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

Donnert, J.M.F.; Jang, H.; Mendygral, P.; Brunetti, G.; Ryu, D.; Jones, T.W.

DRAFT VERSION DECEMBER 12, 2018
Typeset using LaTeX **twocolumn** style in AASTeX62

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, PO Box 9513, NL-2300 RA Leiden, The Netherlands*
[2]*INAF Istituto di Radioastronomia, via P. Gobetti 101, I-40129 Bologna, Italy*
[3]*School of Physics and Astronomy, University of Minnesota, Minneapolis, MN 55455, USA*
[4]*Department of Physics, School of Natural Sciences, Ulsan National Institute of Science and Technology, UNIST-gil 50, Ulsan, 44919, Korea*
[5]*Cray Inc., Bloomington, MN, USA*
[6]*Minnesota Supercomputing Institute for Advanced Computational Research*

(Dated: Accepted ???. Received ???; in original form ???)

ABSTRACT

Due to increase in computing power, high-order Eulerian schemes will likely become instrumental for the 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 into the code `WOMBAT` . We establish the correctness of our implementation with an 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 up to 98 thousand cores on the Cray XC40 machine 'Hazel Hen', with a sustained performance of about 5% of peak at scale.

Keywords: general - methods: numerical - MHD

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 the magneto-hydrodynamic (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 methods for hydrodynamics pre-date the earliest computers and of course transcend the field of Astrophysics (Richardson 1922; LeVeque 2002). After seminal contributions by Lax (1954); Godunov (1959); Babuka & Zlmal (1973); van Leer (1979); Roe (1981) and nearly a century of development, major progress has been made in ever more accurate and efficient finite volume and finite difference schemes to minimize spurious errors and capture shocks accurately (Harten 1983; Colella & Woodward 1984; Harten et al. 1987; Shu 1988; Einfeldt 1988; Liu et al. 1994). For MHD, the development of constrained transport schemes (CT) marks the era of numerical magnetic fields with vanishing divergence to machine precision Evans & Hawley (1988); Londrillo & Del Zanna (2000).

jdonnert@strw.leidenuniv.nl
hanbyul@sirius.unist.ac.kr
pjm@cray.com
brunetti@ira.inaf.it
ryu@sirius.unist.ac.kr
twj@umn.edu

The plethora of available algorithms is only dwarfed by the vast amount of codes that use finite volume or finite difference methods today. In astrophysics, implementations would include AREPO (Springel 2010), ART (Kravtsov et al. 1997), ATHENA (Stone et al. 2008), CHOLLA (Schneider & Robertson 2015), DISPATCH (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 super computers now approaching the regime of 10^{18} floating point operations per second (Flops), algorithms beyond the most commonly used 2nd and 3rd order for MHD are well 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. 2018) 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 not necessarily true in three dimensions for a fifth order finite difference weighted essentially non oscillatory (WENO) scheme (Jiang & Shu 1996; Jiang & Wu 1999; Shu 1998; 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 oscillations near discontinuities. Modern WENO schemes come in too many flavors to list them all. 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 improvements of the scheme and extended it to very high order, for reviews see Shu (2009); Balsara et al. (2016).

We will show that the classical finite difference method doubles the effective resolution of the grid compared to popular PPM or TVD schemes in all dimensions. This is inline with prior results finding that WENO3, WENO5 and WENO9 double the effective resolution compared to the next lower order scheme (Zhang et al. 2003). The excellent fidelity of these schemes allows in principle to simulate a problem at half the resolution with WENO5 compared to PPM or WENO3, which more than makes up for added computational cost by the fifth order scheme. WENO5 also reduces the accumulated round-off error over many time steps and reduces the data size by a factor eight, which might be even more important than the reduction in computational cost.

However, due to the wide use of multi-core computers, most fluid simulations today are not compute bound. Thus a WENO implementation will likely run at the same resolution as the 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 finite difference 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[1] 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.

Here we present the implementation of a classical fifth order finite difference WENO scheme (Jiang & Wu 1999) in the WOMBAT [2] code framework (Mendygral et al. 2017). We use constrained transport 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 second order CT scheme to keep the implementation computationally cheap. High order schemes are much more computationally expensive at fifth order (e.g. Londrillo & del Zanna 2004; Mignone et al. 2010; Verma et al. 2019). In contrast, we will show that the 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). We test the code performance on modern HPC hardware in section 5. Conclusions are drawn in 6. For reference purposes, we once again present the eigenvectors used in the calculation in appendix A.

2. NUMERICAL METHOD

The equations of ideal magneto-hydrodynamics read (e.g. Ryu & Jones 1995):

[1] single instruction multiple data
[2] wombatcode.org

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

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

$$\frac{\partial P}{\partial t} + \mathbf{v} \cdot \nabla P + \gamma P \nabla \cdot \mathbf{v} = 0 \quad (3)$$

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

$$\nabla \cdot \mathbf{B} = 0 \quad (5)$$

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

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

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

$$S = \frac{P}{\rho^{\gamma - 1}}. \quad (8)$$

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

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

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

$$\mathbf{F}_x = \begin{pmatrix} \rho v_x \\ \rho v_x^2 + P^* - B_x^2 \\ \rho v_x v_y - B_x B_y \\ \rho v_x v_z - B_x B_z \\ 0 \\ B_y v_x - B_x v_y \\ B_z v_x - B_x v_z \\ (E + P^*) v_x - B_x (B_x v_x + B_y v_y + B_z v_z) \end{pmatrix} \quad (10)$$

Then the equations 1-4 can be written as a conservation law for[3] $\mathbf{q}$:

$$\frac{\partial \mathbf{q}}{\partial t} + \frac{\partial \mathbf{F}_x}{\partial x} = 0 \quad (11)$$

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

This equation can be approximated in quasi-linear form (Roe 1981, 1997; Einfeldt et al. 1991; van Leer 1979):

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

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

A complete set of left and right eigenvectors (see appendix A) 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 mode, 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, Jang et al. in prep.):

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

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

$$c_{\mathrm{A}} = \frac{B_x}{\sqrt{\rho}} \quad (15)$$

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

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

$$\frac{\mathrm{d}\mathbf{q}}{\mathrm{d}t} + \frac{1}{\Delta x} \left(\mathbf{F}^n_{i+\frac{1}{2}} - \mathbf{F}^n_{i-\frac{1}{2}} \right) = 0, \quad (16)$$

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 4th-order, 4-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 $\mathrm{CFL} = 0.8$ everywhere and is sufficiently robust in practice. Thus, in N_d dimensions the timestep is set as:

$$\Delta t < \frac{\mathrm{CFL}\Delta x}{\sum_{d=1}^{N_\mathrm{d}} \max(|v_d| + c^d_\mathrm{fast}))} \tag{17}$$

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

$$\begin{aligned}
\mathbf{q} &= \mathbf{q_0} = \mathbf{q}^n_{i,j,k} \\
\mathbf{q}_\mathrm{save} &= -\frac{4}{3}\mathbf{q_0} \\
\mathbf{q} &= \mathbf{q_0} - \frac{1}{2}\frac{\Delta t}{\Delta x}\left(\mathbf{F}_{i+1/2,j,k}(\mathbf{q}) - \mathbf{F}_{i-1/2,j,k}(\mathbf{q})\right) \\
\mathbf{q}_\mathrm{save} &= \mathbf{q}_\mathrm{save} + \frac{1}{3}\mathbf{q} \\
\mathbf{q} &= \mathbf{q}_0 - \frac{1}{2}\frac{\Delta t}{\Delta x}\left(\mathbf{F}_{i+1/2,j,k}(\mathbf{q}) - \mathbf{F}_{i-1/2,j,k}(\mathbf{q})\right) \\
\mathbf{q}_\mathrm{save} &= \mathbf{q}_\mathrm{save} + \frac{2}{3}\mathbf{q} \\
\mathbf{q} &= \mathbf{q}_0 - \frac{\Delta t}{\Delta x}\left(\mathbf{F}_{i+1/2,j,k}(\mathbf{q}) - \mathbf{F}_{i-1/2,j,k}(\mathbf{q})\right) \\
\mathbf{q}_\mathrm{save} &= \mathbf{q}_\mathrm{save} + \frac{1}{3}\mathbf{q} \\
\mathbf{q} &= \mathbf{q}_0 - \frac{1}{6}\frac{\Delta t}{\Delta x}\left(\mathbf{F}_{i+1/2,j,k}(\mathbf{q}) - \mathbf{F}_{i-1/2,j,k}(\mathbf{q})\right) \\
\mathbf{q}_\mathrm{save} &= \mathbf{q}_\mathrm{save} + \mathbf{q} \\
\mathbf{q}^{n+1}_{i,j,k} &= \mathbf{q}_\mathrm{save}
\end{aligned}$$

The CT time integration scheme is the same, but updates the face centered magnetic fields after the WENO step (see section 2.3). In 3 dimensions it adds the 3 component magnetic field on the boundary and the 3 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 a finite difference 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 finite volume 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, Jang et al. in prep.).

The fifth order weighted essentially non-oscillatory 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 the scheme is formally split, as we always integrate along the x-direction and rotate the grid to integrate along the other directions successively. However, the time integration scheme averages over all directions four times. Thus effectively the scheme is un-split at fourth order.

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:

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

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

$$\Delta F^s_{k+\frac{1}{2}}(m) = F^s_{k+1}(m) - F^s_k(m), \tag{20}$$

$$\Delta q^s_{k+\frac{1}{2}}(m) = q^s_{k+1}(m) - q^s_k(m), \tag{21}$$

respectivey. 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):

$$F^{s\pm}_i(m) = \frac{1}{2}\left(F^s_i(m) \pm \alpha(m) q^s_i(m)\right) \tag{22}$$

$$\Delta F^{s\pm}_{k+\frac{1}{2}}(m) = \frac{1}{2}\left(\Delta F^s_{k+\frac{1}{2}}(m) \pm \alpha(m)\Delta q^s_{k+\frac{1}{2}}(m)\right), \tag{23}$$

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 $\mathbf{F}_{i+\frac{1}{2}} = F^s_{i+\frac{1}{2}}(m) \cdot \mathbf{R}_{i+\frac{1}{2}}(m)$, 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):

$$\begin{aligned}
F^s_{i+\frac{1}{2}} = &\frac{1}{12}\left(-F^s_{i-1} + 7F^s_i + 7F^s_{i+1} - F^s_{i+2}\right) \\
&- \varphi_N\left(\Delta F^{s+}_{i-\frac{3}{2}}, \Delta F^{s+}_{i-\frac{1}{2}}, \Delta F^{s+}_{i+\frac{1}{2}}, \Delta F^{s+}_{i+\frac{3}{2}}\right) \\
&+ \varphi_N\left(\Delta F^{s-}_{i+\frac{5}{2}}, \Delta F^{s-}_{i+\frac{3}{2}}, \Delta F^{s-}_{i+\frac{1}{2}}, \Delta F^{s-}_{i-\frac{1}{2}}\right),
\end{aligned} \tag{24}$$

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

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

The non-linear weights are:

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

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

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

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 & saddle points). Borges et al. (2008) proposed the WENO-Z weights to make the scheme truly fifth order everywhere:

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

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

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

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

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 maintainance of a zero magnetic field divergence at all times. In constrained transport (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 $\mathbf{b} = (b_x, b_y, b_z)^T$. 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 x-direction (component five and six), g_z, g_x for the y-direction and h_x, h_y for the z-direction. The scheme includes a correction for the convection of the magnetic field (Ryu et al. (1998), compare to e.g. Verma et al. (2019)):

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

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

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

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

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

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

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 $\Omega_x, \Omega_y, \Omega_z$ are then:

$$\Omega_x = \frac{1}{2}\left(g^*_{x,i,j,k} + g^*_{x,i+1,j,k}\right) - \frac{1}{2}\left(f^*_{y,i,j,k} + f^*_{y,i,j+1,k}\right) \tag{42}$$

$$\Omega_y = \frac{1}{2}\left(h^*_{y,i,j,k} + h^*_{y,i,j+1,k}\right) - \frac{1}{2}\left(g^*_{z,i,j,k} + g^*_{z,i,j,k+1}\right) \tag{43}$$

$$\Omega_z = \frac{1}{2}\left(f^*_{z,i,j,k} + f^*_{z,i,j,k+1}\right) - \frac{1}{2}\left(h^*_{x,i,j,k} + h^*_{x,i+1,j,k}\right) \tag{44}$$

The face centered magnetic fields are then updated analogous to equation 16:

$$\frac{\mathrm{d}b_x}{\mathrm{d}t} + \frac{1}{\Delta x}\left(\Omega_{x,i,j,k} - \Omega_{x,i,j-1,k}\right) - \frac{1}{\Delta x}\left(\Omega_{z,i,j,k} - \Omega_{z,i,j,k-1}\right) = 0 \tag{45}$$

$$\frac{\mathrm{d}b_y}{\mathrm{d}t} + \frac{1}{\Delta x}\left(\Omega_{y,i,j,k} - \Omega_{y,i,j,k-1}\right) - \frac{1}{\Delta x}\left(\Omega_{x,i,j,k} - \Omega_{x,i-1,j,k}\right) = 0 \tag{46}$$

$$\frac{\mathrm{d}b_z}{\mathrm{d}t} + \frac{1}{\Delta x}\left(\Omega_{z,i,j,k} - \Omega_{z,i-1,j,k}\right) - \frac{1}{\Delta x}\left(\Omega_{y,i,j,k} - \Omega_{y,i,j-1,k}\right) = 0 \tag{47}$$

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

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

the other component follow accordingly along the j and k index.

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 Jang et al. in prep.).

3. IMPLEMENTATION

We implement the scheme in the numerical code WOMBAT [4]. 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 neighbouring patches and MPI ranks. Thus there is no global data structure keeping track of patch distribution that may grow with 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 neighbouring MPI ranks are included into a ranks domain. Patches are off-loaded 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 neighbours mailbox of user-definable size. A signal per mailbox of 8 bytes (the "heartbeat") is used to notify the neighbouring 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 multi-user 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 exa-scale 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 resides in rank global memory. At this point, no communication is required anymore, 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 (NUMA) 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 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 (eq. 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 pre-requisite to achieve a significant frac-

[4] wombatcode.org

tion 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 $\mathbf{q}$, a state vector buffer $\mathbf{q}_{\rm save}$ to accumulate intermediate results and a copy of the initial state $\mathbf{q}_0 = \mathbf{q}_i^n$. Note that we communicate boundaries twice (once for WENO, once for CT update) per RK4 sub-step. This allows us to retain 3 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 $\mathbf{q}$. If executing the first substep, initialize $\mathbf{q}_0$ and set $\mathbf{q}_{\rm save}$.
2. Compute cell centered pressure P eq. 7, eigenvalues $c_{\rm A}, c_{\rm fast}, c_{\rm slow}$ eq. 13 to 15 and fluxes $\mathbf{F}$ eq. 10.
3. Compute eigenvectors $\mathbf{L}$ and $\mathbf{R}$ on the cell boundaries, compute WENO fluxes, eq. 24.
4. Update $\mathbf{q}$ using WENO fluxes, update $\mathbf{q}_{\rm save}$ using new $\mathbf{q}$.
5. Move $\mathbf{q}, \mathbf{q}_0, \mathbf{q}_{\rm save}$ on the patch, communicate patch boundaries.

Repeat 4 times with corresponding Runge-Kutta factors so that $\mathbf{q}^{n+1} = \mathbf{q}_{\rm save}$ 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 $\mathbf{q}$. If executing first substep, initialize $\mathbf{q}_0$ and set $\mathbf{q}_{\rm save}$, else copy-in and flatten them. Set a buffer $\mathbf{q}_{\rm buf} = \mathbf{q}_0$.
2. Compute from $\mathbf{q}$ cell centered pressure P eq. 7, eigenvalues $c_{\rm A}, c_{\rm fast}, c_{\rm slow}$ eq. 13 to 15 and fluxes $\mathbf{F}$ eq. 10.
3. Compute from $\mathbf{q}$ eigenvectors $\mathbf{L}$ and $\mathbf{R}$ on the cell boundaries, compute WENO fluxes, eq. 24, compute 2d CT-fluxes eq. 36, 37.
4. Update $\mathbf{q}_{\rm buf}$ using WENO fluxes
5. Rotate $\mathbf{q}$ forward so flattened index is along y-direction.
6. Compute from $\mathbf{q}$ the cell centered pressure P eq. 7, eigenvalues $c_{\rm A}, c_{\rm fast}, c_{\rm slow}$ eq. 13 to 15 and fluxes eq. 10.
7. Compute from $\mathbf{q}$, eigenvectors $\mathbf{L}$ and $\mathbf{R}$ on the cell boundaries, WENO fluxes, eq. 24. The compute 2d CT-fluxes 38, 39.
8. Rotate $\mathbf{q}$ and CT-fluxes backwards.
9. Update $\mathbf{q}_{\rm buf}$ using WENO fluxes, copy $\mathbf{q}_{\rm buf}$ into $\mathbf{q}$, compute CT corner flux Ω_x eq. 42.
10. Update $\mathbf{q}_{\rm save}$ using new $\mathbf{q}$, but not the magnetic field.
11. Move $\mathbf{q}, \mathbf{q}_0, \mathbf{q}_{\rm save}$, Ω_x on the patch, communicate patch boundaries, face centered magnetic fields and corner flux.
12. Copy-in and flatten $\mathbf{q}$,$\mathbf{b}$, Ω_x.
13. Update face centered magnetic field $\mathbf{b}$ from Ω_x.
14. Interpolate face centered magnetic field $\mathbf{b}$ to zone centered magnetic field $\mathbf{B}$ at fourth order. Update $\mathbf{q}_{\rm save}$ with new magnetic field.
15. Move $\mathbf{q}$, $\mathbf{b}$, $\mathbf{q}_{\rm save}$ to the patch object.

Repeat 4 times with corresponding Runge-Kutta factors so that $\mathbf{q}^{n+1} = \mathbf{q}_{\rm save}$ at the last substep. Perform another CT update on $\mathbf{q}_{\rm save}$ before moving it into $\mathbf{q}$.

3.1.3. *WENO-WOMBAT in three Dimensions*

In three dimensions the program proceeds as follows:

1. Copy-in and flatten $\mathbf{q}$. If executing first substep, initialize $\mathbf{q}_0$ and set $\mathbf{q}_{\rm save}$, else copy-in and flatten them. Set a buffer $\mathbf{q}_{\rm buf} = \mathbf{q}_0$.
2. Compute from $\mathbf{q}$ cell centered pressure P eq. 7, eigenvalues $c_{\rm A}, c_{\rm fast}, c_{\rm slow}$ eq. 13 to 15 and fluxes $\mathbf{F}$ eq. 10.
3. Compute from $\mathbf{q}$ eigenvectors $\mathbf{L}$ and $\mathbf{R}$ on the cell boundaries, compute WENO fluxes, eq. 24, compute 3d CT-fluxes eq. 36 37.
4. Update $\mathbf{q}_{\rm buf}$ using WENO fluxes
5. Rotate $\mathbf{q}$ and $\mathbf{q}_{\rm buf}$ forward so flattened index is along y-direction.
6. Compute from $\mathbf{q}$ cell centered pressure P eq. 7, eigenvalues $c_{\rm A}, c_{\rm fast}, c_{\rm slow}$ eq. 13 to 15 and fluxes eq. 10.
7. Compute from $\mathbf{q}$ eigenvectors $\mathbf{L}$ and $\mathbf{R}$ on the cell boundaries, compute WENO fluxes, eq. 24, compute 3d CT-fluxes eq. 38 39.
8. Update $\mathbf{q}_{\rm buf}$ using WENO fluxes,
9. Rotate $\mathbf{q}$ and $\mathbf{q}_{\rm buf}$ forward so flattened index is along z-direction. Rotate CT-fluxes backwards.

10. Compute from $\mathbf{q}$ the cell centered pressure P eq. 7, eigenvalues $c_{\rm A}, c_{\rm fast}, c_{\rm slow}$ eq. 13 to 15 and fluxes eq. 10.

11. Compute from $\mathbf{q}$ eigenvectors $\mathbf{L}$ and $\mathbf{R}$ on the cell boundaries, compute WENO fluxes, eq. 24, compute 3d CT-fluxes using eq. 40 41.

12. Update $\mathbf{q}_{\rm buf}$ using WENO fluxes

13. Rotate forward $\mathbf{q}$ and $\mathbf{q}_{\rm buf}$ and CT-fluxes, so the first index runs along x-direction again.

14. Update $\mathbf{q}_{\rm buf}$ using WENO fluxes, copy $\mathbf{q}_{\rm buf}$ into $\mathbf{q}$, compute CT corner flux eq. 42-44 from rotated CT fluxes.

15. Update $\mathbf{q}_{\rm save}$ using new $\mathbf{q}$, but not the magnetic field.

16. Move $\mathbf{q}, \mathbf{q}_0, \mathbf{q}_{\rm save}, \mathbf{\Omega}$ on the patch, communicate patch boundaries, face centered magnetic fields $\mathbf{b}$ and corner flux.

17. Copy-in and flatten $\mathbf{q}$,$\mathbf{b}$, $\mathbf{\Omega}$.

18. Update face centered magnetic field $\mathbf{b}$ from $\mathbf{\Omega}$.

19. Interpolate face centered magnetic field $\mathbf{b}$ to zone centered magnetic field $\mathbf{B}$ at fourth order. Update $\mathbf{q}_{\rm save}$ with new magnetic field.

20. Move $\mathbf{q}, \mathbf{b}, \mathbf{q}_{\rm save}$ to the patch object.

Repeat 4 times with corresponding Runge-Kutta factors so that $\mathbf{q}^{n+1} = \mathbf{q}_{\rm save}$ at the last substep. Perform another CT update on $\mathbf{q}_{\rm save}$ before moving it into $\mathbf{q}$.

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 $\mathbf{U} = (\rho, v_{\rm x}, v_{\rm y}, v_{\rm z}, B_{\rm x}, B_{\rm y}, B_{\rm z}, P)^T$, or as conserved variables $\mathbf{Q} = (\rho, \rho v_{\rm x}, \rho v_{\rm y}, \rho v_{\rm z}, B_{\rm x}, B_{\rm y}, B_{\rm z}, E)^T$, where $E(\mathbf{U})$ 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:

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

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 Gardiner & Stone (2005, 2008); Stone et al. (2008); 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 dimension. This test evaluates the code in the linear regime and is very valuable for debugging. As the setup is not trivial, especially in 3 dimensions, we describe it in greater detail here, but note that the `WOMBAT` or `ATHENA` source code is likely indispensible to fully reproduce the results.

Following Gardiner & Stone (2005, 2008); Stone et al. (2008), we set $\gamma = 5/3$ and $\bar{\mathbf{U}} = (1, v_x, 0, 0, B_x, B_y, B_z, 1/\gamma)^T$, where $v_x = 1$ for shear and entropy modes, and $v_x = 0$ otherwise. For hydro waves, $\mathbf{B} = 0$, for MHD waves $\mathbf{B} = (1, \sqrt{2}, 1/2)^T$. We add a perturbation on the conserved variables: $\mathbf{Q} = \bar{\mathbf{Q}} + A_0 \mathbf{R}_k \sin(2\pi x)$. Here $\mathbf{R}_k$ 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:

$$\mathbf{R}_{\rm Alfven} = \frac{1}{6\sqrt{5}} \left(0, 0, 1, -2\sqrt{2}, 0, -1, 2\sqrt{2}, 0\right)^T \qquad (50)$$

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

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

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

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

$$\mathbf{R}_{\rm shear,y} = (0, 0, 1, 0, 0, 0, 0, 0)^T \qquad (55)$$

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 lab frame. We set a magnetic vector potential:

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

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

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

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

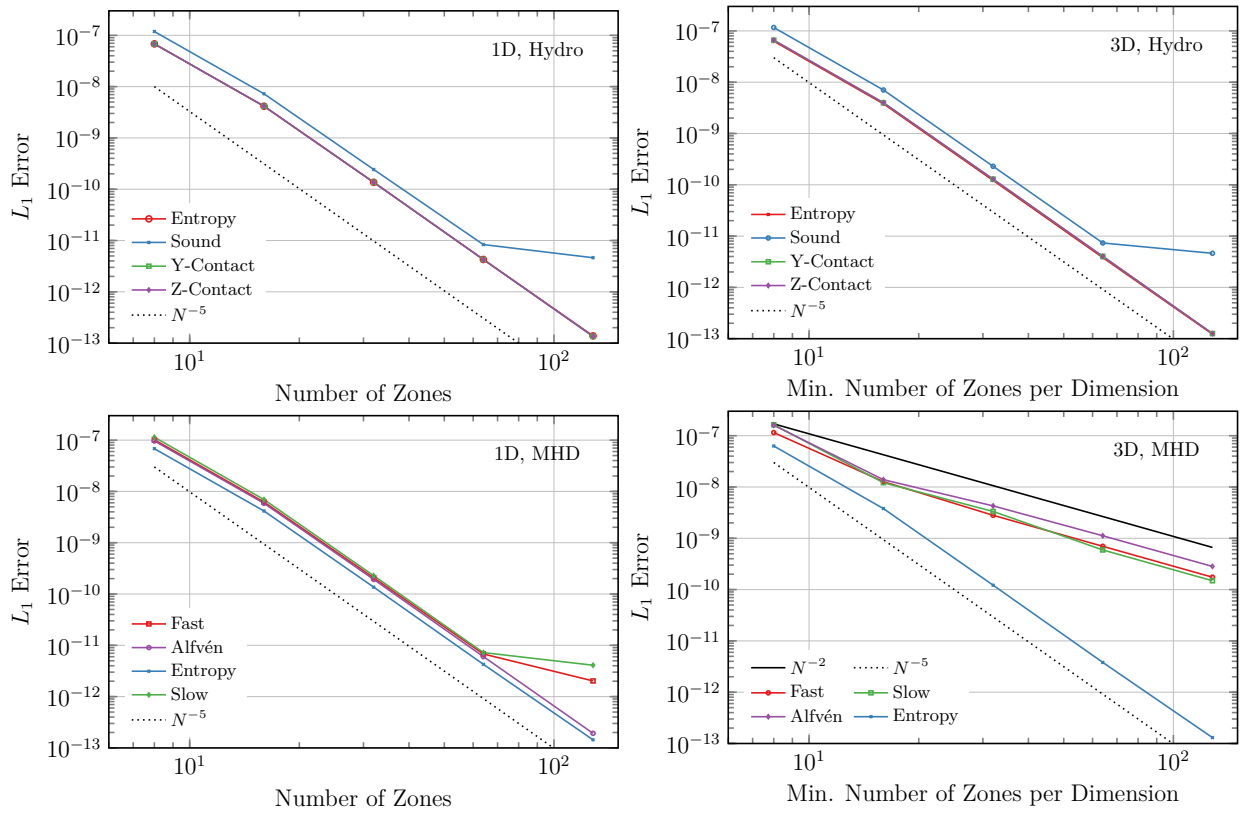

Figure 1. L_1 error over resolution for advecting hydrodynamic (top) and MHD waves (bottom) in one dimension (left) and in three dimensions (right) with WENO5-Z across a unit computational domain for one wave length following (Gardiner & Stone 2008). Fifth order was marked as dotted black line, second order as black line.

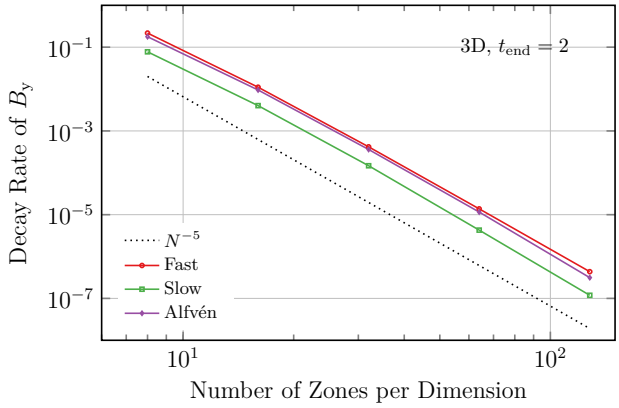

Figure 2. Decay rate (eq. 60) of the y-component of the magnetic field between $t = 0$ and $t = 2$ for the fast mode (red), the slow mode (green) and the Alfvén mode (purple) with WENO5-Z.

vector potential has to be evaluated on the edges of the zone in three dimension, so that the resulting magnetic field is field centered. E.g. in our case A_x is defined at $(i, j + \frac{1}{2}, k + \frac{1}{2})$. 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 finite difference rotation operator. We observe $\nabla \cdot \mathbf{B} < 10^{-12}$ in all tests at all times.

In figure 1 we plot the resulting L_1 error for resolutions between 8 and 128 zones run with WENO5-Z. Top left to bottom right we show 1D hydro waves, 3D hydro 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.

In three dimensions, all hydro 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 (eq.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.

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 2 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 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 constrained transport method from Ryu et al. (1998) is in the form of 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 two, 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 Wave*

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 $\mathbf{U} = (1, 0, v_y, v_z, 1, B_y, B_z, 0.1)^T$, 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 $\mathbf{U}$ by the second Euler angle $\theta = \tan^{-1}(2) \approx 63.5\,\deg$. 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.

On the top left of figure 3, we show the L_1 error norm in 2d (blue) and 3d (green) over number of zones after the wave traveled for five wave period ($c_{\rm A} = 1$) 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 3. However, this time oscillations in the RMS of the state vector components can lead to negative decay rates even at rather late times. In figure 3 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):

$$\delta F_{\rm B,tv} = {\rm RMS}\left(\sqrt{(B_{\rm y}v_{\rm y})^2 + (B_{\rm z}v_{\rm z})^2}\right) \qquad (61)$$

For this quantity, fifth order convergence is observed at all times in figure 3, top left.

In figure 3, 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 resolution. This is a significant improvement over the results shown in Tóth (2000) for the same CT scheme. It also compares favourably 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 $\mathbf{U}_{\rm L} = (1.08, 1.2, 0.01, 2/\sqrt{4\pi}, 3.6/\sqrt{4\pi}, 2/\sqrt{4\pi}, 0.95)^T$. The right hand state vector is $\mathbf{U}_{\rm R} = (1, 0, 0, 0, 2/\sqrt{4\pi}, 4/\sqrt{4\pi}, 2/\sqrt{4\pi}, 1$ We plot the TVD+CTU result with 10^4 zones as red line and the WENO5 result with 256 zones as black circles. The evolved state vector components at $t = 0.2$ for the 1 dimensional computation are found in figure 4, for 3 dimensions in figure 5. For 3 dimensions, we rotated the shock normal along $\mathbf{k} = (1, 1, 1)^T$ 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 5.

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 $\mathbf{U}_{\rm L} = (1, 0, 0, 0, 0.7, 0, 0, 1)^T$. The

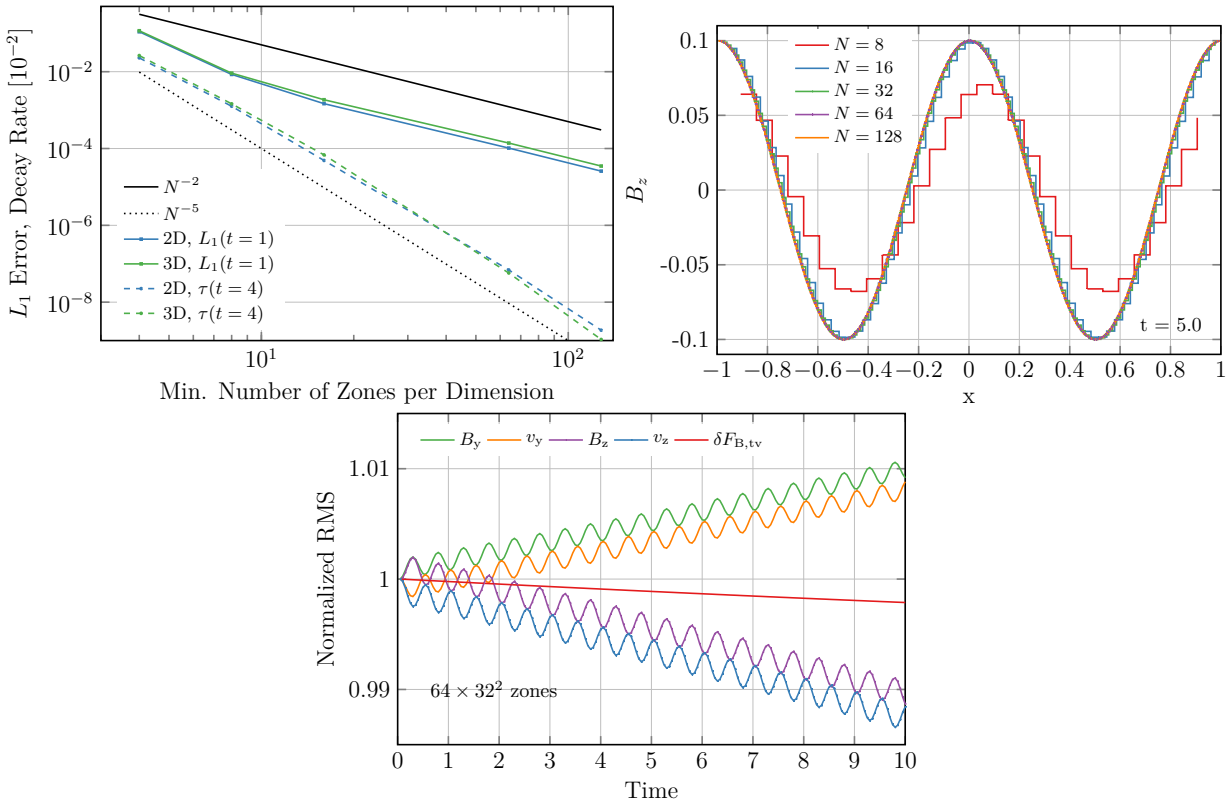

Figure 3. Top left: L_1 error of the circularly polarized Afvén wave test over resolution in 2d (blue) and 3d (green). Decay rate of transverse magnetic flux as dashed lines in units of 100. We mark second order as solid and fifth order as dotted black lines Top right: B_z for the circularly polarized 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.

right hand state vector is $\mathbf{U}_\mathrm{R} = (0.3, 0.0.1, 0.7, 1, 0, 0.2)^T$. We plot the TVD+CTU result with 10^4 zones as 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 6.

4.5. *Brio & Wu Shock Tube*

The MHD shock tube from Brio & Wu (1988a) has the initial conditions: $\mathbf{U}_\mathrm{L} = (0.125, 0, 0, 0, 0.75, 1, 0, 1)^T$, $\mathbf{U}_\mathrm{R} = (0.125, 0, 0, 0, 0.75, -1, 0, 0.1)^T$, with $\gamma = 2$. In figure 7, 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 red line.

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 $B_\mathrm{y} = B_\mathrm{z} = 1/\sqrt{2}$. However, in this particular problem, $B_\mathrm{z} = 0$ at all times. Thus the test exposes this choice in the eigenvectors. Setting $B_\mathrm{z} = 0$ for this problem only, would remove the oscillations in our figure 7. Balbs & Tadmor (2006) have shown that the issue can also be fixed with global smoothness indicators. In real world applications in 2 or more dimension, this is not an issue. Jang et al. in prep. 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 & right state $\mathbf{U}_\mathrm{L} = (3.857143, 2.629369, 0, 0, 0, 0, 0, 31/3)^T$, $\mathbf{U}_\mathrm{R} = (1 + 0.2\sin(5x), 0, 0, 0, 0, 0, 0, 1)^T$ 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 8, we show the result from WENO5 (green circles) and WENO5-Z simulations (black squares) with 300 zones and TVD+CTU with 10000 zones (red line). Clearly the WENO-Z result traces the complex flow behind the shock more accurately than the WENO5 result.

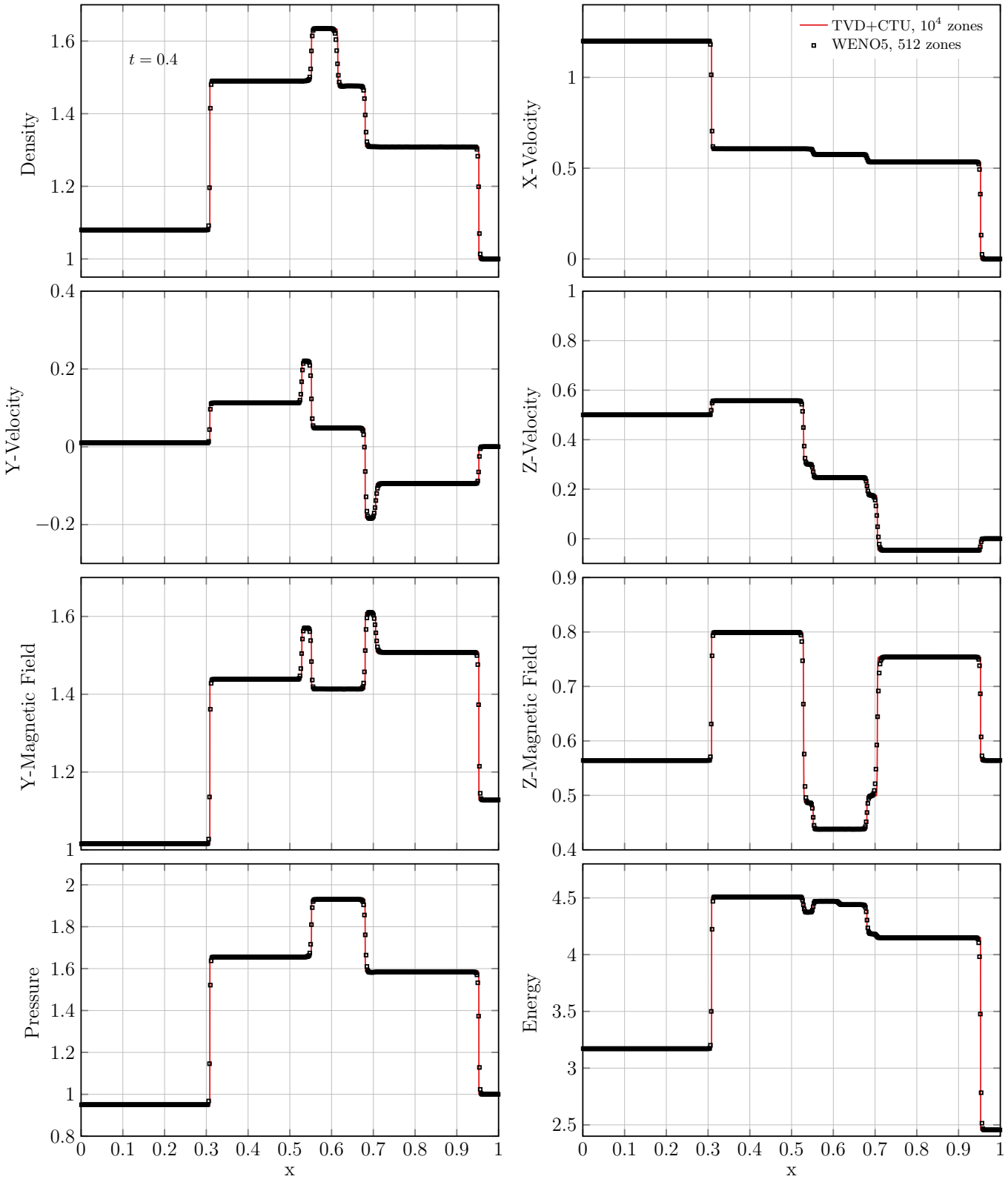

Figure 4. Shock tube test 2a from Ryu & Jones (1995) at time $t = 0.2$ in one dimension. TVD+CTU result with 10000 zones as red line, WENO5 result with 512 zones as black squares. Density, velocity components, magnetic field components, pressure and energy are shown from top left to bottom right.

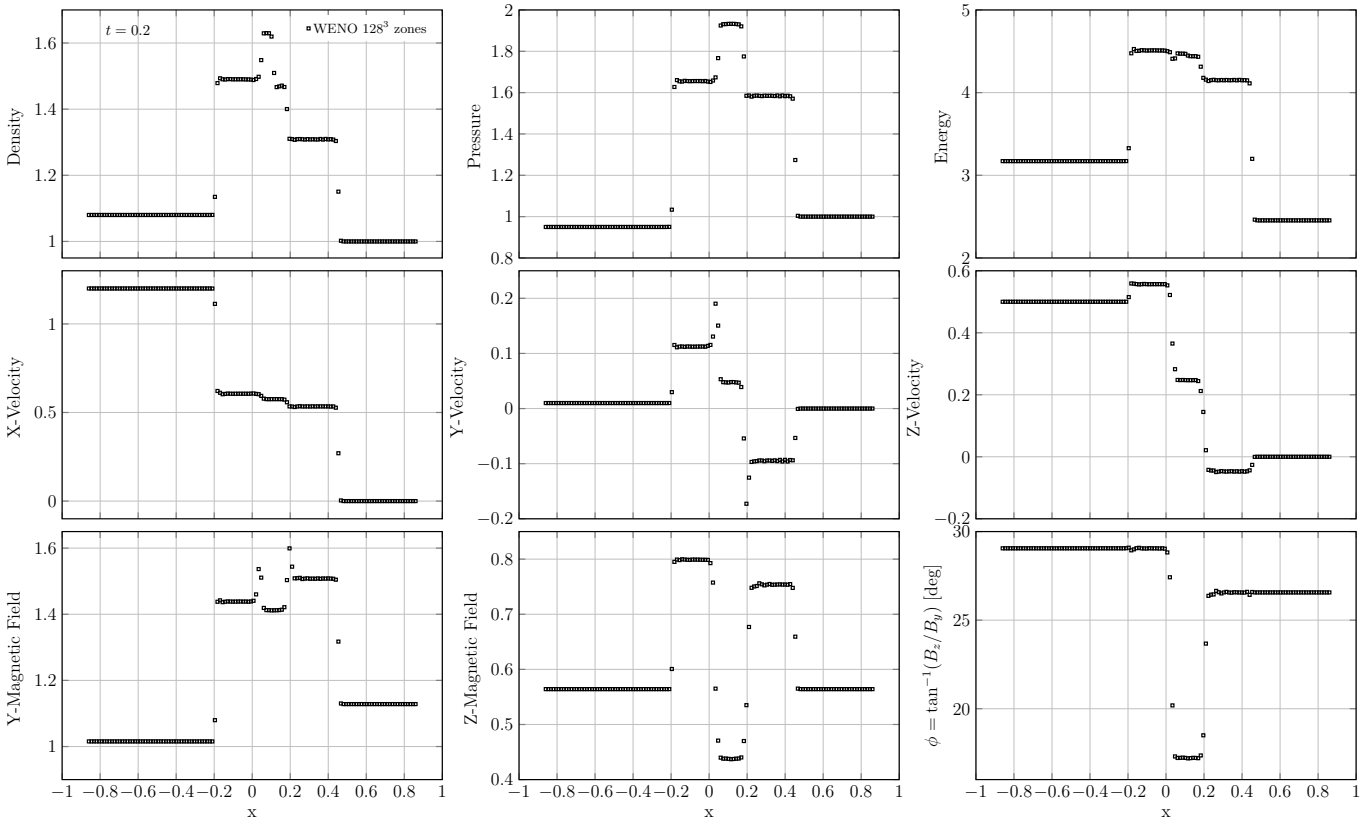

Figure 5. Shock tube test 2a from Ryu & Jones (1995) at time $t = 0.2$ in three dimensions along the zone diagonal of the computational domain. WENO5 result with 128^3 zones as black squares. Density, pressure, energy, velocity components, magnetic field components and magnetic field angle ϕ in degrees are shown from top left to bottom right.

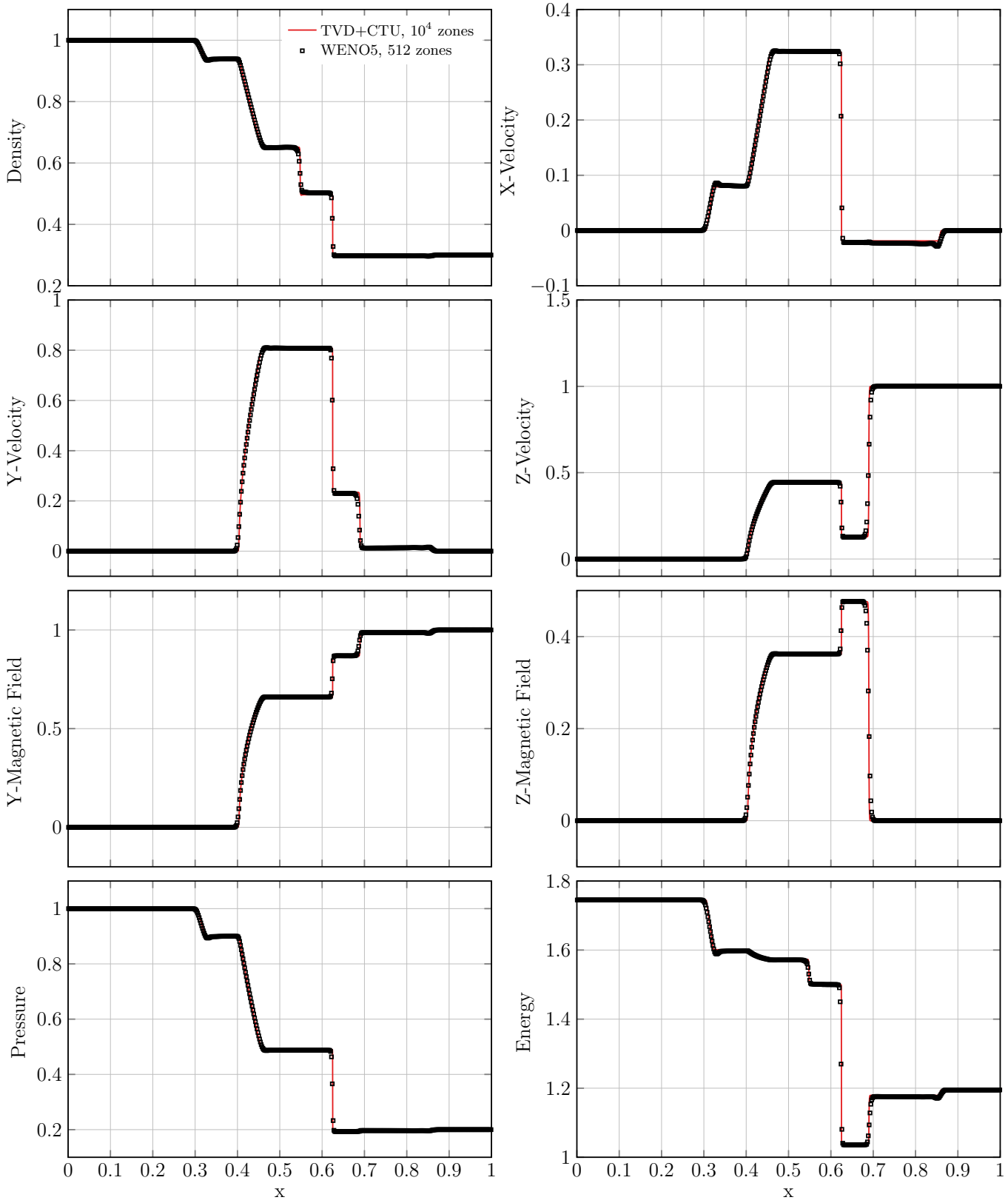

Figure 6. Shock tube test 4d from Ryu & Jones (1995) at time $t = 0.2$ in one dimension. TVD+CTU result with 10000 zones as red line, WENO5 result with 256 zones as black squares. Density, velocity components, magnetic field components, pressure and energy are shown from top left to bottom right.

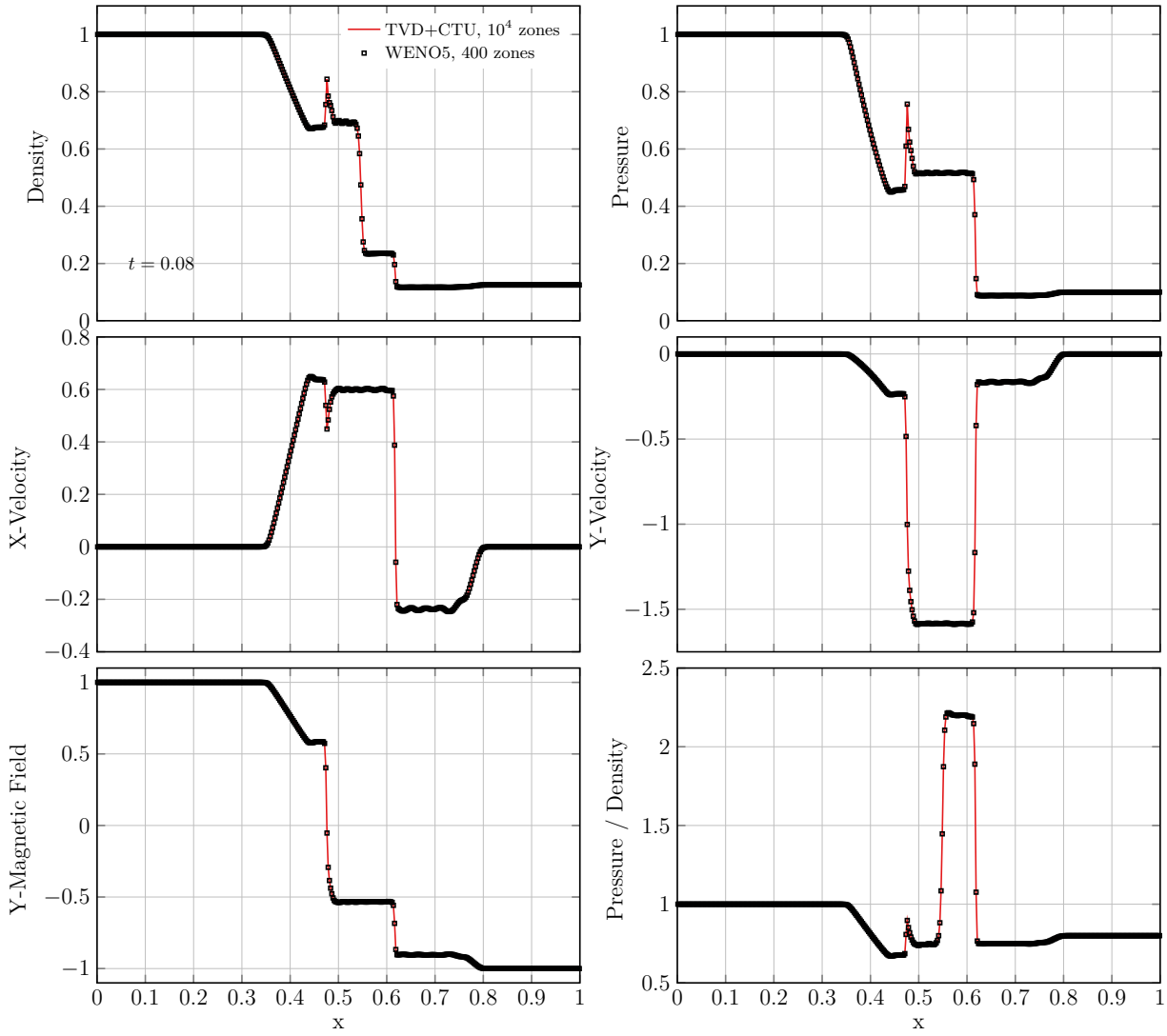

Figure 7. Shock tube test from Brio & Wu (1988b) at time $t = 0.08$. We show Density, x and y component of velocity, y component of magnetic field, pressure and energy from top left to bottom right. Results from WENO5 with 400 zones as black squares, TVD+CTU with 10000 zones as red line.

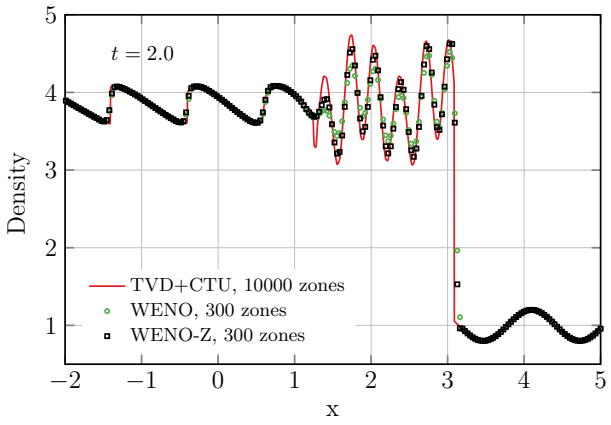

Figure 8. Density of the shock-density wave interaction test at time $t = 1$

4.7. *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 non-linear 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 non-linearity 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 Appendix A), 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 non-linear and thus may not be correct with four byte variables.

We set $\mathbf{U} = (1, 0, 0, 0, 0, 0, 0, 1)^T$ everywhere in a periodic domain with size $L_x = L_y = 0.3$ and 200^2 zones. When $x+y < 0.15$, we set $\mathbf{U} = (0.125, 0, 0, 0, 0, 0, 0, 0.14)^T$. We show the resulting density in figure 9 at $t = 0.2, 0.3, 0.5, 0.7$ (top left to bottom right) on a colour 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); Sijacki et al. (2012). We show the vorticity of the test at $t = 0.3$ in figure 10, 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. There are sizable differences in the shape of the "bird" in the lower left corner of the simulation. In particular, interfaces in the Lagrangian finite volume result 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 non-linearity 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 non-linear solutions of the hydrodynamic equations, which is of course not a new result at all (e.g. McNally et al. 2012).

4.8. *OrszagTang 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: $\mathbf{U} = (25/(36\pi), -\sin(2\pi y), \sin(2\pi y), 0, B_x, B_y, B_z, 5/(12\pi))^T$, 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 11, top left to bottom right: 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 12, 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 13 we compare the pressure at $y = -01875$, $-0.5 < x < 0$ 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 as red line. Apparently, WENO5-Z has comparable or better fidelity than 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 12 shows that the blip becomes fully resolved at 192^2.

In figure 14 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 $K_{\rm GD} = 1$, $L = 1$ and a contrast parameter of $C = 0.075$. Schlieren images visualise the gradient of the solution, which makes the Orszag-Tang vortex test more discriminative between schemes. We observe that all main features of the low resolution TVD+CTU simulation are reproduced in the WENO5-Z simulation at

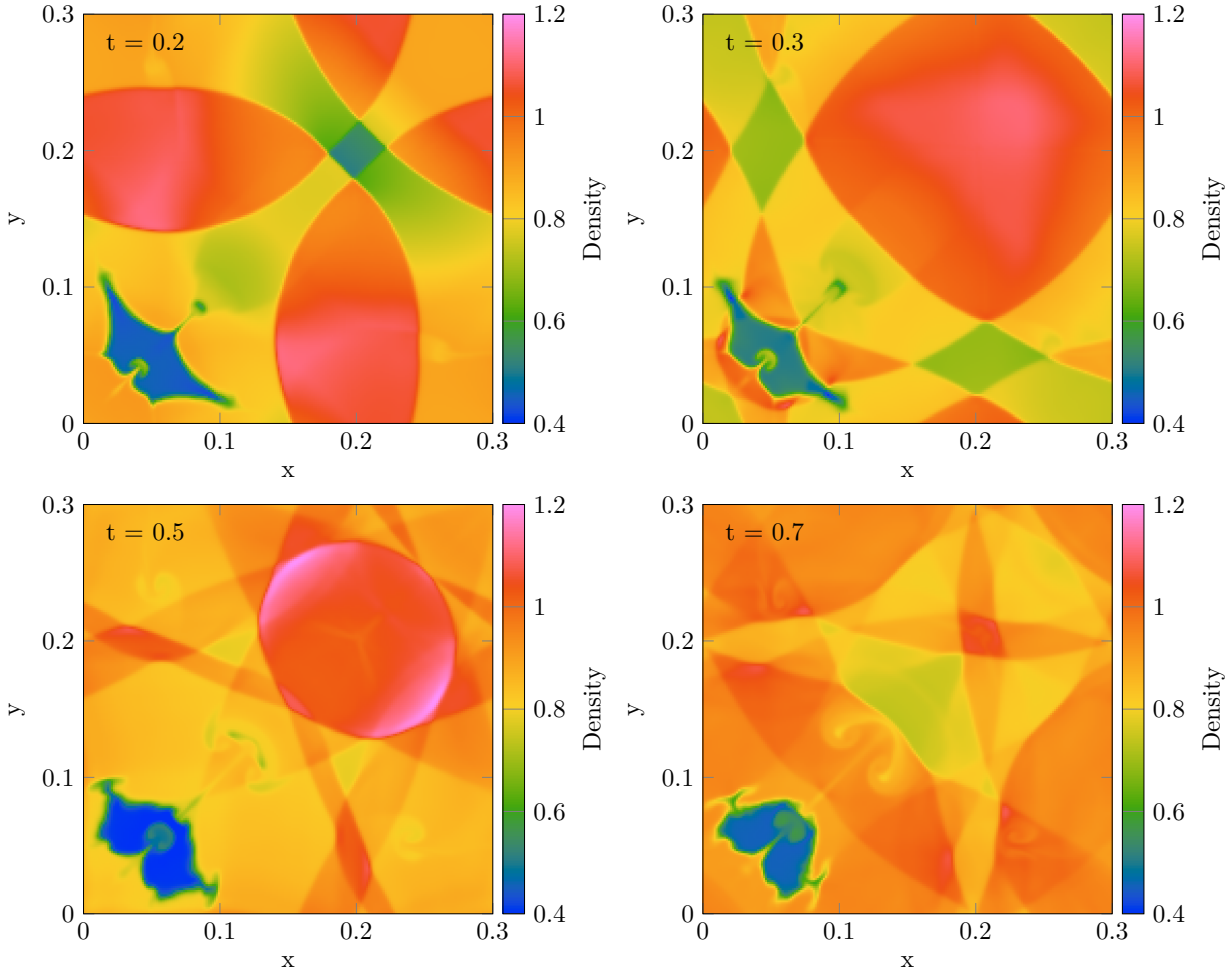

Figure 9. Density of the implosion test at times $t = 0.2$, $t = 0.3$, $t = 0.5$ and $t = 0.7$ (top left to bottom right). The color bar ranges from 0.4 to 1.2, the resolution was 200^2 zones.

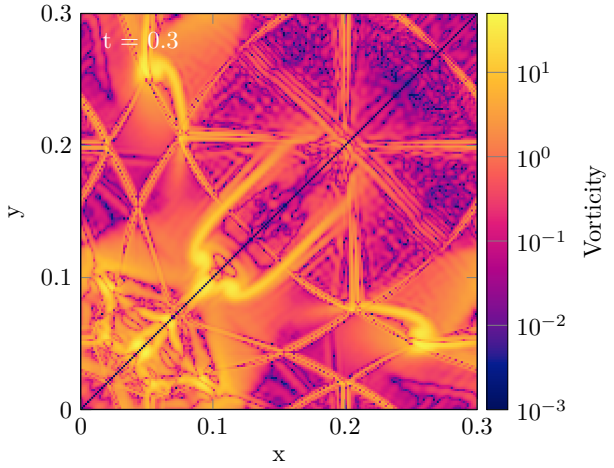

Figure 10. Logarithm of vorticity of the implosion test at time $t = 0.3$. The color bar ranges from 0.001 to 50, the resolution was 200^2 zones.

half that resolution. We conclude that WENO5-Z doubles the effective resolution compared to TVD+CTU in this test.

4.9. *MHD Rotor*

The MHD rotor test simulates a rapidly rotating disk in 2 dimensions, which produces shocks and rare-fraction waves as the disk expands due to the 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 $P = 1$, $\mathbf{B} = (5/\sqrt{4\pi}, 0)^T$,$v_0 = 2$, $r_0 = 0.1$, $r_1 = 0.115$ and $f(r) = \frac{r_1 - r}{r_1 - r_0}$ and (Tóth 2000):

$$\rho = \begin{cases} 10 & \Leftrightarrow r < r_0 \\ 1 + 9f(r) & \Leftrightarrow r_0 \le r \le r_1 \\ 1 & \text{otherwise.} \end{cases} \tag{62}$$

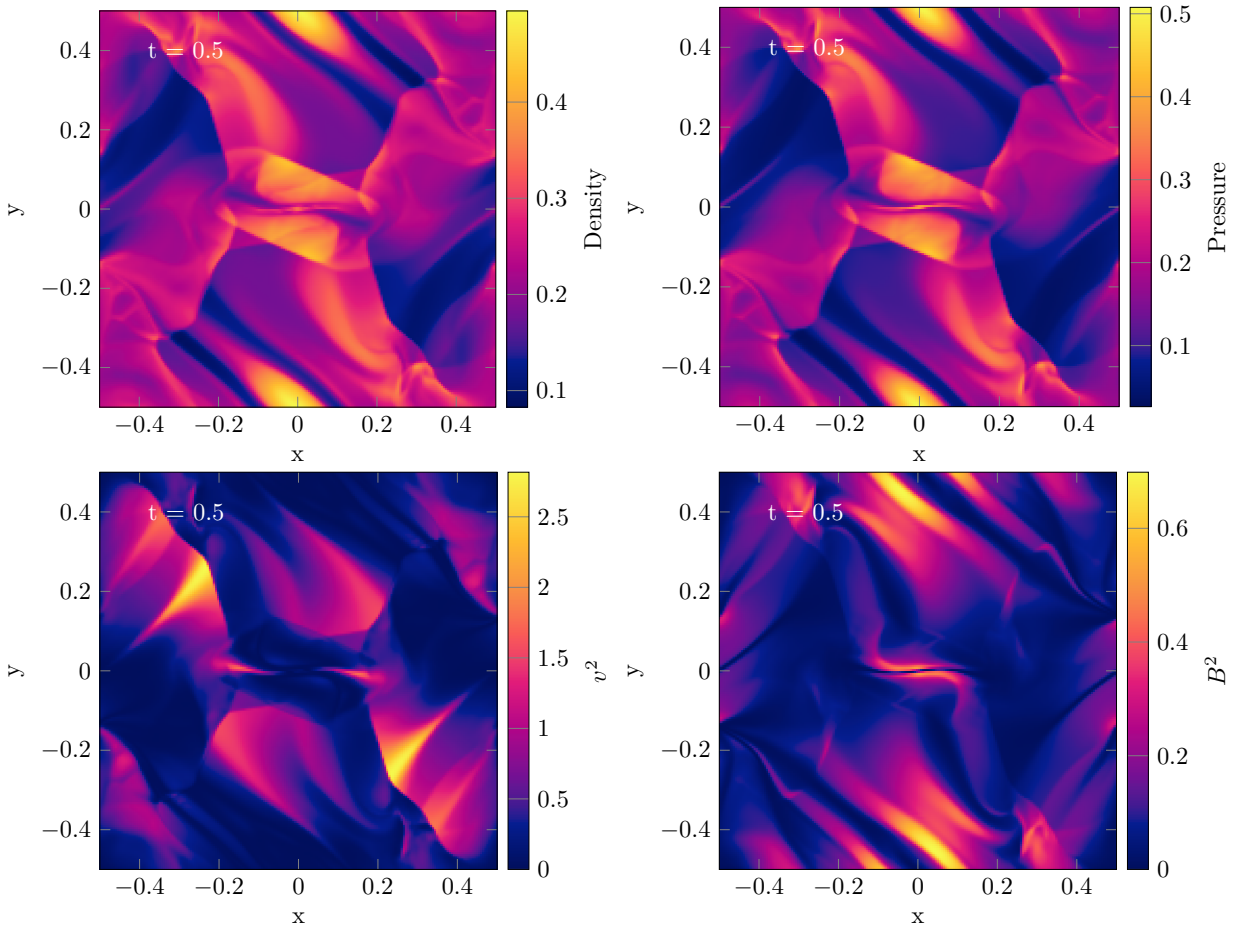

Figure 11. Images of the Orszag-Tang vortex test at time $t = 0.5$ with 192^2 zones. Top-left to bottom right: density, pressure, v^2 and B^2.

the velocity components are:

$$v_x = \begin{cases} -y\frac{v_0}{r_0} & \Leftrightarrow r < r_0 \\ -f(r)y\frac{v_0}{r_0} & \Leftrightarrow r_0 \le r \le r_1 \\ 0 & \text{otherwise} \end{cases} \tag{63}$$

$$v_y = \begin{cases} -x\frac{v_0}{r_0} & \Leftrightarrow r < r_0 \\ -f(r)x\frac{v_0}{r_0} & \Leftrightarrow r_0 \le r \le r_1 \\ 0 & \text{otherwise,} \end{cases} \tag{64}$$

We show the test result with WENO5 and 200^2 zones at time $t = 0.15$ in figure 15. A slice through the rotor at $t = 0.5$ and $y = -0.1875$ (top) and $y = 0.073$ are shown in figure 16. We show the WENO5 solution as black squares, TVD+CTU with 1024^2 zones as red line and 400^2 zones (blue dots). No asymmetries or oscillations are visible, the WENO5 solution at 200^2 resolves the small features comparably 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.

4.10. *Double Mach Reflection Test*

The double Mach reflection test simulates the evolution of a Mach 10 shock at an oblique 60 deg 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 $\mathbf{U} = (1.4, 0, 0, 0, 0, 0, 0, 1)^T$, the domain size is 3.25×1. Because the shock is reflected of 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. $x_{\text{shock}} = 1/6 + (1 + 20t)$. 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 numer-

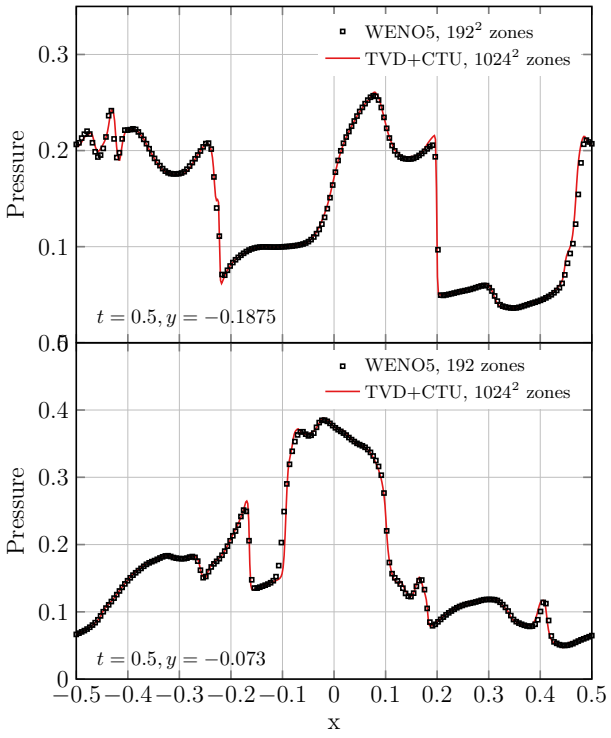

Figure 12. Pressure in the Orszag-Tang vortex test at $y = -01875$ (top) and $y = 0.073$ (bottom) and at time $t = 0.5$. WENO5-Z with 192^2 zones as black squares, TVD+CTU with 1024^2 zones as red line.

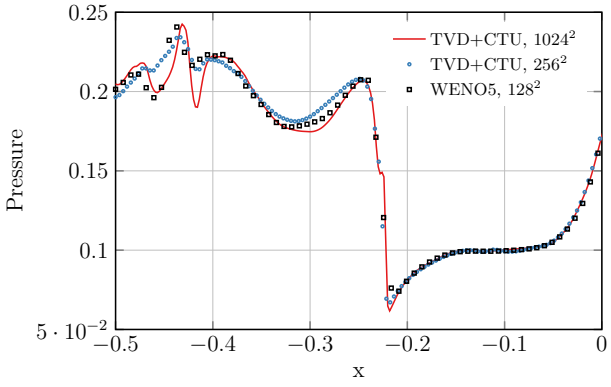

Figure 13. Pressure in the Orszag-Tang vortex test at $y = -01875$, $-0.5 < x < 0$ and at time $t = 0.5$. WENO5-Z with 128^2 zones as black squares, TVD+CTU with 1024^2 zones as red line, TVD+CTU with 256^2 zones as blue circles.

ical 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 17, we show the density from the test run with the standard WENO5 scheme at times $t = 0.05, 0.1, 0.2$ (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. 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 not observe the instability in the standard WENO5 scheme, which points towards good robustness of the scheme in applications that contain complicated sub-grid physics. This is very desirable in cosmological simulations.

4.11. *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; Beck et al. 2016; Schaal et al. 2015; McNally et al. 2012; Hopkins 2015). In particular, the KH problem was used to expose the artifical surface tension of traditional SPH methods (e.g. Hopkins 2013). It also exposes the velocity dependent truncation error in Eulerian grid methods (Robertson et al. 2010). Our setup follows Lecoanet et al. (2016):

$$\rho = 1 + \frac{\Delta\rho}{\rho_0}\frac{1}{2}\left[\tanh\left(\frac{z-z_1}{a}\right) - \tanh\left(\frac{z-z_2}{a}\right)\right] \tag{65}$$

$$v_x = v_{\rm flow}\left[\tanh\left(\frac{z-z_1}{a}\right) - \tanh\left(\frac{z-z_2}{a}\right) - 1\right] \tag{66}$$

$$v_z = A\sin(2\pi x)\left[\exp\left(-\frac{(z-z_1)^2}{\sigma^2}\right) + \exp\left(-\frac{(z-z_2)^2}{\sigma^2}\right)\right] \tag{67}$$

$$P = P_0, \tag{68}$$

with the parameters, $A = 0.01$, $P_0 = 10$, $a = 0.05$, $\sigma = 0.4$, $z_1 = 0.5$, $z_2 = 1.5$, $u_{\rm flow} = 1$, $\Delta\rho/\rho_0 = \rho_0 = 1$. 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 18, we show the density of four simulations at $t = 2$. Top left to bottom right panels show WENO5-Z without boost with 512×1024 zones 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.

The TVD+CTU run does not develop a vortex roll due to numerical diffusivity, even at twice the resolution from the

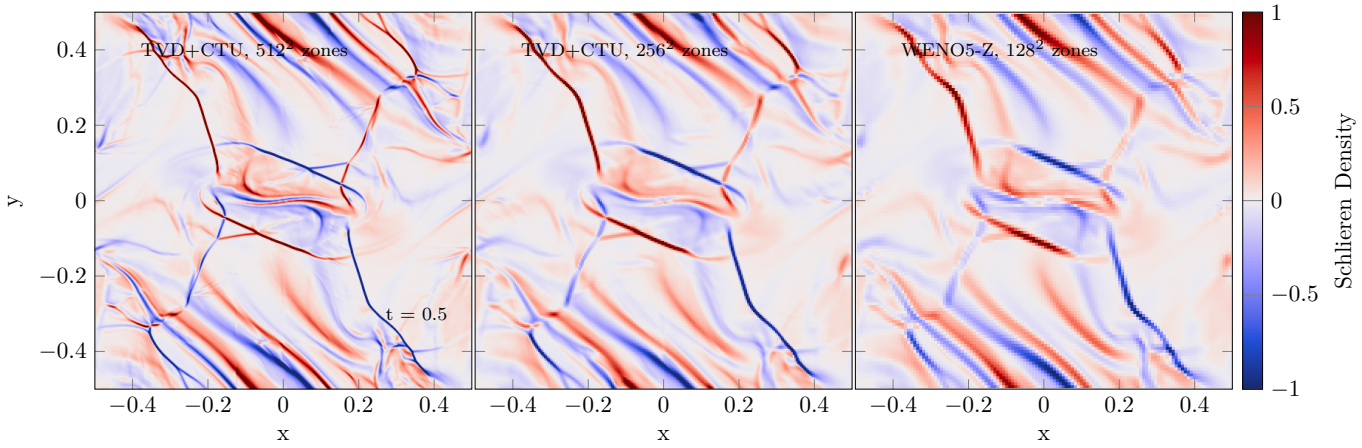

Figure 14. Schlieren image of the density in the OrszagTang vortex test at $t = 0.5$ for TVD+CTU with 512^2 zones (left), 256^2 zones (middle) and WENO5-Z with 128^2 zones (right).

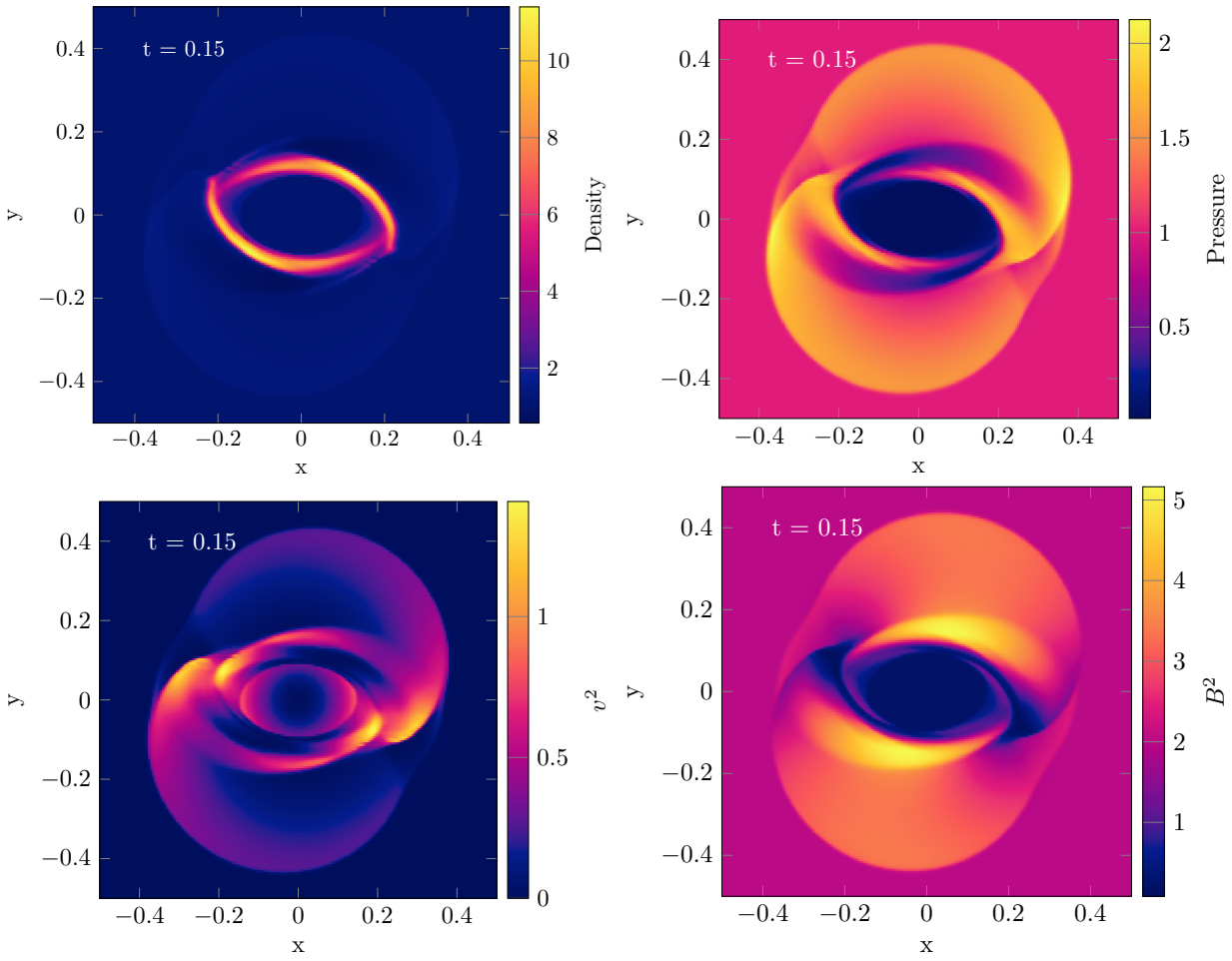

Figure 15. Images of the MHD rotor test at time $t = 0.15$ with 400^2 zones. Top-left to bottom right: density, pressure, v^2 and B^2.

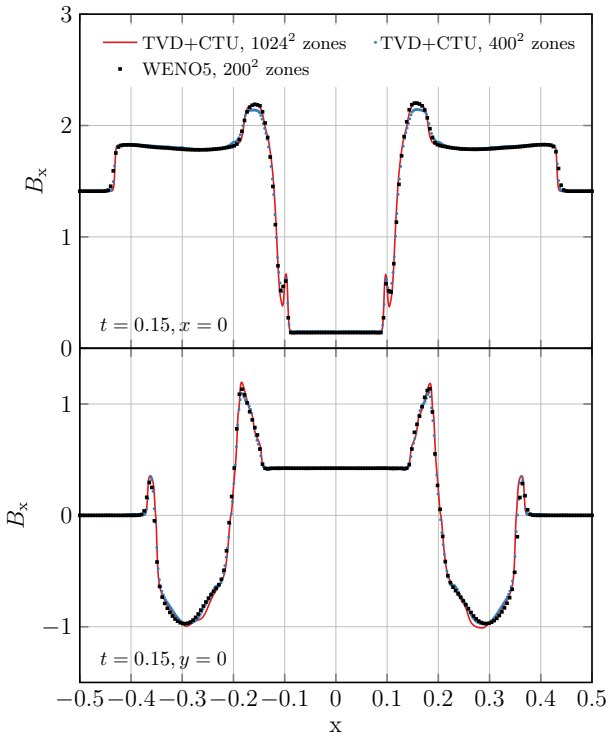

Figure 16. X-component of the magnetic field (top) and y-component of the magnetic field (bottom) in the MHD rotor test at $y = -0.1875$ (top) and $y = 0.073$ (bottom), at time $t = 0.5$. WENO with 200^2 zones as black squares, TVD+CTU with 400^2 zones as blue dots, TVD+CTU with 1024^2 zones as red line for reference.

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).

4.12. *Gresho-Vortex*

The Gresho-Vortex (Gresho & Chan 1990; Liska & Wendroff 2003; Springel 2010) is a two dimensional rotating structure in hydrodynamic equillibrium that evaluates 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.

In a 2 dimensional unit domain we set $\mathbf{U} = (1, v_x(r), v_y(r), 0, 0, 0, 0, P$ with the radius r, $v_x(r) = -v(r)\cos(\phi) + v_{\rm boost}$, $v_y(r) = -v(r)\sin(\phi)$, $\phi = \tan^{-1}(y/x)$[5] and

$$v(r) = \begin{cases} 5r & \Leftrightarrow r \leq 0.2 \\ 2 - 5r & \Leftrightarrow r < 0.4 \\ 0 & \text{otherwise} \end{cases} \tag{69}$$

$$P(r) = \begin{cases} 5 + \frac{25}{2}r^2 & \Leftrightarrow r \leq 0.2 \\ 9 + \frac{25}{2}r^2 - 20r + 4\log\left(\frac{r}{0.2}\right) & \Leftrightarrow r < 0.4 \\ 3 + 4\log(2) & \text{otherwise} \end{cases} \tag{70}$$

We run a convergence study of the test with the WENO5-Z algorithm using three different boost velocities in x-direction: $0, 1, 3$. The results are shown in figure 19 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). Our results compare quite favourably to the results obtained with lower order 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 under-resolved. Below 20^2 the L_1 error with velocity boost reduces 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 20 (left to right) angular velocity v_ϕ, pressure and vorticity $\omega(r) = \nabla \times \mathbf{v}$ over radius. We compute :

$$v(\phi) = r\frac{xv_y - yv_x}{x^2 + y^2}, \tag{71}$$

and the vorticity from the standard finite difference 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 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 under-resolved and it stands to reason that the reduced scatter in the evolved

[5] We use the Fortran ATAN2 function here.

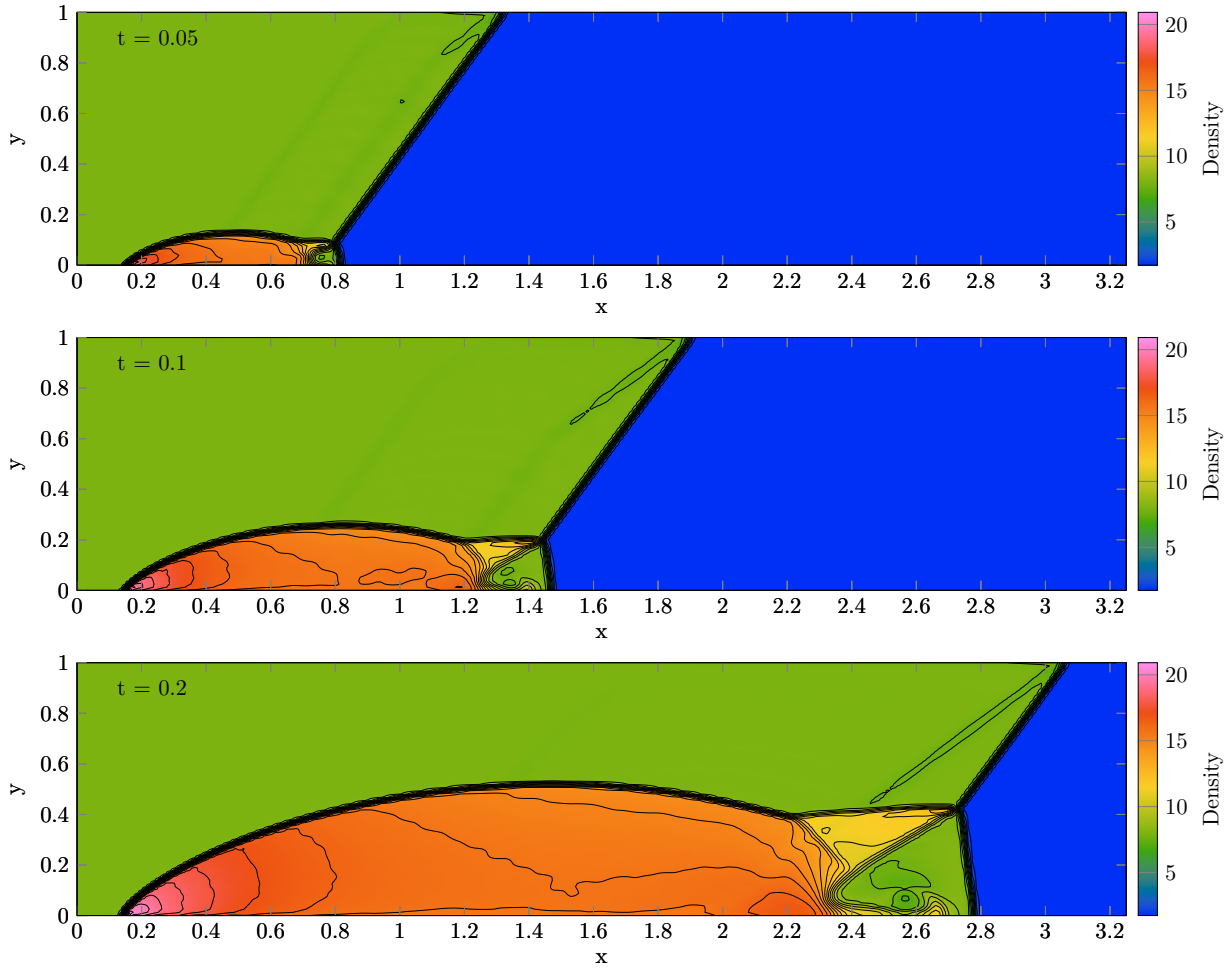

Figure 17. Density of the 2D double Mach reflection test with WENO5 at times 0.05, 0.1, 0.2 (top to bottom) and resolution 260×80 zones. We overplot 30 contours between 1.73 and 20.92, following Woodward & Colella (1984).

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). The pronounced increase in L_1 error around 32^3 zones seems to be a feature of the broad WENO5 reconstruction. Indeed Liska & Wendroff (2003) find a relatively high L_1 error in total kinetic energy for their WENO5 scheme at low resolutions.

We conclude that the stationary solution with WENO5 is very competitive with the AREPO and ATHENA solutions presented in Springel (2010). The error in the boosted solutions is comparable with second order static mesh solutions at intermediate resolutions and approaches the Lagrangian error at resolutions above 100^2 zones.

4.13. *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 2 dimensions a density square in hydrodynamic equilibrium with the surrounding medium is advected supersonically through a unit domain: $\mathbf{U} = (1, 100, 50, 0, 0, 2.4)^T$, 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. While 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 21 at time $t = 10$. We also include a WENO5-Z run with $CFL = 0.2$. The convergence rate is shown in figure 22.

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, however 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 dis-

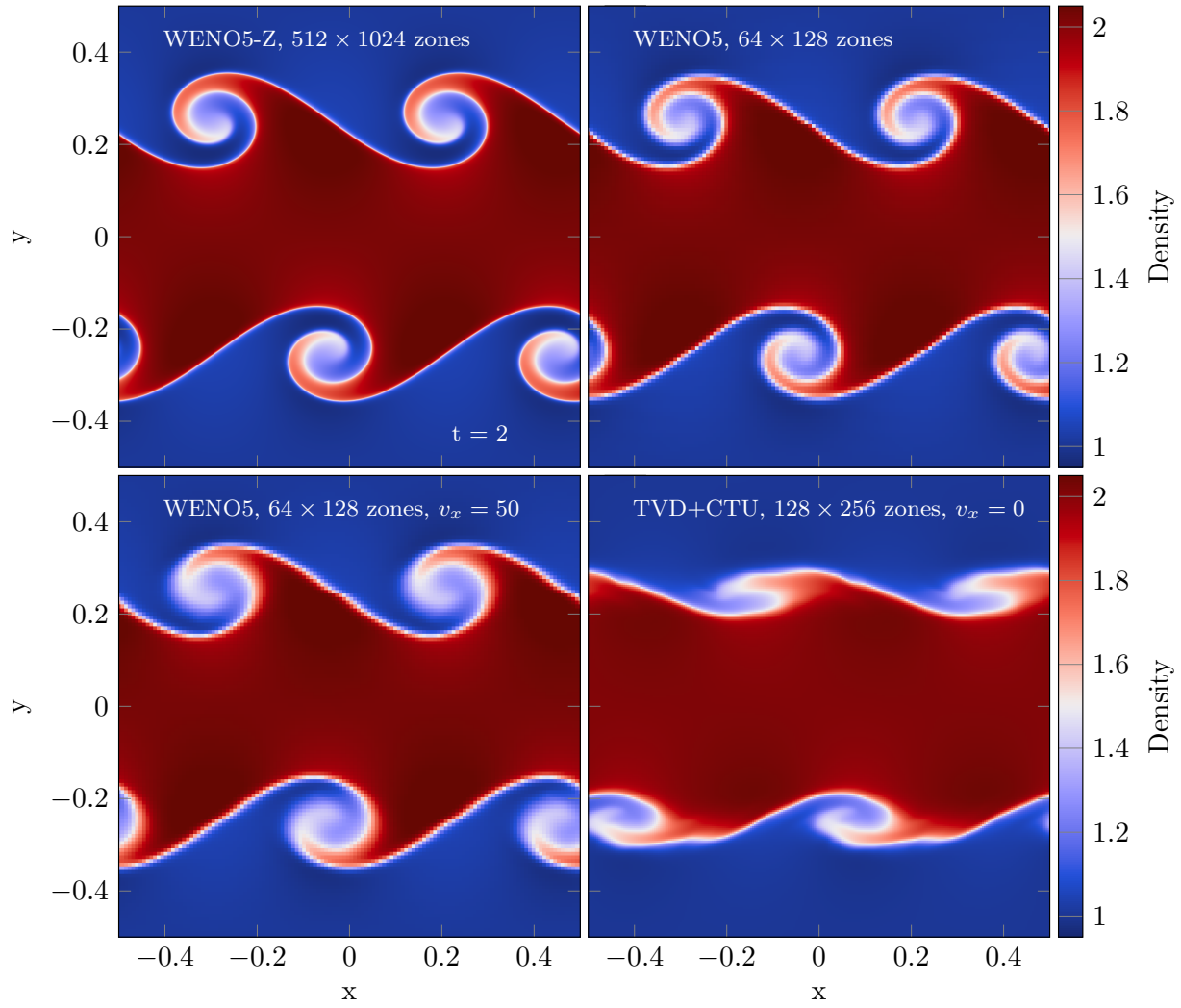

Figure 18. Density in Kelvin-Helmholtz simulations at $t = 2$. The colorbar ranges from 0.95 to 2.05. Top left to bottom right: WENO5-Z with 512×1024 zones, WENO5 with 32×64 zones, WENO5 with 32×64 zones and a velocity boost of $v_x = 10$, TVD+CTU with 128×256 zones and $v_x = 0$. We have duplicated the domain once along the x-direction to ease comparison with other publications.

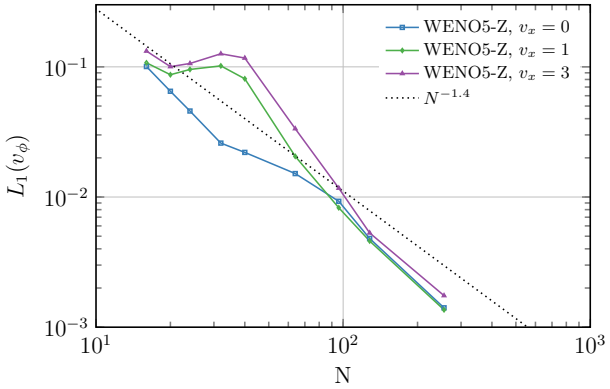

Figure 19. L_1 error norm of the angular velocity v_ϕ in the Gresho Vortex test at $t = 3$ for three WENO5 simulations with $v_x = 0$ (blue), $v_x = 1$ (green) and $v_x = 3$ (purple). The dotted line corresponds to a scaling of $N^{-1.4}$ and is normalized to approximately correspond to the results shown in Springel (2010), their figure 29.

torted. Quantitatively WENO5-Z shows an improvement in L_1 error by roughly a factor of two 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 these DG schemes are more computationally expensive than WENO5, where the time step does not change with order of the scheme. 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.

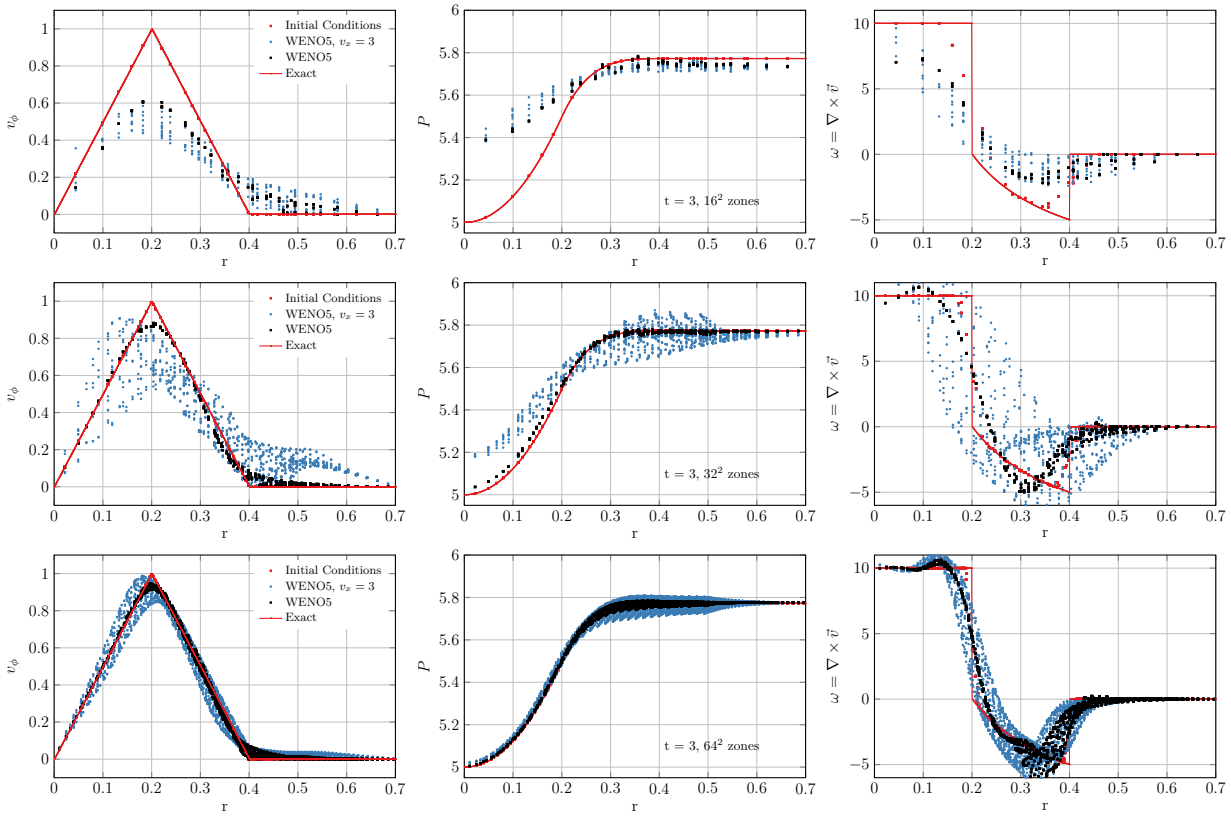

Figure 20. Left to right: Angular velocity, pressure and vorticity profile of the Gresho Vortex simulations at (top to bottom) $16^2, 32^2, 64^2$ zones. In each panel we show the exact solution as red line, the initial conditions as red dots and at $t = 3$ the WENO5 solution without boost as black dots alongside the WENO5 solution with $v_x = 3$ as blue dots.

This result suggests to run strongly advection dominated problems with WENO5-Z not 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, which approach delivers better results.

4.14. *Advection of a Magnetic Field Loop*

The field loop advection test is instrumental for the development and testing of constrained transport 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 2 dimensions, we set up a periodic domain of size $L_x \times L_y = 2 \times 1$ with $\mathbf{U} = (1, 2, 1, 0, B_x, B_y, 0, 1)^T$ and a vector potential on the interfaces with zero x and y component, but :

$$A_z = A_0 \left(r_c - r\right), \tag{72}$$

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 and interpolated to the grid at fourth order with (Jang et al. in prep.):

$$\begin{aligned} B_x = (&3B_{x,i-3,j} - 25B_{x,i-2,j} + 150B_{x,i-1,j} \\ &+ 150B_{x,i,j} - 25B_{x,i+1,j} + 3B_{x,i+2,j})/16 \end{aligned} \tag{73}$$

In figure 23, 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 normalized magnetic energy in three runs with 32, 64, and 128 zone base resolution in figure 24. We mark 98 % as 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 favourably with the results shown in Lee (2013). Even though our CT scheme is much less elaborate than theirs, the magnetic field fluxes are

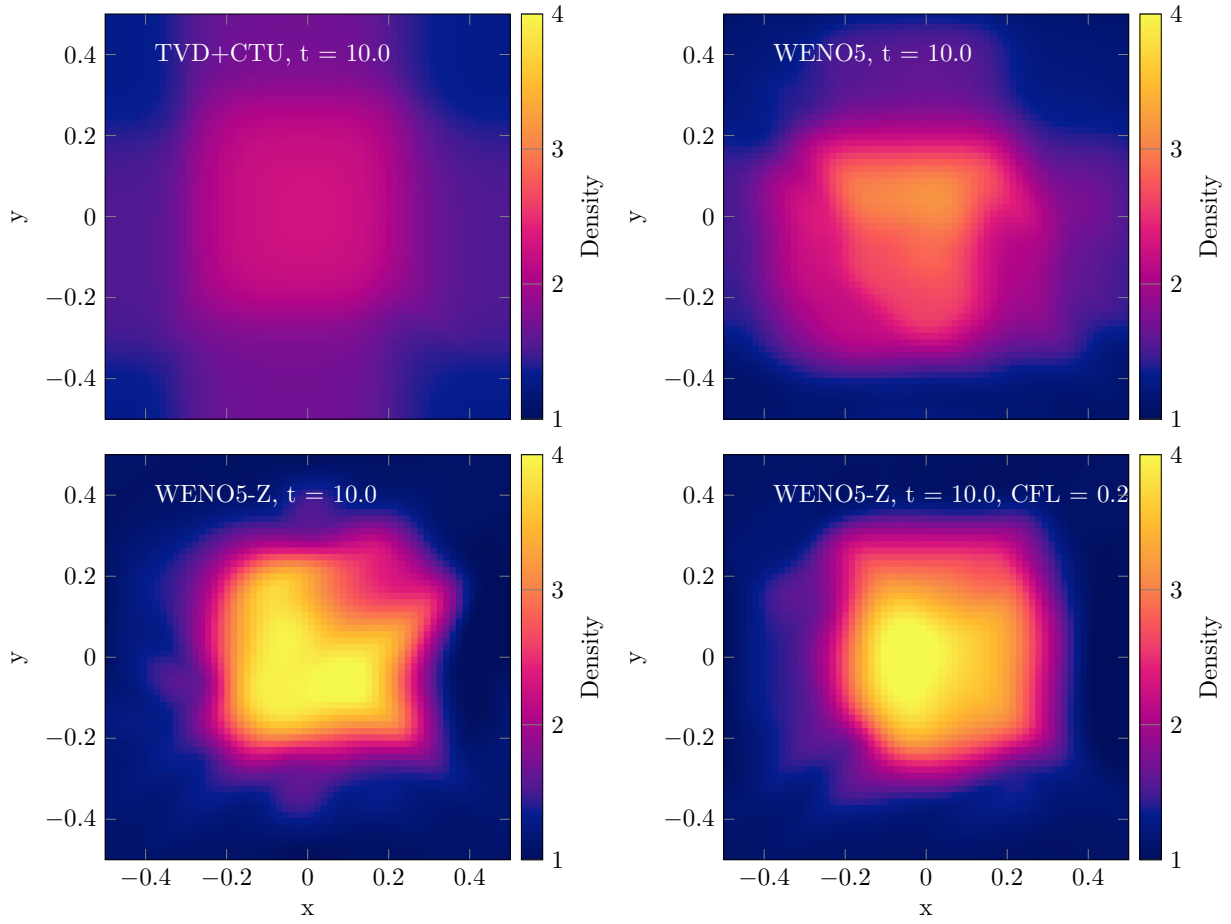

Figure 21. Density in the square advection test at $t = 10$ with 64^2 zones. Top left to bottom right: TVD+CTU, WENO5, WENO-Z, WENO-Z with CFL=0.2.

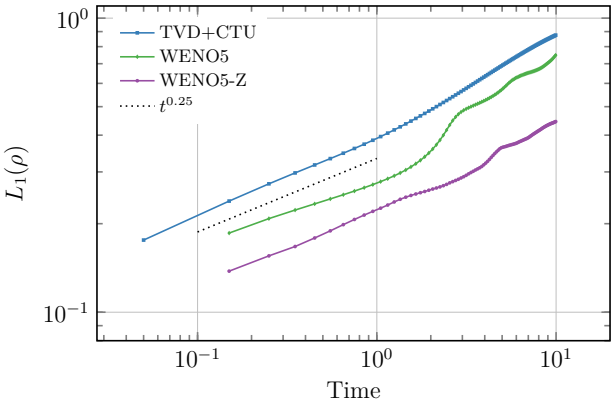

Figure 22. L_1 error norm over time in square advection test with 64^2 zones. We show TVD+CTU (blue), WENO5 (green) and WENO-Z (purple).

fifth order accurate and thus improve energy conservation. Again WENO5 roughly doubles the effective resolution of the simulation. i.e. our run 64^2 zones conserves magnetic energy roughly as 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.

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:

$$A_x = -A_0 \frac{(r_c - r^\star)}{\sqrt{5}} \tag{74}$$

$$A_y = 0 \tag{75}$$

$$A_z = -A_0 \frac{2\,(r_c - r^\star)}{\sqrt{5}}, \tag{76}$$

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

$$x^\star = \begin{cases} \frac{(2x+z)+2}{\sqrt{5}} & \Leftrightarrow x < -\sqrt{5} \\ \frac{(2x+z)-2}{\sqrt{5}} & \Leftrightarrow x > -\sqrt{5} \end{cases} \tag{77}$$

$$y^\star = y \tag{78}$$

$$z^\star = \frac{-x + 2z}{\sqrt{5}} \tag{79}$$

A rendering of the resulting magnetic energy at time $t = 2$ is shown in figure 25, i.e. after advecting the loop twice

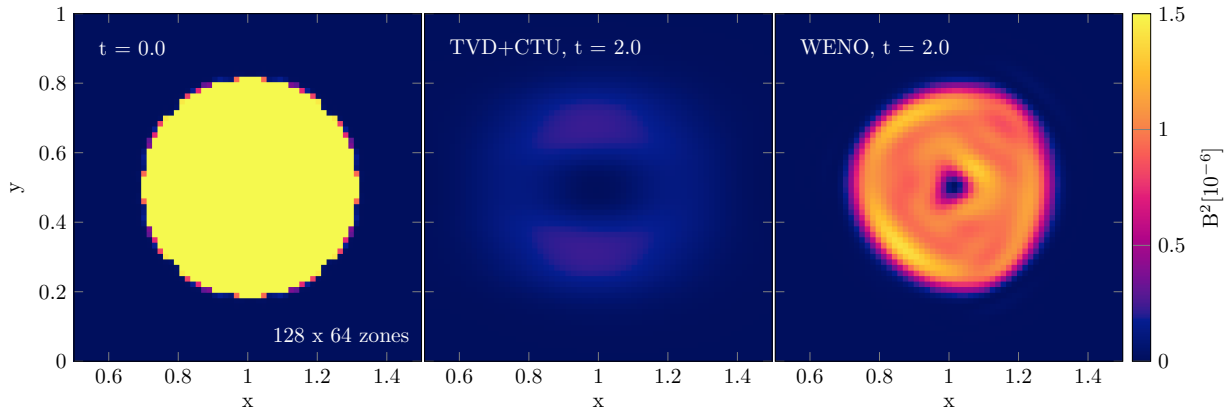

Figure 23. Magnetic field energy density B^2 at zone center in units of 10^{-6} of the magnetic field loop advection test in 2 dimensions with 128 x 64 zones. Left to right: Initial conditions, TVD+CTU at time $t = 2$ and WENO5 at time $t = 2$.

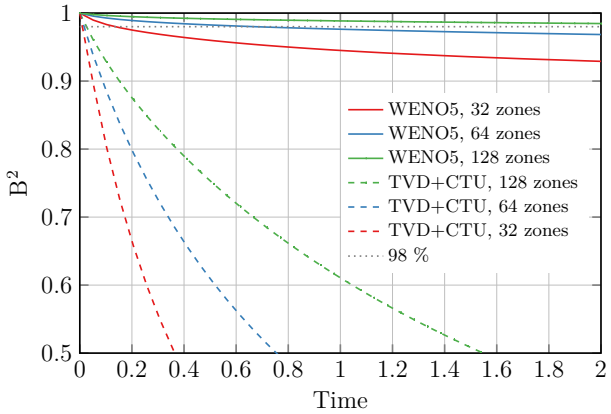

Figure 24. Total magnetic field energy density over time for the magnetic field loop test in 2 dimension. We show WENO5 at three resolutions: 64x32 zones (red), 128x64 zones (blue) and 256x128 zones (green).

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).

4.15. *Sedov-Taylor Blast Wave*

This test models the self-similar evolution of a point-like thermal detonation (von Neumann 1942; Taylor 1950; Sedov 1946). Despite its unpleasant historical context, it also is a 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 $\mathbf{U} = (1, 0, 0, 0, 10^{-8}, 0, 0, 0, 10^{-5})$ 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 $\sigma = 0.5\,\mathrm{dx}$, where dx is the zone size. Our simulation uses $\gamma = 5/3$ and $N = 64^3$ zones. The self-similar

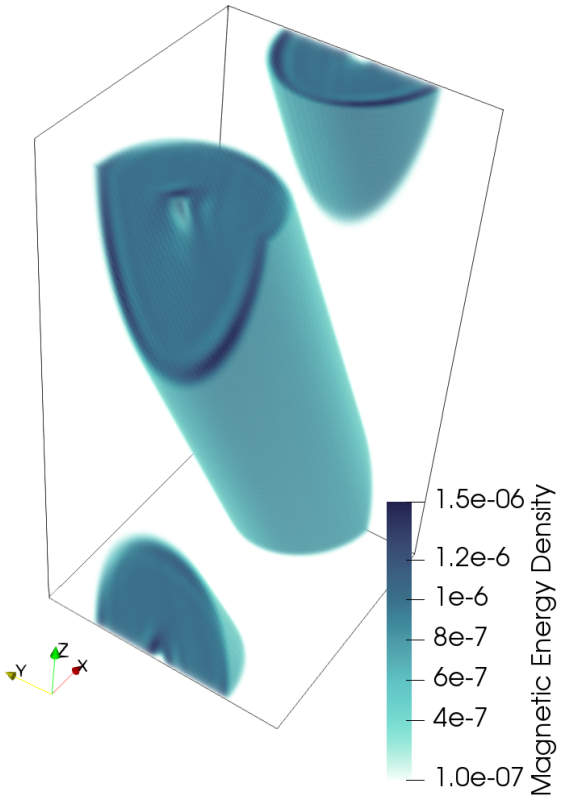

Figure 25. Magnetic field energy density B^2 of the magnetic field loop advection test in 3 dimensions with $128^2 \times 256$ zones at time $t = 2$. A threshold was imposed at 10^{-7}.

analytic solution of the problem can be found in Landau & Lifshitz (1966), with $\beta = 1.15$.

In the left of figure 26, we plot the radial profile of the shock at $t = 0.06$: analytic solution as red line and every 20th zone of the simulation as black dot. On the right of figure 26, we show a slice through the simulation at $z = 0, t = 0.06$ with colours on a log scale from 10^{-3} to 2 and contours at

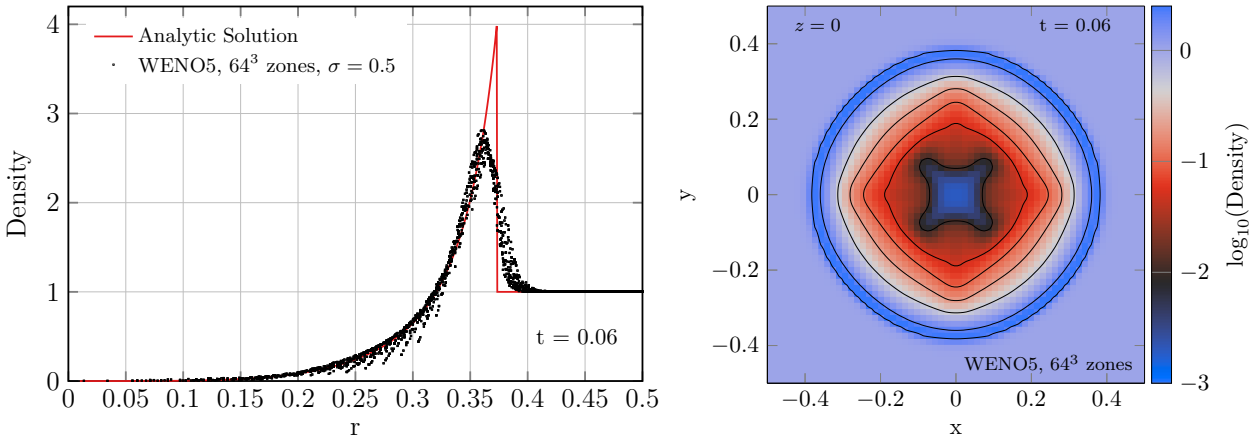

Figure 26. Left: Radial density profile of the three dimensional Sedov-Taylor test at $t = 0.06$, WENO5 simulation as black dots, analytic solution as red line. Plotted is only every 20th zone. Right: Slice through the simulation at $z = 0$. We include contours at 0.01, 0.05,0.1,0.2,0.5 and 2.

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

4.16. *MHD Blast Wave*

The magnetized blast wave simulation test 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 $\mathbf{U} = (1, 0, 0, 0, B_0/\sqrt{2}, B_0/\sqrt{2}, 0, P)^T$, 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 $200 \times 300 \times 200$ zones are shown in figure 27. 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. 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 sub-problems 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.

We run a patch size optimization study with 27 nodes to saturate MPI communication on the Aries interconnect. In figure 28, we show the performance of the WENO5 algorithm on the Broadwell architecture with 2 x 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. Colours correspond to runs with 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 behaviour differs from the TVD+CTU solver that peaks at 32^3 and shows a strong drop in performance for 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 pre-fetcher 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 4 MPI rank per node. WENO5 is thus a factor 7.5 slower than the TVD+CTU solver at the same resolution, but doubles the effective resolution. It follows

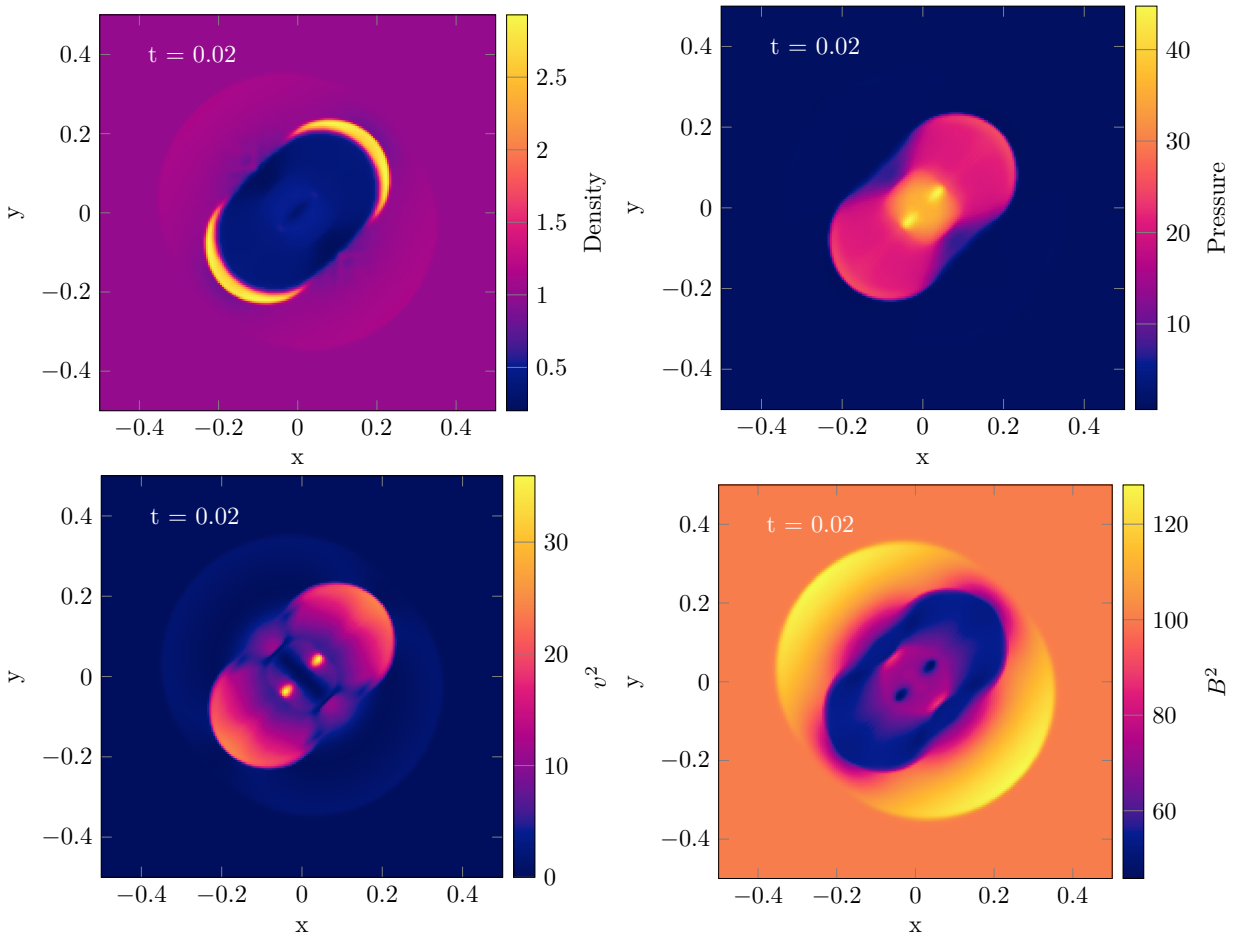

Figure 27. Slices through the MHD blast wave test at time $t = 0.02$. Shown is the center of the domain with $200 \times 300 \times 200$ zones along $z = 0$. Top-left to bottom right: density, pressure, v^2 and B^2.

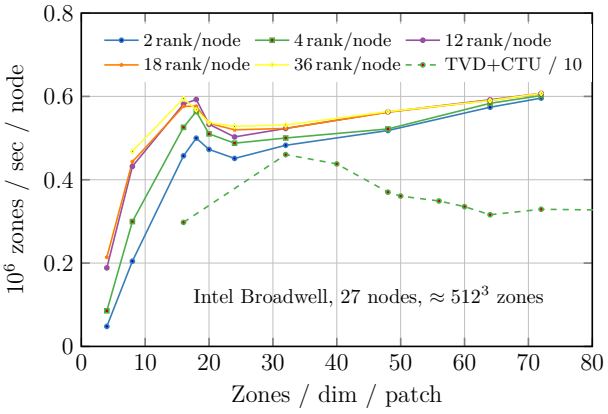

Figure 28. Compute performance in million zones per second per node over patch size (i.e. number of zones per dimension per patch) for 27 nodes on Intel Broadwell architecture. Problem size was about 512^3 zones. Blue: 2 ranks & 18 threads per node. Green: 4 ranks & 9 threads per node. Purple: 12 ranks & 3 threads per node. Orange: 18 rank & 2 theads per node. Yellow: 36 ranks & 1 thread per node.

that the WENO5 solver is more efficient than the second order scheme, because increasing the resolution by a factor 2 per dimension increases runtime of the TVD+CTU solver by roughly a factor 16: 2x per dimension and roughly 2x 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).

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 4kb 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 two (TLB thrashing). Thus it is important to run `WOMBAT` with some form of *huge pages* enabled.

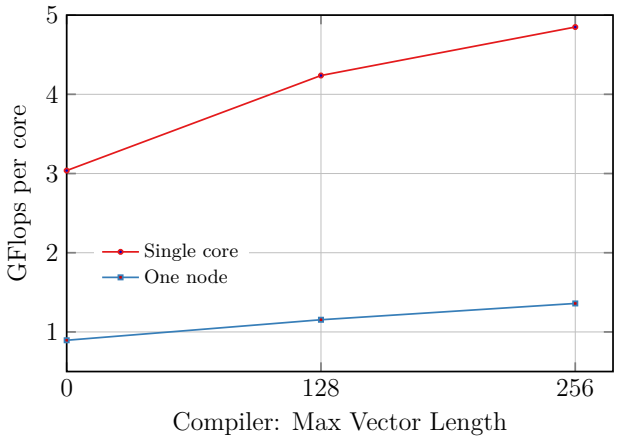

Figure 29. Performance per core in billion floating point operations per second over SIMD vector length of the WENO solver on a single Broadwell core (red) and a full Broadwell node (blue).

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 29. 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 30 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 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 512bit vectors, scalar instructions make up half 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 other constraints 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 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 exa-scale systems in the next

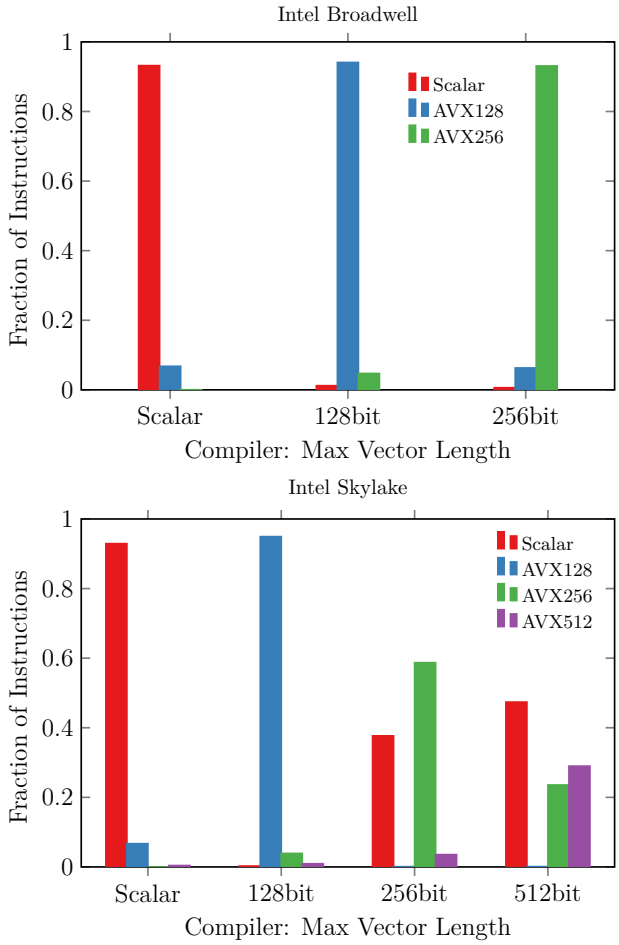

Figure 30. Fractional number of instructions generated by the Cray compiler over limit of the vector length. Top for Intel Broadwell, bottom for Intel Skylake.

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 high rate of small MPI messages in `WOMBAT` 's communication pattern. 16 MB large pages are enabled by default. We note that the problem is well balanced, i.e. that the work load per node is the same on all ranks. Thus `WOMBAT` 's load balancing capabilities are not tested here. For large problems, communication is only 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 straight forward. We also note that for large enough machines (Millions

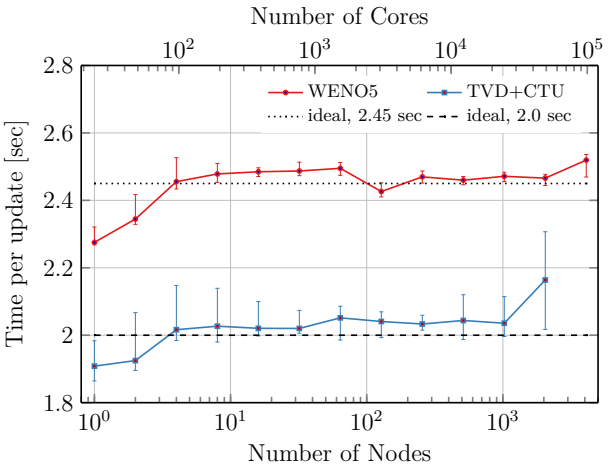

Figure 31. Time per update in seconds over number of nodes and cores (weak scaling) on HLRS Cray XC 40 "Hazel Hen". Red: WENO5 solver with 18^3 patches and $3 \times 4 \times 4$ patches per rank. Blue: TVD+CTU solver with same configuration but 32^3 patches.

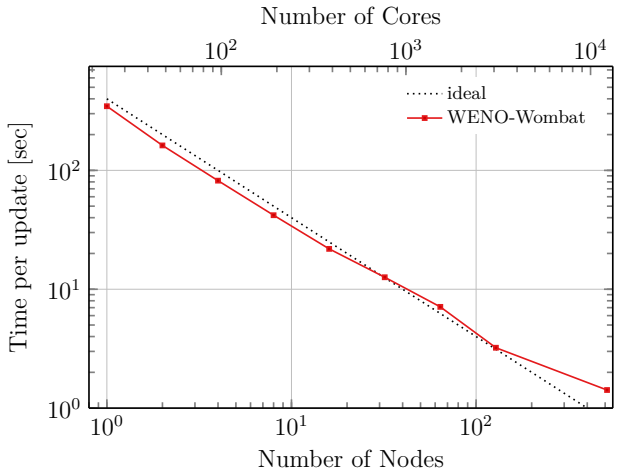

Figure 32. Strong scaling of WENO-Wombat on HLRS "Hazel Hen" as time per update over number of nodes/cores.

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 weak 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 seconds 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 number of nodes & cores is shown in figure 31, from 1 node (24 cores) to 4096 nodes (98304 cores). The minimum and maximum time per step is shown as error bars. Ideal scaling at 2.45 sec per update is shown as 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 resulting scaling is shown as a blue line. The step time is reduced by a factor of 1.2 relative to the WENO solver and the problem is a factor 5.6 larger, so the TVD+CTU solver is a factor 6.7 faster at scale than the WENO solver. But the WENO solver doubles 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.

The scaling is quite flat at about 2.45 seconds 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 note that the performance is quite competitive with recent efforts presented in Nordlund et al. (2018).

5.4. *Strong Scaling*

Strong scaling refers to the speedup 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 be dominated by the non-parallelizable part of the work eventually, 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 x 12 x 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 32 as seconds per time step over 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. 2018). The inevitable growth of computing power has lead 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 sub-grid physics are within reach for the community.

We have presented an implementation of a fifth-order Weighted-Essentially-Non-Oscillatory scheme in the numerical WOMBAT framework. We combined the classical finite difference scheme with a simple constrained transport 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:

- doubles the effective resolution compared to common second order schemes like CTU+CT or TVD+CTU.
- resolves instabilities better than a lower order scheme, due to its very low numerical diffusivity.
- is less affected by advection relative to the grid. It is competitive with Lagrangian methods in tests with moderate advection velocities.
- is more computationally efficient than a lower order scheme in three dimensions at similar solution quality.
- 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. E.g. WENO5 is significantly cheaper than third or higher order discontinuous Galerkin implementations, due to the 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 $\sim 5\%$ on 98k 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 favourable 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 (Jang et al. in prep). 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 finite volume approaches (e.g. Londrillo & del Zanna 2004; Verma et al. 2019; Felker & Stone 2018), but need to be adapted to our finite difference algorithm. Global smoothness indicators could be used to improve the robustness of the scheme (e.g. Balbs & Tadmor 2006). Worth another look is likely a low-storage fourth order Runge-Kutta integrator that reduces the memory needed to store the grid by one third. 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 super-computers reach exa-Flop performance, further doubling the effective resolution of the simulation and reducing storage and memory requirements. An extension of our WENO5 implementation would be straight forward. 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/. Data is made available wherever possible, but has to adhere to certain space limitations. A public version of WOMBAT is available under MIT license, however not yet with the WENO5 solver.

7. ACKNOWLEDGEMENTS

JD thanks Louiz de Rose and the programming environment group for 2 years of hospitality and a great work environment at Cray Inc. in Minnesota. JD thanks H. Roettgering and E. Deul for support towards the end of the project. The authors thank J. Delgado for comments on the manuscript.

We used Julia[6] and PGFPlots for post-processing and graphs shown in this document. Volume rendering was done

[6] www.julialang.org

with ParaView. Perceptually uniform colour maps were obtained from (Kovesi 2015) and colorbrewer.org.

Parts of this work were computed on the Cray development system "Kay", JD thanks Cray Inc. for continuous access to the system. Some code tests were performed on the MACH64 machine at IRA Bologna. JD thanks F. Bedosti for support with the system. TWJ was supported at the University of Minnesota by NSF grant AST. The work of H.J. and D.R. was supported by the National Research Foundation (NRF) of Korea through grants 2016R1A5A1013277 and 2017R1A2A1A05071429. This research has received funding from the People Programme (Marie Sklodowska Curie Actions) of the European Unions Eighth Framework Programme H2020 under REA grant agreement no 658912, "Cosmo Plasmas". Access to the 'Hazel Hen' at HLRS has been granted through PRACE preparatory access project "PRACE 4477".

REFERENCES

Almgren, A. S., Bell, J. B., Lijewski, M. J., Lukić, Z., & Van Andel, E. 2013, ApJ, 765, 39

Amdahl, G. M. 1967, in Proceedings of the April 18-20, 1967, Spring Joint Computer Conference, AFIPS '67 (Spring) (New York, NY, USA: ACM), 483–485

Babuka, I., & Zlmal, M. 1973, SIAM Journal on Numerical Analysis, 10, 863

Balbs, J., & Tadmor, E. 2006, SIAM Journal on Scientific Computing, 28, 533

Balsara, D. S. 2017, Living Reviews in Computational Astrophysics, 3, 2

Balsara, D. S., Garain, S., & Shu, C.-W. 2016, Journal of Computational Physics, 326, 780

Balsara, D. S., & Shu, C.-W. 2000, Journal of Computational Physics, 160, 405

Balsara, D. S., & Spicer, D. D. 1999a, Journal of Computational Physics, 149, 270

Balsara, D. S., & Spicer, D. S. 1999b, Journal of Computational Physics, 149, 270

Beck, A., Murante, G., Arth, A., et al. 2016, MNRAS, 455, 2110

Borges, R., Carmona, M., Costa, B., & Don, W. S. 2008, Journal of Computational Physics, 227, 3191

Brio, M., & Wu, C. C. 1988a, Journal of Computational Physics, 75, 500

—. 1988b, Journal of Computational Physics, 75, 400

Bryan, G. L., Norman, M. L., O'Shea, B. W., et al. 2014, ApJS, 211, 19

Colella, P., & Woodward, P. R. 1984, Journal of Computational Physics, 54, 174

Dai, W., & Woodward, P. R. 1998, ApJ, 494, 317

De Pontieu, B., McIntosh, S. W., Carlsson, M., et al. 2007, Science, 318, 1574

Del Zanna, L., Velli, M., & Londrillo, P. 2001, A&A, 367, 705

Donnert, J., Jang, H., Mendygral, P., et al. 2018, Galaxies, 6, 104

Einfeldt, B. 1988, 25, 294

Einfeldt, B., Roe, P. L., Munz, C. D., & Sjogreen, B. 1991, Journal of Computational Physics, 92, 273

Evans, C. R., & Hawley, J. F. 1988, ApJ, 332, 659

Felker, K. G., & Stone, J. M. 2018, Journal of Computational Physics, 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, Journal of Computational Physics, 205, 509

—. 2008, Journal of Computational Physics, 227, 4123

Godunov, S. K. 1959, Mat. Sb. (N.S.), 47, 271

Goldstein, M. L. 1978, ApJ, 219, 700

Greenough, J. A., & Rider, W. J. 2004, J. Comput. Phys., 196, 259

Gresho, P. M., & Chan, S. T. 1990, International Journal for Numerical Methods in Fluids, 11, 621

Guillet, T., Pakmor, R., Springel, V., Chandrashekar, P., & Klingenberg, C. 2018, ArXiv e-prints, arXiv:1806.02343

Guo, L.-d., & Zhang, L. 2004, Journal of Visualization, 7, 225

Harten, A. 1983, Journal of Computational Physics, 49, 357

Harten, A., Engquist, B., Osher, S., & Chakravarthy, S. R. 1987, Journal of Computational Physics, 71, 231

Helzel, C., Rossmanith, J. A., & Taetz, B. 2011, Journal of Computational Physics, 230, 3803

Henrick, A. K., Aslam, T. D., & Powers, J. M. 2005, Journal of Computational Physics, 207, 542

Hooke, R. 1665, Micrographia, Observation LVIII, 217219

Hopkins, P. F. 2013, MNRAS, 428, 2840

—. 2015, MNRAS, 450, 53

Hopkins, P. F., & Raives, M. J. 2016, MNRAS, 455, 51

Jeffrey, A., & Taniuti, T. 1964, Non-linear wave propagation

Jiang, G.-S., & Shu, C.-W. 1996, Journal of Computational Physics, 126, 202

Jiang, G.-S., & Wu, C.-c. 1999, Journal of Computational Physics, 150, 561

Kovesi, P. 2015, ArXiv e-prints, arXiv:1509.03700

Kravtsov, A. V., Klypin, A. A., & Khokhlov, A. M. 1997, ApJS, 111, 73

Kulsrud, R. M., & Ostriker, E. C. 2006, Physics Today, 59, 010000

Landau, L. D., & Lifshitz, E. M. 1966, Hydrodynamik

Lax, P. D. 1954, Communications on Pure and Applied Mathematics, 7, 159
Lecoanet, D., McCourt, M., Quataert, E., et al. 2016, MNRAS, 455, 4274
Lee, D. 2013, Journal of Computational Physics, 243, 269
LeVeque, R. 2002, Finite volume methods for hyperbolic problems , Cambridge Texts in applied mathematics (Cambridge University Press)
Liska, R., & Wendroff, B. 2003, SIAM Journal on Scientific Computing, 25, 995
Liu, X.-D., Osher, S., & Chan, T. 1994, Journal of Computational Physics, 115, 200
Londrillo, P., & Del Zanna, L. 2000, ApJ, 530, 508
Londrillo, P., & del Zanna, L. 2004, Journal of Computational Physics, 195, 17
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, Computing in Science Engineering, 19, 44
Mignone, A., Bodo, G., Massaglia, S., et al. 2007, ApJS, 170, 228
Mignone, A., Tzeferacos, P., & Bodo, G. 2010, Journal of Computational Physics, 229, 5896
Miura, A., & Pritchett, P. L. 1982, J. Geophys. Res., 87, 7431
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, Journal of Fluid Mechanics, 90, 128
Parker, E. N. 1958, ApJ, 128, 664
Richardson, L. F. 1922, Quarterly Journal of the Royal Meteorological Society, 48, 282
Robertson, B. E., Kravtsov, A. V., Gnedin, N. Y., Abel, T., & Rudd, D. H. 2010, MNRAS, 401, 2463
Roe, P. 1997, Journal of Computational Physics, 135, 250
Roe, P. L. 1981, J. COMP. PHYS, 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, U.S.S.R. 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, J. Appl. Math. Mech., 10
Sheardown, A., Roediger, E., Su, Y., et al. 2018, ApJ, 865, 118
Shu, C. 1988, SIAM Journal on Scientific and Statistical Computing, 9, 1073
—. 2009, SIAM Review, 51, 82
Shu, C.-W. 1998, in Advanced Numerical Approximate of Nonlinear Hyperbolic Equations. Lecture notes in Mathematics 1697, 1997 C.I.M.E. course in Cetraro, Italy, June 1997 (A. Quarteroni ed.) (Springer Verlag), 285
Shu, C.-W., & Osher, S. 1988, Journal of Computational Physics, 77, 439
—. 1989, Journal of Computational Physics, 83, 32
Sijacki, D., Vogelsberger, M., Kereš, D., Springel, V., & Hernquist, L. 2012, MNRAS, 424, 2999
Spiteri, R., & Ruuth, S. 2002, SIAM Journal on Numerical Analysis, 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, Proceedings of the Royal Society of London Series A, 201, 159
Teyssier, R. 2002, A&A, 385, 337
Tóth, G. 2000, Journal of Computational Physics, 161, 605
van Leer, B. 1979, Journal of Computational Physics, 32, 101
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 Document AM-9, National Defense Research Council, Division B, Washington, DC, USA
Woodward, P., & Colella, P. 1984, Journal of Computational Physics, 54, 115
Zhang, Y.-T., Shi, J., Shu, C.-W., & Zhou, Y. 2003, PhRvE, 68, 046709

APPENDIX

A. 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 $L_i R^i = \mathsf{I}$ (Jeffrey & Taniuti 1964). The eigenvalues by component m are $\lambda(1,7) = v_x \mp c_{\rm fast}, \lambda(2,6) = v_x \mp c_{\rm A}, \lambda(3,5) = v_x \mp c_{\rm slow}, \lambda(4) = v_x$. For MHD, degeneraries occur in the case of vanishing magnetic field components (Brio & Wu 1988b), so we set:

$$(\beta_y, \beta_z) = \begin{cases} \frac{(B_y, B_z)}{\sqrt{B_x^2 + B_z^2}} & \text{if } B_y^2 + B_z^2 > \epsilon_B \\ \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right) & \text{otherwise} \end{cases} \tag{A1}$$

$$\mathrm{sgn}(B_x) = \begin{cases} 1 & \text{if } B_x \geq \epsilon_B \\ -1 & \text{otherwise} \end{cases} \tag{A2}$$

$$(\alpha_f, \alpha_s) = \begin{cases} \frac{\sqrt{c_s^2 - c_{\rm slow}^2}\cdot\sqrt{c_{\rm fast}^2 - c_s^2}}{\sqrt{c_{\rm fast}^2 - c_{\rm slow}^2}} & \text{if } \sqrt{c_{\rm fast}^2 - c_{\rm slow}^2} \geq \epsilon_B \\ (1,1) & \text{otherwise} \end{cases} \tag{A3}$$

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 mode) in the calculation of equation A3. This can lead to noise in the calculation that can be exposed e.g. in the implosion test, section 4.7. It is advisable to explicitly check for the eigenvalue degeneracy in the code and set equation A3 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), Jang et al. in prep. With

$$\gamma_1 = \frac{\gamma - 1}{2} \qquad \gamma_2 = \frac{\gamma - 2}{\gamma - 1} \qquad \tau = \frac{\gamma - 1}{c_s^2} \tag{A4}$$

and

$$\Gamma_f = \alpha_f c_{\rm fast} v_x - \alpha_s c_{\rm slow} \mathrm{sgn}(B_x) \left(\beta_y v_y + \beta_z v_z\right) \tag{A5}$$
$$\Gamma_a = \mathrm{sgn}(B_x) \left(\beta_z v_y - \beta_y v_z\right) \tag{A6}$$
$$\Gamma_s = \alpha_s c_{\rm slow} v_x + \alpha_f c_{\rm fast} \mathrm{sgn}(B_x) \left(\beta_y v_y + \beta_z v_z\right), \tag{A7}$$

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

$$L_1(m) = \frac{1}{2c_s^2}\left[\alpha_f\left(\gamma_1 v^2 \pm \Gamma_f\right)\right] \qquad R^1(m) = \alpha_f \tag{A8}$$
$$L_2(m) = (1-\gamma)\alpha_f v_x \mp \alpha_f v_y \pm c_{\rm slow}\alpha_s\beta_y \mathrm{sgn}(B_x) \qquad R^2(m) = \alpha_f(v_x \mp c_{\rm fast}) \tag{A9}$$
$$L_3(m) = (1-\gamma)\alpha_f v_y \pm c_{\rm slow}\alpha_s\beta_y \mathrm{sgn}(B_x) \qquad R^3(m) = \alpha_f v_y \pm c_{\rm slow}\alpha_s\beta_y \mathrm{sgn}(B_x) \tag{A10}$$
$$L_4(m) = (1-\gamma)\alpha_f v_z \pm c_{\rm slow}\alpha_s\beta_z \mathrm{sgn}(B_x) \qquad R^4(m) = \alpha_f v_z \pm c_{\rm slow}\alpha_s\beta_z \mathrm{sgn}(B_x) \tag{A11}$$
$$L_5(m) = (1-\gamma)\alpha_f B_y - \sqrt{\rho} c_s \alpha_s \beta_y \qquad R^5(m) = c_s \alpha_s \beta_y / \sqrt{\rho} \tag{A12}$$
$$L_6(m) = (1-\gamma)\alpha_f B_z - \sqrt{\rho} c_s \alpha_s \beta_z \qquad R^6(m) = c_s \alpha_s \beta_z / \sqrt{\rho} \tag{A13}$$
$$L_7(m) = (\gamma - 1)\alpha_f \qquad R^7(m) = \alpha_f \left(\frac{1}{2}v^2 + c_{\rm fast}^2 - \gamma_2 c_s^2\right) \mp \Gamma_f \tag{A14}$$

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

$$L_1(m) = \frac{1}{2}\Gamma_a \qquad R^1(m) = 0 \tag{A15}$$

$$L_2(m) = 0 \qquad R^2(m) = 0 \tag{A16}$$

$$L_3(m) = -\frac{1}{2}\beta_z \mathrm{sgn}(B_x) \qquad R^3(m) = -\beta_z \mathrm{sgn}(B_x) \tag{A17}$$

$$L_4(m) = \frac{1}{2}\beta_y \mathrm{sgn}(B_x) \qquad R^4(m) = \beta_y \mathrm{sgn}(B_x) \tag{A18}$$

$$L_5(m) = \mp\frac{1}{2}\sqrt{\rho}\beta_z \qquad R^5(m) = \mp\beta_z/\sqrt{\rho} \tag{A19}$$

$$L_6(m) = \pm\frac{1}{2}\sqrt{\rho}\beta_y \qquad R^6(m) = \pm\beta_y/\sqrt{\rho} \tag{A20}$$

$$L_7(m) = 0 \qquad R^7(m) = -\Gamma_a \tag{A21}$$

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

$$L_1(m) = \frac{1}{2c_s^2}\left(\gamma_1 \alpha_s v^2 \pm \Gamma_s\right) \qquad R^1(m) = \alpha_s \tag{A22}$$

$$L_2(m) = \frac{1}{2c_s^2}\left((1-\gamma)\alpha_s v_x \mp \alpha_s c_{\mathrm{slow}}\right) \qquad R^2(m) = \alpha_s(v_x \mp c_{\mathrm{slow}}) \tag{A23}$$

$$L_3(m) = \frac{1}{2c_s^2}\left((1-\gamma)\alpha_s v_y \mp c_{\mathrm{fast}}\alpha_f \beta_y \mathrm{sgn}(B_x)\right) \qquad R^3(m) = \alpha_s v_y \mp c_{\mathrm{fast}}\alpha_f \beta_y \mathrm{sgn}(B_x) \tag{A24}$$

$$L_4(m) = \frac{1}{2c_s^2}\left((1-\gamma)\alpha_s v_z \mp c_{\mathrm{fast}}\alpha_f \beta_z \mathrm{sgn}(B_x)\right) \qquad R^4(m) = \alpha_s v_z \mp c_{\mathrm{fast}}\alpha_f \beta_z \mathrm{sgn}(B_x) \tag{A25}$$

$$L_5(m) = \frac{1}{2c_s^2}\left((1-\gamma)\alpha_s B_y - \sqrt{\rho}c_s \alpha_f \beta_y\right) \qquad R^5(m) = -c_s \alpha_f \beta_y/\sqrt{\rho} \tag{A26}$$

$$L_6(m) = \frac{1}{2c_s^2}\left((1-\gamma)\alpha_s B_z - \sqrt{\rho}c_s \alpha_f \beta_z\right) \qquad R^6(m) = -c_s \alpha_f \beta_z/\sqrt{\rho} \tag{A27}$$

$$L_7(m) = \frac{1}{2c_s^2}\left((\gamma-1)\alpha_s\right) \qquad R^7(m) = \alpha_s\left(\frac{1}{2}v^2 + c_{\mathrm{slow}}^2 - \gamma_2 c_s^2\right) \mp \Gamma_s \tag{A28}$$

$$\tag{A29}$$

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

$$L_1(m) = 1 - \frac{1}{2}\tau v^2 \qquad R^1(m) = 1 \tag{A30}$$

$$L_2(m) = \tau v_x \qquad R^2(m) = v_x \tag{A31}$$

$$L_3(m) = \tau v_y \qquad R^3(m) = v_y \tag{A32}$$

$$L_4(m) = \tau v_z \qquad R^4(m) = v_z \tag{A33}$$

$$L_5(m) = \tau B_y \qquad R^5(m) = 0 \tag{A34}$$

$$L_6(m) = \tau B_z \qquad R^6(m) = 0 \tag{A35}$$

$$L_7(m) = -\tau \qquad R^7(m) = \frac{1}{2} \tag{A36}$$

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