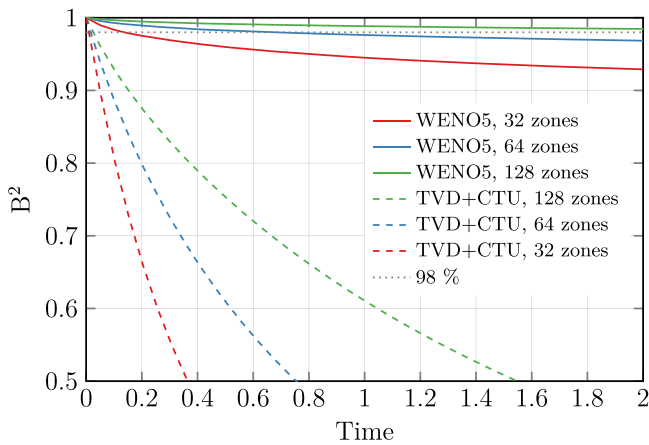

Figure 29. Total magnetic field energy density over time for the magnetic field loop test in two dimensions. We show WENO5 at three resolutions: 64×32 zones (red), 128×64 zones (blue), and 256×128 zones (green).

5.2. Vectorization

Our implementation uses vector-processing registers (SIMD) in modern CPUs extensively. Flattening the data arrays leads to large trip counts in the vector loops that are well cache-blocked. For every memory access, the virtual address has to be translated into a physical address. In the translation look-aside buffer (TLB), the memory management unit buffers page addresses for this translation from virtual to physical memory. On a standard system with 4 kb memory pages, our optimized vector loops can lead to a large miss rate in the TLB; the resulting page walk reduces performance by a factor of 2 (TLB thrashing). Thus, it is important to run `WOMBAT` with some form of huge pages enabled.

To demonstrate the performance gain from vectorization, we show the performance of the WENO5 solver on a 3D problem in GFlops on the Intel Broadwell architecture in Figure 34: in red, a single core without threads; in blue, a single core with whole node active (36 cores as 4 ranks with 9 threads). The performance of the run was measured by Cray Perftools.

In Figure 35, we compare the fractional number of instructions generated by the Cray compiler for the Intel Broadwell and Intel Skylake architecture, given a largest vector length using the `-vector0` and `-preferred-vector-length` compiler flags. The top panel shows the instruction
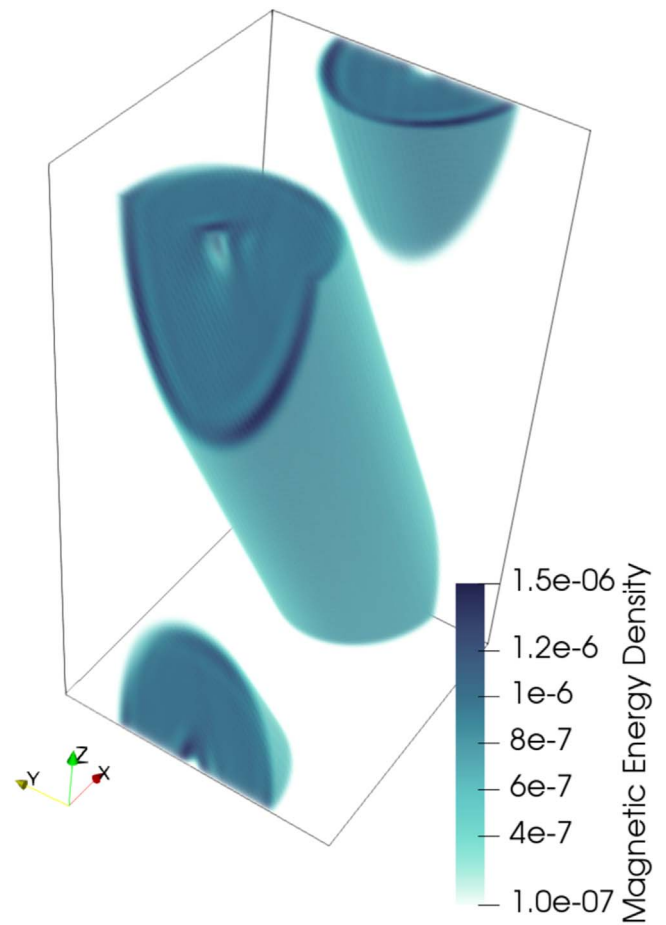

Figure 30. Magnetic field energy density B^2 of the magnetic field loop advection test in three dimensions with $128^2 \times 256$ zones at time $t = 2$. A threshold was imposed at 10^{-7}.

for the Intel Broadwell architecture. Clearly, the compiler vectorizes virtually all of the code at the highest vector length. The bottom panel shows the same graph, but for the Intel Skylake architecture. Here, the compiler only vectorizes the complete code at 128 bit vector length, but chooses a mix of vectorization and scalar instructions at broader vector lengths. In particular, at 512 bit vectors, scalar instructions make up half
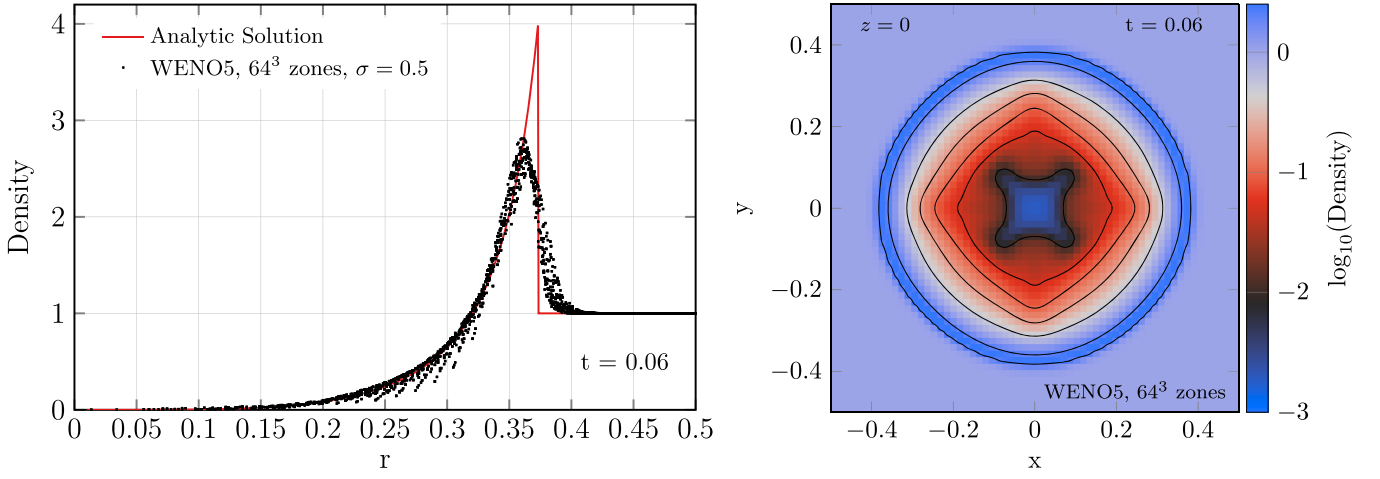
Figure 31. Left: radial density profile of the three-dimensional Sedov–Taylor test at $t = 0.06$, WENO5 simulation, shown as black dots; the analytic solution is shown by the red line. Plotted is only every 20th zone. Right: slice through the simulation at $z = 0$. We include contours at 0.01, 0.05, 0.1, 0.2, 0.5, and 2.
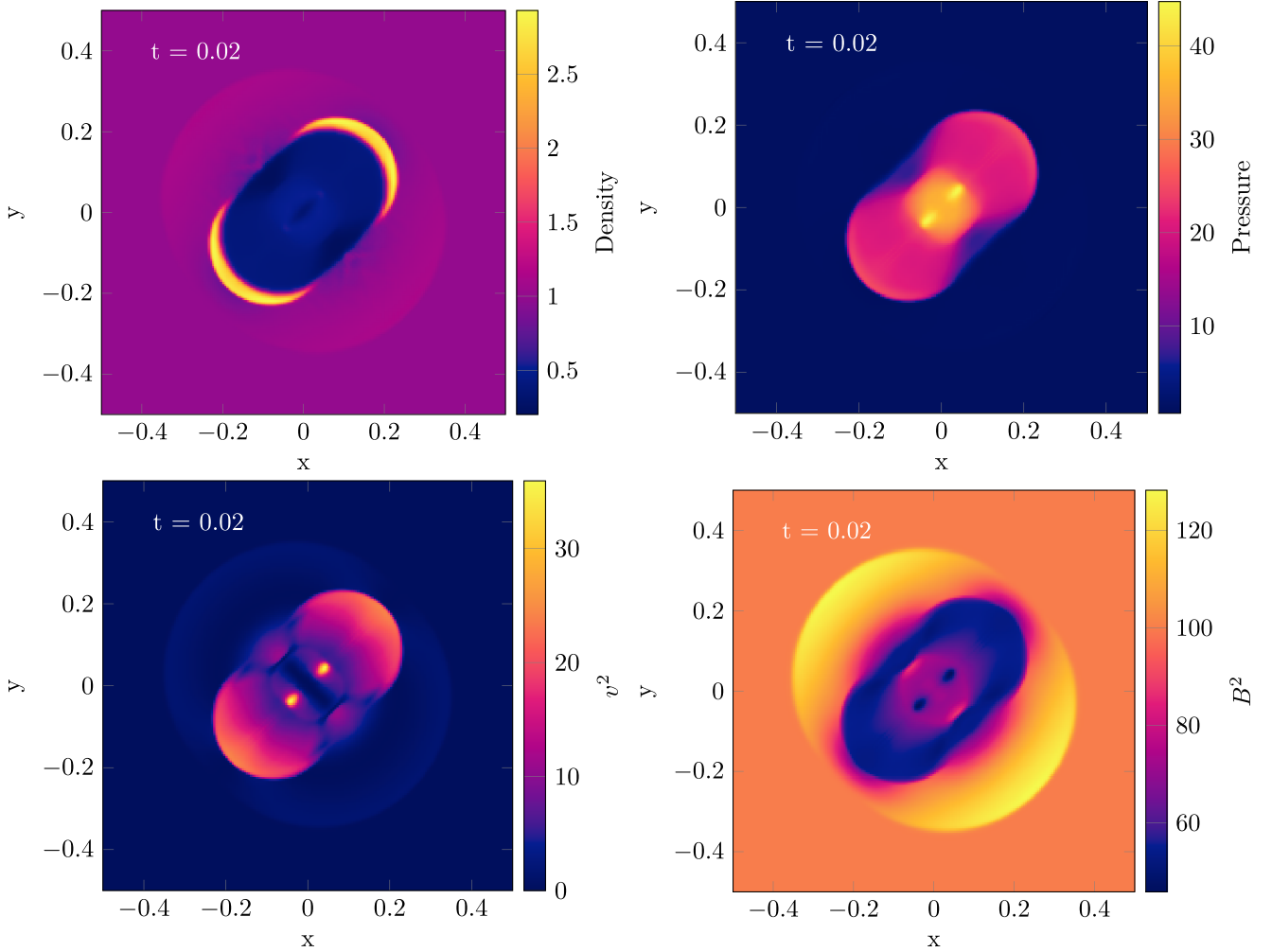

Figure 32. Slices through the MHD blast wave test at time $t = 0.02$. Shown is the center of the domain with $200 \times 300 \times 200$ zones along $z = 0$. Top left to bottom right: density, pressure, v^2, and B^2.

of the program. We note that the version with the widest vectors is the fastest for both architectures, reaching about 20% of double precision peak performance. Thus, Skylake is still faster than Broadwell by about 500 MFlops.

This shows that on some architectures, constraints other than vector length influence performance of the code. This is a good argument against using intrinsics or compiler pragmas to vectorize the whole program manually. Aside from the additional
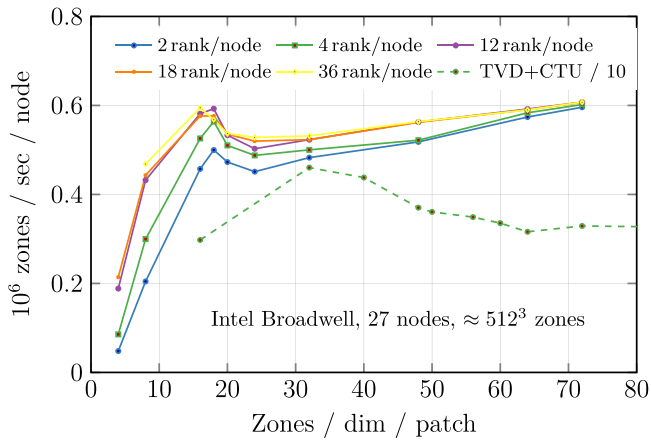
Figure 33. Compute performance in million zones per second per node over patch size (i.e., number of zones per dimension per patch) for 27 nodes on Intel Broadwell architecture. Problem size was about 512^3 zones. Blue: 2 ranks and 18 threads per node. Green: 4 ranks and 9 threads per node. Purple: 12 ranks and 3 threads per node. Orange: 18 rank and 2 threads per node. Yellow: 36 ranks and 1 thread per node.
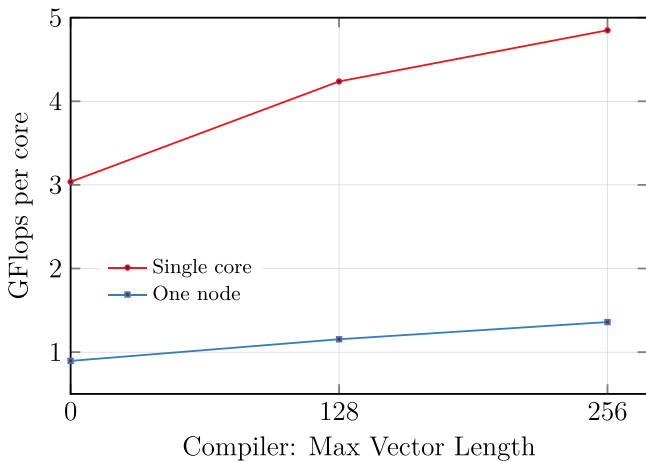

Figure 34. Performance per core in billion floating point operations per second over SIMD vector length of the WENO solver on a single Broadwell core (red) and a full Broadwell node (blue).

maintenance effort required to port the program to new architectures, the approach can actually increase execution time on architectures like Skylake. The better strategy is to expose instruction-level parallelism to the compiler and let the auto-vectorizer choose the instructions depending on its internal metrics. Given the increasing variety of HPC architectures competing for new exascale systems in the next years, the compiler remains the central tool for the programmer to achieve optimal performance on a given architecture.

5.3. Weak Scaling

We test the weak scaling of WOMBAT on the Cray XC40 "Hazel Hen" at HLRS Stuttgart. The machine features a Cray Aries interconnect, which uses a Dragonfly network with adaptive routers that are well suited for the high rate of small MPI messages in WOMBAT's communication pattern. Large pages of 16 MB are enabled by default. We note that the problem is well balanced, i.e., that the workload per node is the same on all ranks. Thus, WOMBAT's load-balancing capabilities are not tested here. For large problems, communication is only
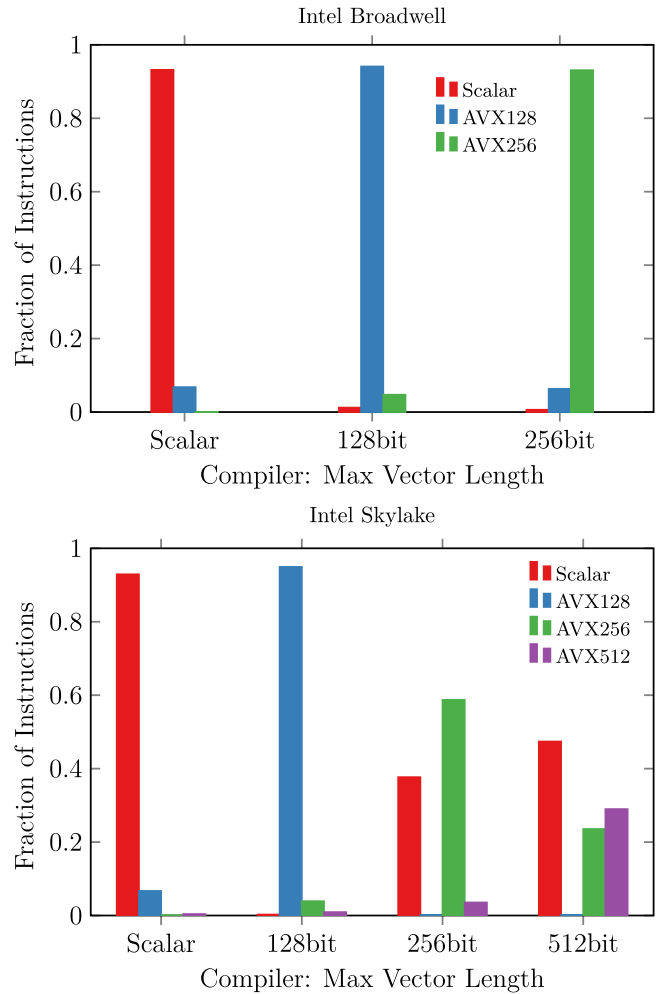

Figure 35. Fractional number of instructions generated by the Cray compiler over the limit of the vector length. Top for Intel Broadwell, bottom for Intel Skylake.

a small fraction of the total workload, which effectively hides communication inefficiencies. Hence, we choose a particularly small workload to clearly expose overhead from the MPI communication in the run. Unfortunately, this is not true for all weak scaling tests in the literature, and direct comparison with other codes is not always straightforward. We also note that for large-enough machines (millions of cores), communication overhead will eventually always be exposed. Thus, the argument that enough work is available per time step to hide communication inefficiencies is just another way of saying that an implementation does not scale beyond a certain point.

Here we use $3 \times 4 \times 4$ patches per MPI rank with 18^3 zones per rank, which results in a step time of about 2.5 s and a throughput of about 500,000 zones per second per node. We evolve the 3D problem for 100 steps, which means a total runtime of about 4 minutes. Every run was a separate submission on a different set of nodes on the system. The resulting mean step time over the number of nodes and cores is shown in Figure 36, from 1 node (24 cores) to 4096 nodes (98,304 cores). The minimum and maximum time per step is shown by the error bars. Ideal scaling at 2.45 s per update is shown by the dotted line. We also run the same test with the TVD+CTU solver, but with 32^3 patches, which is its optimal cache-blocked patch size on Broadwell. The difference stems from the flattened array implementation, which TVD does not
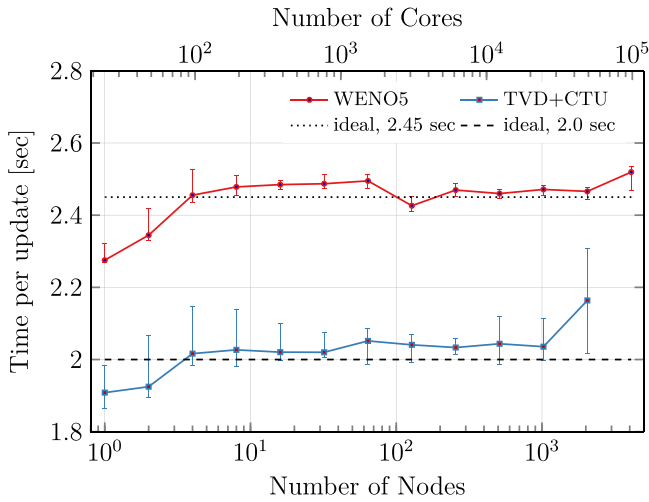
Figure 36. Time per update in seconds over the number of nodes and cores (weak scaling) on HLRS Cray XC40 "Hazel Hen." Red: WENO5 solver with 18^3 patches and $3 \times 4 \times 4$ patches per rank. Blue: TVD+CTU solver with same configuration but 32^3 patches.
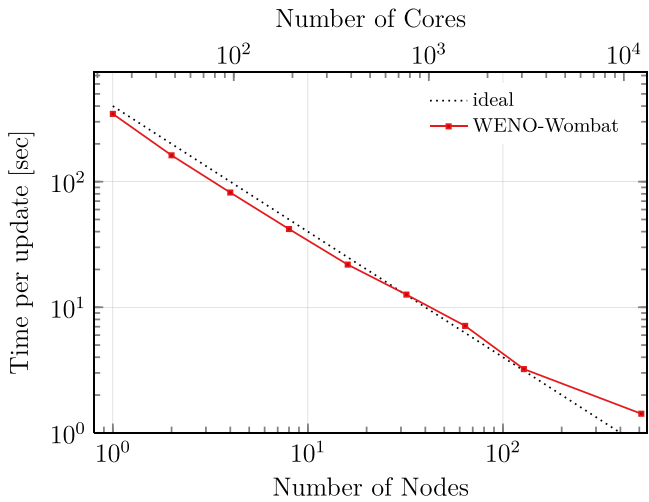

Figure 37. Strong scaling of WENO–Wombat on HLRS "Hazel Hen" as time per update over the number of nodes/cores.

share with WENO5. The resulting scaling is shown by the blue line. The step time is reduced by a factor of 1.2 relative to the WENO solver, and the problem is a factor of 5.6 larger, so the TVD+CTU solver is a factor of 6.7 faster at scale than the WENO solver. But the WENO solver doubles the effective resolution, which would increase the runtime of TVD+CTU by a factor of 16. TVD+CTU runs with CFL = 0.4, half the step size of WENO. Thus, the WENO solver is more efficient ("faster") than TVD+CTU by a factor of about $2 \times 2.3 = 4.6$ at scale.

We expect this difference in throughput (zones per second per code) to hold across architectures. Although TVD does not use flattened arrays, both implementations reach similar performance on the Broadwell architecture (14% versus 20% of peak), if properly cache-blocked. WOMBAT is written in vanilla Fortran 2008, which makes the code as portable as possible and gives access to the most mature compilers in the industry. Thus, the higher efficiency of WENO5 cannot be explained simply by implementation details or architecture and will, due its complete vectorization, very likely translate to

accelerators as well. The cache-blocking properties will change of course, but the program design allows adapting to this side constraint and achieving optimal performance independent of cache size and properties.

We note that the scaling is quite flat at about 2.45 s per update with about 10% scatter around the mean. Differences arise from network contention on the system, as runtime is dominated by the slowest node during the run. They are much smaller than seen previously on the Blue Waters system with the Gemini interconnect (Mendygral et al. 2017). This view is supported by the decrease in scatter in step times for the largest runs, where a significant fraction of the machine is used. The largest run uses more than half of the Hazel Hen system and performs at about 150 TFlops, or about 5% of the peak performance of the system. We leave it to the reader to compare with other community codes (e.g., Nordlund et al. 2018).

5.4. Strong Scaling

Strong scaling refers to the speed-up of an algorithm as the problem size is kept fixed and the compute power is increased. Ideally, execution time should half as the computing power doubles. According to Amdahl's law (Amdahl 1967), the execution time of any parallel algorithm will eventually be dominated by the nonparallelizable part of the work, which can be in the form of branching, communication, or load imbalance.

We run a strong scaling test on the Cray XC40 "Hazel Hen" at HLRS Stuttgart 4 MPI ranks and 12 OpenMP threads per node. We use 18^3 patches with an initial patch distribution of $96 \times 12 \times 12$ patches, so the world grid has 1728×216^2 zones. For the final point at 512 nodes, we reduced the patch size to 9×18^2 zones.

The result is shown in Figure 37 as seconds per time step over the number of nodes and cores. WENO–Wombat follows the ideal scaling closely down to 128 nodes. The point at 512 nodes deviates from the ideal scaling, due to the decreased patch size. There was simply not enough work available anymore with 512 nodes, which limits WOMBAT's ability to further strong scale.

6. Conclusions

High-order Eulerian schemes will likely become instrumental to simulations of turbulence and magnetic field amplification in the astrophysical context (Balsara 2017; Felker & Stone 2018; Guillet et al. 2019). The inevitable growth of computing power has led to ample resources available for highly optimized numerical implementations that scale to $>10^5$ cores. With the advent of accelerators and high-bandwidth memory on the horizon, complex high-order simulations with rich subgrid physics are within reach for the community.

We have presented an implementation of a fifth-order WENO scheme in the numerical WOMBAT framework. We combined the classical FD scheme with a simple CT scheme for the evolution of magnetic fields. We have demonstrated algorithmic correctness and fidelity on a variety of test problems in one, two, and three dimensions. We argued that the CT-WENO5 scheme:

1. Doubles the effective resolution compared to common second-order schemes like CTU+CT or TVD+CTU.
2. Resolves instabilities better than a lower order scheme, due to its very low numerical diffusivity.

3. Is less affected by advection relative to the grid. It is competitive with Lagrangian methods in tests with moderate advection velocities.
4. Is more computationally efficient than a lower order scheme in three dimensions at similar solution quality.
5. Needs to use double precision floating point arithmetic to deliver good results. The low diffusivity of the scheme exposes noise very quickly.

The limitations of our CT-WENO5 implementation are in the dispersion of MHD waves, where, due to the simple CT scheme, the implementation roughly matches a good second-order scheme. Nonetheless, we have shown that MHD wave dissipation by numerical diffusion is accurate to fifth order. Furthermore, the CT-WENO5 scheme does not handle advection and angular momentum conservation as well as discontinuous Galerkin schemes of third or higher order. These limitations are compromises that keep the implementation computationally cheap. For example, WENO5 is significantly cheaper than third-order discontinuous Galerkin implementations, due to the about 20 times larger time step in three dimensions.

We showed that our implementation reaches about 20% of peak double precision performance on a single Broadwell or Skylake core and ∼5% on 98,000 cores on a Cray XC40. On Intel Broadwell processors, we observe a throughput of about 0.5 million zones per node at twice the solution fidelity of a typical second-order scheme with a Roe solver. We conclude that our implementation represents a favorable compromise of fidelity, robustness, and computational efficiency, awaiting astrophysical applications.

6.1. Outlook

WENO-WOMBAT already includes a treatment of cold supersonic flows using an elegant entropy scheme based on flux splitting (H. Jang et al. 2019, in preparation). A few improvements to the implementation presented here come to mind. A high-order CT scheme that matches the accuracy of the spatial interpolation would reduce magnetic field dispersion, albeit at considerable computational cost. High-order CT schemes exist for FV approaches (e.g., Londrillo & del Zanna 2004; Felker & Stone 2018; Verma et al. 2019), but need to be adapted to our FD algorithm. Global smoothness indicators could be used to improve the robustness of the scheme (e.g., Balbás & Tadmor 2006). Worth another look is likely a low-storage fourth-order Runge–Kutta integrator. A wide variety of strong-stability-preserving schemes exist, usually with five of more stages and CFL numbers >1 (Spiteri & Ruuth 2002). Furthermore, a ninth-order WENO implementation might become affordable once supercomputers reach exaflop performance, further doubling the effective resolution of the simulation and reducing storage and memory requirements. An extension of our WENO5 implementation would be straightforward. On the technical side, the flattened array implementation will make it trivial to port the WENO solver to accelerators. The OpenMP 4 standard would be the natural choice, providing a portable approach for a wide variety of accelerator technologies.

6.2. Reproducibility

It has been shown that computational fluid dynamics without code-level transparency is not reproducible, even by the authors of the original publication: Mesnard & Barba (2017) found that open source code, logs, and data are a bare minimum to reproduce results years later. WOMBAT follows their reproducibility policy: sources and logs used to produce this document are available at https://wombatcode.org/publications/. Most data are made available, but have to adhere to certain space limitations. This data file also contains the complete source code tree of the WOMBAT version used in this paper, for reference purposes only. We did not yet include WENO5 in the public version of WOMBAT as we do not yet consider the scheme robust enough for applications.

Appendix
MHD Eigenvectors

The eigenvectors decouple the system of partial differential equations into scalar advection equations. Every component m of the decoupled system corresponds to an eigenvalue λ_m with a left eigenvector $L_i(m)$ and right eigenvector $R^i(m)$ with $L_i R^i = 1$ (Jeffrey & Taniuti 1964). The eigenvalues by component m are $\lambda(1, 7) = v_x \mp c_{\text{fast}}$, $\lambda(2, 6) = v_x \mp c_A$, $\lambda(3, 5) = v_x \mp c_{\text{slow}}$, and $\lambda(4) = v_x$. For MHD, degeneracies occur in the case of vanishing magnetic field components (Brio & Wu 1988b), so we set

$$(\beta_y, \beta_z) = \begin{cases} \frac{(B_y, B_z)}{\sqrt{B_y^2 + B_z^2}} & \text{if } B_y^2 + B_z^2 > \epsilon_B \\ \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right) & \text{otherwise} \end{cases} \quad (86)$$

$$\text{sgn}(B_x) = \begin{cases} 1 & \text{if } B_x \geqslant \epsilon_B \\ -1 & \text{otherwise} \end{cases} \quad (87)$$

$$(\alpha_f, \alpha_s) = \begin{cases} \frac{\sqrt{c_s^2 - c_{\text{slow}}^2}, \sqrt{c_{\text{fast}}^2 - c_s^2}}{\sqrt{c_{\text{fast}}^2 - c_{\text{slow}}^2}} & \text{if } \sqrt{c_{\text{fast}}^2 - c_{\text{slow}}^2} \geqslant \epsilon_B \\ (1, 1) & \text{otherwise} \end{cases} \quad (88)$$

where $\epsilon_B = 10^{-30}$. We note that in some cases, numerical noise from compiler optimizations can break the degeneracy of the MHD eigenvalues with the sound speed (but not between fast and slow modes) in the calculation of Equation (88). This can lead to noise in the calculation that can be exposed, e.g., in the implosion test, Section 4.8. It is advisable to explicitly check for the eigenvalue degeneracy in the code and set Equation (88) explicitly to zero. For similar reasons, we found it practical to explicitly store the sound speed and not its square in the eigenvector calculations.

Our eigenvectors follow Jiang & Wu (1999) and H. Jang et al. (2019, in preparation) with

$$\gamma_1 = \frac{\gamma - 1}{2}, \quad \gamma_2 = \frac{\gamma - 2}{\gamma - 1}, \quad \tau = \frac{\gamma - 1}{c_s^2} \quad (89)$$

and

$$\Gamma_f = \alpha_f c_{\text{fast}} v_x - \alpha_s c_{\text{slow}} \text{sgn}(B_x)(\beta_y v_y + \beta_z v_z) \quad (90)$$

$$\Gamma_a = \text{sgn}(B_x)(\beta_z v_y - \beta_y v_z) \quad (91)$$

$$\Gamma_s = \alpha_s c_{\text{slow}} v_x + \alpha_f c_{\text{fast}} \text{sgn}(B_x)(\beta_y v_y + \beta_z v_z), \quad (92)$$

The eigenvectors for the fast mode ($m = 1$, $m = 7$) are

$$L_1(m) = \frac{1}{2c_s^2}[\alpha_f(\gamma_1 v^2 \pm \Gamma_f)], \quad R^1(m) = \alpha_f \quad (93)$$

$$L_2(m) = (1 - \gamma)\alpha_f v_x \mp \alpha_f v_y \pm c_{\text{slow}}\alpha_s \beta_y \text{sgn}(B_x),$$
$$R^2(m) = \alpha_f(v_x \mp c_{\text{fast}}) \quad (94)$$

$$L_3(m) = (1 - \gamma)\alpha_f v_y \pm c_{\text{slow}}\alpha_s \beta_y \text{sgn}(B_x),$$
$$R^3(m) = \alpha_f v_y \pm c_{\text{slow}}\alpha_s \beta_y \text{sgn}(B_x) \quad (95)$$

$$L_4(m) = (1 - \gamma)\alpha_f v_z \pm c_{\text{slow}}\alpha_s \beta_z \text{sgn}(B_x),$$
$$R^4(m) = \alpha_f v_z \pm c_{\text{slow}}\alpha_s \beta_z \text{sgn}(B_x) \quad (96)$$

$$L_5(m) = (1 - \gamma)\alpha_f B_y - \sqrt{\rho} c_s \alpha_s \beta_y,$$
$$R^5(m) = c_s \alpha_s \beta_y / \sqrt{\rho} \quad (97)$$

$$L_6(m) = (1 - \gamma)\alpha_f B_z - \sqrt{\rho} c_s \alpha_s \beta_z,$$
$$R^6(m) = c_s \alpha_s \beta_z / \sqrt{\rho} \quad (98)$$

$$L_7(m) = (\gamma - 1)\alpha_f,$$
$$R^7(m) = \alpha_f\left(\frac{1}{2}v^2 + c_{\text{fast}}^2 - \gamma_2 c_s^2\right) \mp \Gamma_f. \quad (99)$$

The eigenvectors for the slow mode ($m = 2$, $m = 6$) are

$$L_1(m) = \frac{1}{2}\Gamma_a, \quad R^1(m) = 0 \quad (100)$$

$$L_2(m) = 0, \quad R^2(m) = 0 \quad (101)$$

$$L_3(m) = -\frac{1}{2}\beta_z \text{sgn}(B_x), \quad R^3(m) = -\beta_z \text{sgn}(B_x) \quad (102)$$

$$L_4(m) = \frac{1}{2}\beta_y \text{sgn}(B_x), \quad R^4(m) = \beta_y \text{sgn}(B_x) \quad (103)$$

$$L_5(m) = \mp\frac{1}{2}\sqrt{\rho}\beta_z, \quad R^5(m) = \mp\beta_z / \sqrt{\rho} \quad (104)$$

$$L_6(m) = \pm\frac{1}{2}\sqrt{\rho}\beta_y, \quad R^6(m) = \pm\beta_y / \sqrt{\rho} \quad (105)$$

$$L_7(m) = 0, \quad R^7(m) = -\Gamma_a. \quad (106)$$

The eigenvectors for the Alfvén mode ($m = 3$, $m = 5$) are

$$L_1(m) = \frac{1}{2c_s^2}(\gamma_1 \alpha_s v^2 \pm \Gamma_s), \quad R^1(m) = \alpha_s \quad (107)$$

$$L_2(m) = \frac{1}{2c_s^2}((1 - \gamma)\alpha_s v_x \mp \alpha_s c_{\text{slow}}),$$
$$R^2(m) = \alpha_s(v_x \mp c_{\text{slow}}) \quad (108)$$

$$L_3(m) = \frac{1}{2c_s^2}((1 - \gamma)\alpha_s v_y \mp c_{\text{fast}}\alpha_f \beta_y \text{sgn}(B_x)),$$
$$R^3(m) = \alpha_s v_y \mp c_{\text{fast}}\alpha_f \beta_y \text{sgn}(B_x) \quad (109)$$

$$L_4(m) = \frac{1}{2c_s^2}((1 - \gamma)\alpha_s v_z \mp c_{\text{fast}}\alpha_f \beta_z \text{sgn}(B_x)),$$
$$R^4(m) = \alpha_s v_z \mp c_{\text{fast}}\alpha_f \beta_z \text{sgn}(B_x) \quad (110)$$

$$L_5(m) = \frac{1}{2c_s^2}((1 - \gamma)\alpha_s B_y - \sqrt{\rho} c_s \alpha_f \beta_y),$$
$$R^5(m) = -c_s \alpha_f \beta_y / \sqrt{\rho} \quad (111)$$

$$L_6(m) = \frac{1}{2c_s^2}((1 - \gamma)\alpha_s B_z - \sqrt{\rho} c_s \alpha_f \beta_z),$$
$$R^6(m) = -c_s \alpha_f \beta_z / \sqrt{\rho} \quad (112)$$

$$L_7(m) = \frac{1}{2c_s^2}((\gamma - 1)\alpha_s),$$
$$R^7(m) = \alpha_s\left(\frac{1}{2}v^2 + c_{\text{slow}}^2 - \gamma_2 c_s^2\right) \mp \Gamma_s. \quad (113)$$

The eigenvectors for the entropy mode ($m = 4$) are

$$L_1(m) = 1 - \frac{1}{2}\tau v^2, \quad R^1(m) = 1 \quad (114)$$

$$L_2(m) = \tau v_x, \quad R^2(m) = v_x \quad (115)$$

$$L_3(m) = \tau v_y, \quad R^3(m) = v_y \quad (116)$$

$$L_4(m) = \tau v_z, \quad R^4(m) = v_z \quad (117)$$

$$L_5(m) = \tau B_y, \quad R^5(m) = 0 \quad (118)$$

$$L_6(m) = \tau B_z, \quad R^6(m) = 0 \quad (119)$$

$$L_7(m) = -\tau, \quad R^7(m) = \frac{1}{2}. \quad (120)$$

These components are evaluated at the boundary using simple arithmetic averaging.

ORCID iDs

J. M. F. Donnert ⓘ https://orcid.org/0000-0001-9627-2980
P. Mendygral ⓘ https://orcid.org/0000-0002-1638-8930
D. Ryu ⓘ https://orcid.org/0000-0002-5455-2957

References

Almgren, A. S., Bell, J. B., Lijewski, M. J., Lukić, Z., & Van Andel, E. 2013, ApJ, 765, 39
Amdahl, G. M. 1967, in Proc. Spring Joint Computer Conf., AFIPS '67 (Spring) (New York: ACM), 483, https://dl.acm.org/citation.cfm?doid=1465482.1465560

Babuška, I., & Zlámal, M. 1973, SJNA, 10, 863
Balbás, J., & Tadmor, E. 2006, SIAM J. Sci. Comput., 28, 533
Balsara, D. S. 2017, LRCA, 3, 2
Balsara, D. S., Garain, S., & Shu, C.-W. 2016, JCoPh, 326, 780
Balsara, D. S., Rumpf, T., Dumbser, M., & Munz, C.-D. 2009, JCoPh, 228, 2480
Balsara, D. S., & Shu, C.-W. 2000, JCoPh, 160, 405
Balsara, D. S., & Spicer, D. D. 1999a, JCoPh, 149, 270
Balsara, D. S., & Spicer, D. D. 1999b, JCoPh, 149, 270
Beck, A., Murante, G., Arth, A., et al. 2016, MNRAS, 455, 2110
Borges, R., Carmona, M., Costa, B., & Don, W. S. 2008, JCoPh, 227, 3191
Brio, M., & Wu, C. C. 1988a, JCoPh, 75, 500
Brio, M., & Wu, C. C. 1988b, JCoPh, 75, 400
Bryan, G. L., Norman, M. L., O'Shea, B. W., et al. 2014, ApJS, 211, 19
Colella, P., & Woodward, P. R. 1984, JCoPh, 54, 174
Dai, W., & Woodward, P. R. 1998, ApJ, 494, 317
De Pontieu, B., McIntosh, S. W., Carlsson, M., et al. 2007, Sci, 318, 1574
Del Zanna, L., Velli, M., & Londrillo, P. 2001, A&A, 367, 705
Donnert, J., Jang, H., Mendygral, P., et al. 2018, Galax, 6, 104
Einfeldt, B. 1988, SJNA, 25, 294
Einfeldt, B., Roe, P. L., Munz, C. D., & Sjogreen, B. 1991, JCoPh, 92, 273
Evans, C. R., & Hawley, J. F. 1988, ApJ, 332, 659
Felker, K. G., & Stone, J. M. 2018, JCoPh, 375, 1365
Feng, L.-L., Shu, C.-W., & Zhang, M. 2004, ApJ, 612, 1
Frank, A., Jones, T. W., Ryu, D., & Gaalaas, J. B. 1996, ApJ, 460, 777
Fryxell, B., Olson, K., Ricker, P., et al. 2000, ApJS, 131, 273
Gardiner, T. A., & Stone, J. M. 2005, JCoPh, 205, 509
Gardiner, T. A., & Stone, J. M. 2008, JCoPh, 227, 4123
Godunov, S. K. 1959, Matematicheskii Sbornik, 47, 271
Goldstein, M. L. 1978, ApJ, 219, 700
Gottlieb, S., Ketcheson, D. I., & Shu, C.-W. 2009, JSCom, 38, 251
Gottlieb, S., Shu, C., & Tadmor, E. 2001, SIAMR, 43, 89
Greenough, J. A., & Rider, W. J. 2004, JCoPh, 196, 259
Gresho, P. M., & Chan, S. T. 1990, IJNMF, 11, 621
Guillet, T., Pakmor, R., Springel, V., Chandrashekar, P., & Klingenberg, C. 2019, MNRAS, in press
Guo, L.-d., & Zhang, L. 2004, J. Visual., 7, 225
Harten, A. 1983, JCoPh, 49, 357
Harten, A., Engquist, B., Osher, S., & Chakravarthy, S. R. 1987, JCoPh, 71, 231
Helzel, C., Rossmanith, J. A., & Taetz, B. 2011, JCoPh, 230, 3803
Henrick, A. K., Aslam, T. D., & Powers, J. M. 2005, JCoPh, 207, 542
Hooke, R. 1665, Micrographia, Observation LVIII, 217
Hopkins, P. F. 2013, MNRAS, 428, 2840
Hopkins, P. F. 2015, MNRAS, 450, 53
Hopkins, P. F., & Raives, M. J. 2016, MNRAS, 455, 51
Jeffrey, A., & Taniuti, T. 1964, Non-linear Wave Propagation (New York: Academic Press)
Jiang, G.-S., & Shu, C.-W. 1996, JCoPh, 126, 202
Jiang, G.-S., & Wu, C.-c. 1999, JCoPh, 150, 561
Kovesi, P. 2015, arXiv:1509.03700
Kravtsov, A. V., Klypin, A. A., & Khokhlov, A. M. 1997, ApJS, 111, 73
Kulsrud, R. M., & Ostriker, E. C. 2006, PhT, 59, 010000
Landau, L. D., & Lifshitz, E. M. 1966, Hydrodynamik (Berlin: Akademie-Verlag)
Lax, P. D. 1954, Commun. Pure Appl. Math., 7, 159
Lecoanet, D., McCourt, M., Quataert, E., et al. 2016, MNRAS, 455, 4274
Lee, D. 2013, JCoPh, 243, 269
LeVeque, R. 2002, Finite Volume Methods for Hyperbolic Problems (Cambridge: Cambridge Univ. Press)
Liska, R., & Wendroff, B. 2003, SIAM J. Sci. Comput., 25, 995
Liu, X.-D., Osher, S., & Chan, T. 1994, JCoPh, 115, 200
Londrillo, P., & Del Zanna, L. 2000, ApJ, 530, 508
Londrillo, P., & del Zanna, L. 2004, JCoPh, 195, 17
McCorquodale, P., & Colella, P. 2011, Commun. Appl. Math. Comput. Sci., 6, 1

McNally, C. P., Lyra, W., & Passy, J.-C. 2012, ApJS, 201, 18
Mendygral, P. J., Radcliffe, N., Kandalla, K., et al. 2017, ApJS, 228, 23
Mesnard, O., & Barba, L. A. 2017, CSE, 19, 44
Miczek, F., Röpke, F. K., & Edelmann, P. V. F. 2015, A&A, 576, A50
Mignone, A., Bodo, G., Massaglia, S., et al. 2007, ApJS, 170, 228
Mignone, A., Tzeferacos, P., & Bodo, G. 2010, JCoPh, 229, 5896
Miura, A., & Pritchett, P. L. 1982, JGR, 87, 7431
Mocz, P., Vogelsberger, M., Sijacki, D., Pakmor, R., & Hernquist, L. 2014, MNRAS, 437, 397
Muñoz, D. J., Springel, V., Marcus, R., Vogelsberger, M., & Hernquist, L. 2013, MNRAS, 428, 254
Nordlund, Å., Ramsey, J. P., Popovas, A., & Küffmeier, M. 2018, MNRAS, 477, 624
Orzang, S. A., & Tang, C. M. 1979, JFM, 90, 128
Parker, E. N. 1958, ApJ, 128, 664
Richardson, L. F. 1922, QJRMS, 48, 282
Robertson, B. E., Kravtsov, A. V., Gnedin, N. Y., Abel, T., & Rudd, D. H. 2010, MNRAS, 401, 2463
Roe, P. 1997, JCoPh, 135, 250
Roe, P. L. 1981, JCoPh, 43, 357
Ruderman, M. S., & Simpson, D. 2005, SSRv, 121, 287
Ryu, D., & Jones, T. W. 1995, ApJ, 442, 228
Ryu, D., Miniati, F., Jones, T. W., & Frank, A. 1998, ApJ, 509, 244
Ryu, D., Ostriker, J. P., Kang, H., & Cen, R. 1993, ApJ, 414, 1
Sarmin, E. N., & Chudov, L. A. 1967, USSR Comput. Math. Math. Phys., 3, 1537
Schaal, K., Bauer, A., Chandrashekar, P., et al. 2015, MNRAS, 453, 4278
Schive, H.-Y., ZuHone, J. A., Goldbaum, N. J., et al. 2018, MNRAS, 481, 4815
Schneider, E. E., & Robertson, B. E. 2015, ApJS, 217, 24
Sedov, L. I. 1946, JApMM, 10, 241
Sheardown, A., Roediger, E., Su, Y., et al. 2018, ApJ, 865, 118
Shu, C. 1988, SIAM J. Sci. Stat. Comput., 9, 1073
Shu, C. 2009, SIAMR, 51, 82
Shu, C.-W. 1998, in Advanced Numerical Approximate of Nonlinear Hyperbolic Equations, ed. A. Quarteroni (Berlin: Springer), 285
Shu, C.-W. 2003, IJCFD, 17, 107
Shu, C.-W., & Osher, S. 1988, JCoPh, 77, 439
Shu, C.-W., & Osher, S. 1989, JCoPh, 83, 32
Sijacki, D., Vogelsberger, M., Kereš, D., Springel, V., & Hernquist, L. 2012, MNRAS, 424, 2999
Spiteri, R., & Ruuth, S. 2002, SJNA, 40, 469
Springel, V. 2010, MNRAS, 401, 791
Stone, J. M., Gardiner, T. A., Teuben, P., Hawley, J. F., & Simon, J. B. 2008, ApJS, 178, 137
Stone, J. M., & Norman, M. L. 1992, ApJS, 80, 753
Taylor, G. 1950, RSPSA, 201, 159
Teyssier, R. 2002, A&A, 385, 337
Tóth, G. 2000, JCoPh, 161, 605
Tóth, G., Chen, Y., van der Holst, B., & Daldorff, L. K. S. 2014, in ASP Conf. Ser. 488, 8th Int. Conf. Numerical Modeling of Space Plasma Flows (ASTRONUM 2013), ed. N. V. Pogorelov, E. Audit, & G. P. Zank (San Francisco, CA: ASP), 273
van Leer, B. 1979, JCoPh, 32, 101
Velasco Romero, D. A., Han Veiga, M., Teyssier, R., & Masset, F. S. 2018, MNRAS, 478, 1855
Verma, P. S., & Müller, W.-C. 2018, JSCom, 75, 941
Verma, P. S., Teissier, J.-M., Henze, O., & Müller, W.-C. 2019, MNRAS, 482, 416
von Neumann, J. 1942, The Point Source Solution U.S. Government Doc. AM-9 (National Defense Research Council, Division B: Washington, DC)
Wongwathanarat, A., Grimm-Strele, H., & Müller, E. 2016, A&A, 595, A41
Woodward, P., & Colella, P. 1984, JCoPh, 54, 115
Zhang, Y.-T., Shi, J., Shu, C.-W., & Zhou, Y. 2003, PhRvE, 68, 046709