Universität zu Köln

Discontinuous Galerkin Spectral Element Methods for Astrophysical Flows in Multi-physics Applications

Inaugural-Dissertation

zur

Erlangung des Doktorgrades

der Mathematisch-Naturwissenschaftlichen Fakultät

der Universität zu Köln

vorgelegt von

Johannes Markert

aus Stollberg/Sachsen

Köln, 2022

Berichterstatter:	Prof. Dr.-Ing. Gregor Gassner
	Prof. Dr.-Ing. Michael Dumbser
Tag der mündlichen Prüfung:	30.05.2022

Preface

This thesis was developed during my work as an academic employee at the Mathematical Institute of the University of Cologne. My scientific endeavor was funded by the Klaus-Tschira Stiftung via the project "DG^{2}RAV" and by the European Research Council through the ERC Starting Grant "An Exascale aware and Un-crashable Space-Time-Adaptive Discontinuous Spectral Element Solver for Non-Linear Conservation Laws (EXTREME, project no. 71448)".

I thank my doctoral supervisor Prof. Dr.-Ing. Gregor Gassner for the excellent working conditions in his research group, especially for the generous scientific freedom I was granted under his supervision. Many thanks to all my colleagues in our research group for the welcoming atmosphere, the fruitful scientific discussions and our fun times together. A special thank goes to my former colleague Dr. Gero Schnücke for putting me under his wings and for his friendship.

Furthermore, I cordially thank my scientific mentor Prof. Dr. Stefanie Walch-Gassner for her valuable advice and her firm push to encourage me in exploring all the potentials and limits of high order methods in challenging astrophysics simulations. She welcomed me into her research group at the I. Physics Institute at the University of Cologne and made it possible to work on a joint project at the exciting interface between numerical analysis and theoretical astrophysics.

I thank Prof. Dr.-Ing. Michael Dumbser for being the second assessor of this thesis.

Last but not least, I thank my family for their continued encouragement, patience, love, and support through this long journey.

Köln, 20.03.2022 Johannes Markert

Abstract

In engineering applications, discontinuous Galerkin methods (DG) have been proven to be a powerful and flexible class of high order methods for problems in computational fluid dynamics. However, the potential benefits of DG for applications in astrophysical contexts is still relatively unexplored in its entirety. To this day, a decent number of studies surveying DG for astrophysical flows have been conducted. But the adoption of DG by the astrophysics community is just beginning to gain traction and integration of DG into established, multi-physics simulation frameworks for comprehensive astrophysical modeling is still lacking. It is our firm believe, that the full potential of novel approaches for numerically solving the fluid equations only shows under the pressure of real-world simulations with all aspects of multi-physics, challenging flow configurations, resolution and runtime constraints, and efficiency metrics on high-performance systems involved. Thus, we see the pressing need to propel DG from the well-trodden path of cataloguing test results under "optimal" laboratory conditions towards the harsh and unforgiving environment of large-scale astrophysics simulations.

Consequently, the core of this work is the development and deployment of a robust DG scheme solving the ideal magneto-hydrodynamics equations with multiple species on three-dimensional Cartesian grids with adaptive mesh refinement. We chose to implement DG within the venerable simulation framework FLASH, with a specific focus on multi-physics problems in astrophysics. This entails modifications of the vanilla DG scheme to make it fit seamlessly within FLASH in such a way that all other physics modules can be naturally coupled without additional implementation overhead. A key ingredient is that our DG scheme uses mean value data organized into blocks - the central data structure in FLASH. Having the opportunity to work on mean values, allows us to rely on a rock-solid, monotone Finite Volume (FV) scheme as "backup" whenever the high order DG method fails in cases when the flow gets too harsh. Finding ways to combine the two schemes in a fail-safe manner without loosing primary conservation while still maintaining high order accuracy for smooth, well-resolved flows involves a series of careful considerations, which we document in this thesis. The result of our work is a novel shock capturing scheme - a hybrid between FV and DG - with smooth transitions between low and high order fluxes according to solution smoothness estimators.

We present extensive validations and test cases, specifically its interaction with multi-physics modules in FLASH such as (self-)gravity and radiative transfer. We also in-

vestigate the benefits and pitfalls of integrating end-to-end entropy stability into our numerical scheme, with special focus on highly compressible turbulent flows and shocks. Our implementation of DG in FLASH allows us to conduct preliminary yet comprehensive astrophysics simulations proving that our new solver is ready for assessments and investigations by the astrophysics community.

Contents

Symbols

Acronyms

1 Introduction **1**

1.1 Background 1
1.2 State of DG in Astrophysics 6
1.3 Research Mission & Structure of the Thesis 8
1.4 Peer-reviewed Publications 10
1.5 Open Source Codes 11

2 Hyperbolic Conservation Laws **12**

2.1 Introduction 12
2.2 Systems of Conservation Laws 15
2.3 Weak Formulation 16
2.4 Entropy Condition 18
2.5 Riemann Problem 22

3 Astrophysical Model **28**

3.1 Introduction 28
3.2 Interstellar Medium 31
3.3 Compressible Euler Equations 32
3.4 Ideal Magneto-hydrodynamic Equations 37
3.5 Shocks 47
3.6 Multi-species Fluids 52
3.7 Gravity 55

3.8 Radiative Transfer . . . 59
3.9 Final Remarks . . . 62

4 Numerical Scheme **64**
4.1 Introduction . . . 64
4.2 Shock Capturing . . . 66
4.3 Anti-Aliasing . . . 71
4.4 Time Integration . . . 72
4.5 Finite Volume Scheme . . . 74
4.5.1 Building Blocks . . . 75
4.5.2 CFL Condition . . . 78
4.5.3 Experimental Order of Convergence . . . 79
4.5.4 Entropy Conservation . . . 80
4.5.5 Cell Entropy Production Rate . . . 86
4.5.6 Satisfying the Cell Entropy Inequality . . . 88
4.6 Discontinuous Galerkin Spectral Element Method . . . 92
4.6.1 Building Blocks . . . 93
4.6.2 CFL Condition . . . 101
4.6.3 Experimental Order of Convergence . . . 101
4.6.4 Element Entropy Production Rate . . . 103
4.6.5 Satisfying the Element Entropy Inequality . . . 106
4.7 Convex Blending Scheme . . . 114
4.7.1 Building Blocks . . . 115
4.7.2 CFL Condition . . . 121
4.7.3 Experimental Order of Convergence . . . 121
4.7.4 Element Entropy Production Rate . . . 122
4.7.5 Satisfying the Element Entropy Inequality . . . 122
4.7.6 Enforcing Density and Pressure Positivity . . . 128
4.8 Final Remarks . . . 128

5 Nemo - a modular fluid dynamics code for prototyping **131**
5.1 Introduction . . . 131
5.2 Hardware & Compiler Stack . . . 133
5.3 Adaptive Mesh Refinement . . . 134
5.4 Hybrid Parallelization . . . 137

5.5 Runtime Performance & Scaling Behavior . . . 140

6 Implementing DG in FLASH **148**
6.1 Introduction . . . 148
6.2 Implementation Details . . . 150
6.3 Numerical Results . . . 153
6.3.1 Experimental Order of Convergence . . . 153
6.3.2 Divergence Control . . . 155
6.3.3 Shock Problems . . . 160
6.3.4 Coupling to Gravity . . . 181
6.3.5 Coupling to TreeRay . . . 183
6.4 Final Remarks . . . 187

7 Simulations **189**
7.1 Introduction . . . 189
7.2 Young Supernova Remnant . . . 189
7.3 Differentially Rotating MHD Torus . . . 197
7.4 Expansion of HII Region into Fractal Medium . . . 202
7.5 Molecular Cloud in Hot Galactic Wind . . . 206

8 Conclusion & Outlook **210**
8.1 Recapitulation . . . 210
8.2 Revisiting the Scientific Objectives . . . 212
8.3 Follow-up Research . . . 215

A Appendix **217**
A.1 Satisfying the Element Entropy Condition via Convex Blending (proof-of-concept) . . . 217
A.2 3D MHD Blast run by Bouchut5 . . . 219
A.3 Young Supernova Remnant from Nemo . . . 220
A.4 Comparative Study for the MHD Torus Setup . . . 221
A.5 Turbulent Expanding HII Region with Bouchut5 . . . 223
A.6 Stellar Feedback into Turbulent ISM . . . 224
A.7 Cloud in Hot Galactic Wind with Bouchut5 . . . 225

Bibliography **260**

List of Figures **260**

List of Tables **271**

Curriculum Vitae **273**

Erklärung **273**

Symbols

m	number of state variables
n	number of nodes
N	number of cells
Q	number of blocks/elements
$d = 1, 2, 3$	index of space dimension
q	block/element index
i, j, k	cell/node index
t	time variable
$\vec{x} = (x, y, z)^T$	vector of space variables
$\vec{n} = (n_1, n_2, n_3)^T$	orthonormal surface vector
Δt, Δx	timestep, spatial distance measure of grid
∂_t, $\frac{d}{dt}$	partial time derivative, total time derivative
∂_{x_d}	partial space derivative in direction $d = 1, 2, 3$
$\delta_{i,j}$	Kronecker delta
$\mathbb{1} = \mathrm{diag}(1, 1, \ldots, 1)$	identity matrix
$\nabla = \partial_{\vec{x}}$	gradient of space variables
$\vec{a} \cdot \vec{b}$, $\vec{a}^2$	scalar product of two space vectors
$\vec{a} \otimes \vec{b}$	outer product of two space vectors
$\partial_{\boldsymbol{u}}$	gradient of state variables
$\boldsymbol{a} \cdot \boldsymbol{b}$, $\boldsymbol{a}^2$	scalar product of two state vectors
$\{\!\{(\cdot)\}\!\}$, $\{\!\{(\cdot)\}\!\}_{\ln}$, $[\![(\cdot)]\!]$	mean operator, logarithmic mean operator, jump operator
Ω, Ω_q	computational domain, computational sub-domain
$\lvert\Omega_q\rvert$, $\partial\Omega_q$	volume of Ω_q, closed surface around Ω_q
$\boldsymbol{u}$	vector of state variables
$\boldsymbol{u}^+, \boldsymbol{u}^-$	left (+) states, right (-) states
$\dot{\boldsymbol{u}}$	right hand side
$\boldsymbol{f}$	flux function
$\boldsymbol{f}^*$, $\boldsymbol{f}^\#$	numerical two-point flux, entropy conservative two-point flux
$\overline{\boldsymbol{u}}$, $\widetilde{\boldsymbol{u}}$	state variables in mean space and nodal space
$\widetilde{\widetilde{\boldsymbol{u}}}$	entropy projected state variables in nodal space

Π	set of physically permissible states
$\boldsymbol{J}$	flux Jacobian
λ	eigenvalue of flux Jacobian
$\boldsymbol{r}$	eigenvector of flux Jacobian
$\mathcal{S}$	entropy function
$\mathcal{F}$	entropy flux
$\boldsymbol{w}$	vector of entropy variables
$\Delta\mathcal{S}$, $\Delta\dot{\mathcal{S}}$	entropy production, entropy production rate
θ	entropy potential
T	temperature
$\hat{\beta}$	inverse, halfed temperature
s	"thermodynamic" entropy
ρ	density
$\vec{v}$	vector of velocities
p, p_{mag}, P	hydrodynamic pressure, magnetic pressure, total pressure
E	total energy
$\vec{B}$	vector of magnetic flux densities
Ψ	hyperbolic correction field
$\boldsymbol{\Upsilon}^{\mathrm{Powell}}$, $\boldsymbol{\Phi}^{\mathrm{Powell}}$	Powell source term, Powell vector
$\boldsymbol{\Upsilon}^{\mathrm{GLM}}$, $\boldsymbol{\Phi}^{\mathrm{GLM}}$	GLM source term, GLM vector
σ	species abundance
γ	ratio of heat capacities
c, c_{A}, c_{H}, c_{p}	sonic speed, Alfvén speed, hyperbolic cleaning speed, damping speed
$\mathcal{M}$, $\mathcal{M}_{\mathrm{A}}$, $\mathcal{T}$	sonic Mach number, Alfvén Mach number, turbulent Mach number
β	plasma-beta
$\vec{g}$	vector of gravitational accelerations

$\boldsymbol{\phi}$	test function
ℓ	Lagrange polynomial
ξ	interpolation/quadrature/collocation node
ω	quadrature weight
$\boldsymbol{D}$, $\boldsymbol{M}$, $\boldsymbol{B}^{\pm}$	differentiation matrix, mass matrix, boundary matrices
$\boldsymbol{S}$	skew-symmetric flux differencing matrix
$\boldsymbol{P}$, $\boldsymbol{R}$	projection matrix, reconstruction matrix
α	blending factor

Acronyms

AMR	Adaptive Mesh Refinement
CFD	Computational Fluid Dynamics
CFL	Courant-Friedrichs-Levy
CPU	Central Processing Unit
DG	Discontinuous Galerkin
DGSEM	Discontinuous Galerkin Spectral Element Method
DOF	Degrees of Freedom
EBP	Entropy Boundary Projected
EC	Entropy Conservative
ENO	Essentially Non-Oscillatory
EOC	Experimental Order of Convergence
EOS	Equations-of-State
EPR	Entropy Production Rate
ES	Entropy Stable
FD	Finite Difference
FE	Finite Element
FV	Finite Volume
GLM	Generalized Lagrangian Multiplier
GPU	Graphics Processing Unit
HPC	High Performance Computing
HWENO	Hermite Weighted Essentially Non-Oscillatory
ISM	Interstellar Medium/Matter
IVP	Initial Value Problem
KHI	Kelvin-Helmholtz Instability
LF	Lax-Friedrichs
LS-RK	Low Storage Runge-Kutta
MC	Molecular Cloud
MHD	Magnetohydrodynamic
MPI	Message Passing Interface
ODE	Ordinary Differential Equation
PDE	Partial Differential Equation
PID	Performance Index
PPM	Piecewise Parabolic Method

RHS	Right Hand Side
RH	Rankine-Hugoniot
RK	Runge-Kutta
SBP	Summation by Parts
SN	Supernova
SPH	Smoothed Particle Hydrodynamics
SSP-RK	Strong Stability Preserving Runge-Kutta
TP	Throughput
TVB	Total Variation Bounded
TVD	Total Variation Diminishing
TV	Total Variation
UV	Ultra-violet
WENO	Weighted Essentially Non-Oscillatory
WNM	Warm Neutral Medium

Chapter 1

Introduction

1.1 Background

Computational fluid dynamics (CFD) is defined as a branch of fluid mechanics that uses numerical analysis and computer programs to analyze and solve problems that involve fluid and gas flows. Before the era of computer-assisted research, basic formation mechanisms and processes of cosmic objects, such as galaxies, black holes, stars, and planets were only qualitatively understood at the level of coarse theoretical ideas. In the past, theoretical astronomy was the application of only analytical models based on principles from physics and chemistry to describe and explain astronomical objects and astronomical phenomena. Observations of a phenomenon predicted by a theoretical model then allowed astronomers to select between several alternate or conflicting theories as the one most suitable to describe the process. However, due to the high nonlinearity and huge complexity of physical processes observed in the Universe, pure analytical modeling is quickly stretched to its limits. With the advent of computer systems the complexity and predictive power of models grew along with available automated data processing capacity. Nowadays, CFD is standard in astrophysics and has proven to be a crucial tool for researchers in extending, refining and validating theoretical models for a wide range of cosmic phenomena.

On average, the space between stars is basically empty. The density of the most abundant constituents in the Universe by far, hydrogen and helium, is only a couple of particles per cubic meter. Still, the shear distances between stars add up to vast amounts of interstellar

matter (ISM), where the mean free path between particle collisions is much smaller than the characteristic size of the system on galactic scales. This fact allows us to adequately approximate the dynamics of interstellar gases as fluids. Simulation of astrophysical flows is of particular importance, since many objects and processes of astronomical interest such as stars and molecular clouds (MC) involve gases. In order to obtain a comprehensive picture of the dynamics, it is essential to take all the physics in the system into account. Besides gravity, nuclear physics, radiative transfer and chemistry, particle interactions are also strongly influenced by magnetic fields, which are omnipresent in the Universe. Hence, hydrodynamics alone does not suffice and must be extended to magneto-hydrodynamics (MHD).

The dynamics of fluids are physically described by nonlinear systems of hyperbolic conservation laws. Conservation laws consist of partial differential equations (PDE) tracking the fluid behavior and evolution over time on a given spatial domain. Their solutions are conserved quantities such as density, momentum and energy. Conserved quantities can only change in time depending on the fluid flux in or out of a system. This is referred to as primary conservative. Due to the nonlinearity of most hyperbolic systems, irregular solutions in form of discontinuities (shocks) can develop, even when the flow was initially smooth. Hence, any numerical scheme for solving fluid equations has not only to be correct but must also be robust. Robust schemes do not produce wildly different or even nonsensical results under small changes in the input data - that is to say, their behavior is verifiable and predictable. Furthermore, algorithms that can be proven not to magnify approximation errors are called numerically stable.

With high-speed supercomputers, better solutions can be achieved, and are often required to solve the largest and most complex problems. Ongoing research is directed towards algorithms and software that improves the accuracy and speed of complex simulation scenarios such turbulent flows. Besides astrophysics, CFD is applied to a wide range of research and engineering problems in many fields of study and industries, including aerodynamics and aerospace analysis, weather simulation, natural science and environmental engineering, industrial system design and analysis, biological engineering, fluid flows and heat transfer, and engine and combustion analysis. Since CFD is so universal, advances in one research discipline will directly benefit the progress in other fields.

For approximating the solution of partial differential equations there is a huge variety of numerical methods available and the amount of literature about this broad and actively

researched topic is enormous. Among many, introductory text books into practical numerical analysis and methods from the last thirty years are given by LeVeque (1992); Versteeg and Malalasekera (1995); Toro (1999); LeVeque et al. (2002); Ferziger et al. (2002); Knight (2006); LeVeque (2007); Wesseling (2009); Durran (2010); Dziuk (2010); Lapidus and Pinder (2011); Sastry (2012); Cockburn et al. (2012); Segal (2013); Godlewski and Raviart (2013); Pinder (2018).

Extracting the gist in the literature given above, the most popular grid-based numerical methods for PDEs can be categorized into the following three classes: finite volume (FV), finite difference (FD) and finite element (FE) methods. There are also spectral methods, being a class of their own (Canuto et al. 2007; Shen et al. 2011; Canuto et al. 2012). Classical spectral methods have excellent convergent properties with minimal dissipation and dispersion errors, but only perform well for regularized problems and their computations depend on information from the whole domain. Nevertheless, the dream to leverage the accuracy of spectral methods has led to the construction of high order variants of FV, FD and FE methods with spectral-like properties. High order extensions for FD are straightforward, robust and go along naturally with the ansatz of exact solution points, however, at the price of extensive reconstruction stencils reaching far into the surrounding domain. A very popular and actively researched family of robust, high order FD schemes are the so-called WENO-type (weighted essentially non-oscillatory) FD methods devised by Liu et al. (1994) and Shu and Osher (1988). The adaption of the WENO-mechanism for high order FV schemes, e.g., Zanotti et al. (2014); Núnez-De La Rosa and Munz (2016); Núñez-de la Rosa and Munz (2016), is considered a promising path towards higher accuracy in astrophysics simulations. However, schemes with large stencils are at disadvantage with regards to curved, unstructured meshes and complicate parallel processing due to increased data dependencies. A proper implementation of high order FV schemes can become quite intricate considering the need for appropriate quadrature rules (multiple reconstructed solution points) at volume surfaces. These drawbacks might be the reason why widely used FV implementations for large-scale simulations settle for less accuracy, but in turn are widely appreciated for their simplicity, versatility, and ruggedness. Hence, low order FV schemes, today, are the de facto standard solver class for most CFD frameworks in science and engineering.

A nascent and promising sub-class of FE methods with aspects of FD and FV are so-called discontinuous Galerkin (DG) methods. DG schemes can be interpreted as a mixture of high order FE methods with local polynomial basis functions pinned on high order solution

points and FV methods in the sense that the ansatz space is discontinuous across mesh element interfaces enabling the use of Riemann solvers based numerical surface fluxes. Introductory texts about DG schemes are given, for example, by Karniadakis et al. (2005); Hesthaven and Warburton (2007); Kopriva (2009a). DG offers spectral-like properties (Ainsworth 2004; Gassner and Kopriva 2011; Dobrev et al. 2012; He et al. 2020) while keeping solution data compact with minimal information transfer between neighbors. Their potential for vectorized computing and parallel processing are excellent allowing for very performant CFD codes, e.g., FLEXI (Hindenlang et al. 2012; Krais et al. 2020), FLUXO (Gassner et al. 2016c; Rueda-Ramírez et al. 2021), TRIXI.JL (Schlottke-Lakemper et al. 2021; Ranocha et al. 2022), Horses3D (Manzanero et al. 2019), and ExaHyPe (Reinarz et al. 2020). However, DG methods are not renowned for their robustness (Kirby and Sherwin 2006; Manzanero et al. 2018b). Making DG stable for under-resolved solutions without compromising the spectral-like properties or losing data locality is subject of current research and no "golden" way has been found so far.

It is hoped that the key to universally robust high order methods are so-called entropy stable (ES) schemes (Fjordholm et al. 2012) faithfully obeying the second law of thermodynamics on a discrete level. High order split-form DG schemes based on summation-by-part operators and the simultaneous-approximation-terms technique (Fisher and Carpenter 2013; Chen and Shu 2017; Gassner et al. 2016d, 2018) already showed their stabilizing potential in successfully and reliably carrying out unsteady flow simulations in engineering applications. Stabilized high order DG schemes are capable of directly simulating viscid, weakly compressible turbulence models with considerably elevated accuracy, at lower resolution and with better performance compared to traditional fluid solvers (Gassner and Beck 2013; Beck et al. 2014; Garai et al. 2015; Flad and Gassner 2017). A review on the state of the art of entropy stable DG methods is given, for example, by Gassner and Winters (2021).

To recap, we characterize the three major classes of numerical schemes, namely FV, FD and DG, by the following three broadly defined properties we want from a scheme: accuracy, robustness, and speed. In Figure 1.1 we show a Venn diagram with three filled circles representing the mentioned properties. Their overlapping regions can be associated with our three classes of numerical schemes. We consider high order FD schemes as very accurate and robust, since there are precise shock capturing methods available by selectively switching between different reconstruction polynomials (Engquist et al. 1987; Liu et al. 1994). Their large stencils, however, lessens their potential for highly scalable

codes. Low order FV methods, on the other hand, have moderate data dependency with neighbors and are very robust. Factoring in their comparably relaxed timestep constraints, we deem FV methods as fast albeit not highly accurate. Finally, DG are very accurate and have great performance properties, but, as already discussed, robustness is their weak spot. Of course, this classification is rather crude and can be debated. But at the core, we see it justified to make our point.

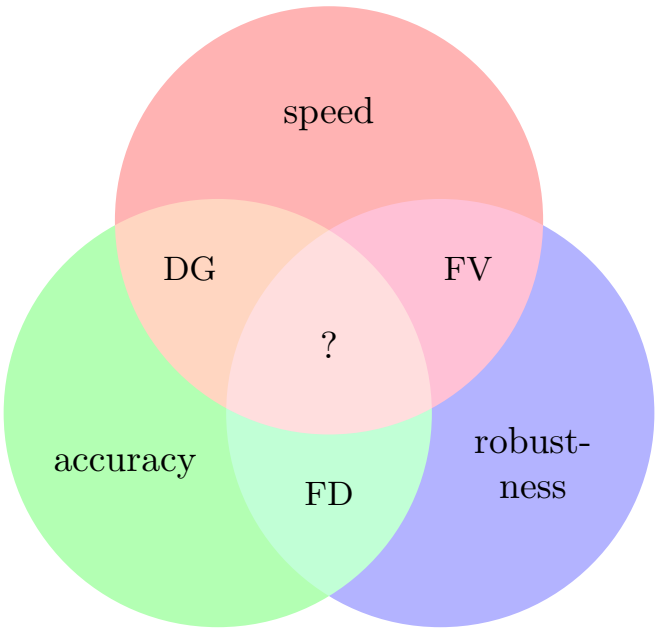

Figure 1.1: Venn diagram with the three properties expected from any numerical scheme. Overlapping regions are associated to the three classes of schemes according to their characteristics in accuracy, robustness and speed.

There is a notable region at the center of Figure 1.1 labeled with a question mark. The naïve reader might be tempted to fantasize about a scheme uniting all three of the fine properties. A scheme that is of high-precision, rock-solid and blazingly fast - the Holy Grail of computational fluid dynamics! Unfortunately, this might be a completely wrong interpretation of the diagram. Instead, it represents a scheme that is neither good at any of the three properties. Mixing all colors never gives you bright shiny gold, but in reality what you get is a faint gray. Consequently, a seasoned designer of novel fluid solvers should be guided by the wisdom from Figure 1.1. There ain't no such thing as a triune scheme, choose two!

Besides purely grid-based techniques for CFD, there are also grid-less approaches available

of which the most widely used method is smoothed-particle hydrodynamics (SPH). An introduction about SPH is given, for example, by Violeau (2012). SPH's inherent adaptive resolution, natural angular momentum conservation, and its ability to simulate phenomena covering a large dynamic range in density makes it very popular for applications in theoretical astrophysics (Springel 2010b). However, comparative studies showed that SPH is not as accurate as grid-based schemes for astrophysical turbulence models since they can become quite diffusive (Agertz et al. 2007). Moreover, the integration of the complex multi-physics aspects in comprehensive astrophysics simulations is still an open research question (Vacondio et al. 2021). The wish to leverage the best of both worlds, namely hybrid methods combining aspects of particles and grids, has led to the construction of moving mesh schemes. For example, Springel (2010a) developed a successful framework for applications in cosmology. However, the implementation of such schemes is generally quite complex and reliable high order accuracy has not yet been achieved (Pakmor et al. 2016).

1.2 State of DG in Astrophysics

Star formation simulations, in particular, are faced with the challenge of accounting for ever expanding models of the physics, while requiring increasingly accurate numerical methods and scalable high-performance implementations. Among a multitude of physical processes, such as highly compressible fluid dynamics with strong shocks, self-gravity of the gas, heating and cooling, multi-component chemistry, and radiation feedback (Walch et al. 2015), magnetic fields are recognized as playing a key role in star formation in the interstellar medium, and possibly impacting the dynamics of the highly turbulent gas in galaxies at larger scales as they get amplified by multiple dynamos.

There are many (open source) simulation frameworks in the astrophysics community that include (a subset of) the aforementioned physical models with different fidelity levels. A few examples are AREPO (Springel 2010a), RAMSES (Fromang et al. 2006), ENZO (Collins et al. 2010), Dedalus (Burns et al. 2020), ATHENA (Stone et al. 2008), ATHENA++ (Stone et al. 2020), ORION2 (Li et al. 2021), and the FLASH code (Fryxell et al. 2000). Here, we list code frameworks that are based on discretizations with meshes, as this strategy is also the focus of this thesis. Most of these codes feature adaptive mesh refinement (AMR), which is crucial to resolve the vastly different spatial scales in complex astrophysical examples. Furthermore, most of these codes are based on FV

type discretization of the fluid/plasma dynamics with numerical fluxes based on Riemann solvers at the interface and second order reconstructions with slope limiting to increase accuracy by decreasing artificial dissipation. FV methods have proven to be versatile, robust and capable to handle strong shocks. Thus, they are currently the state-of-the-art in almost all grid-based multi-physics astrophysical simulation codes. While second order FV methods have proven to be generally very effective, they are hitting a wall in achieving better fidelity and precision while serving the necessity for high efficiency in the upcoming era of exascale computing. The ever-increasing supply of parallel processing power motivates the search for alternative techniques for CFD in astrophysics. A review of the potential for higher order methods in astrophysics is given, for example, by Balsara (2017).

A promising candidate for paving the way towards exascale computing in astrophysical simulations is DG. As already mentioned before, DG methods can be naturally extended to arbitrarily high order of accuracy while keeping the local compute kernels very dense. Furthermore, at least for subsonic turbulence, high order DG offers significant benefits in computational efficiency for reaching a desired target accuracy, due to their spectral-like very low dispersion and dissipation errors. Low numerical dissipation is important to reduce artificial damping and heating. In addition, low dispersion errors are equally important as it guarantees high accuracy for wave propagation and interaction. On top, DG methods intrinsically conserve angular momentum for approximation orders three and higher (Després and Labourasse 2015). These beneficial properties of DG are the reason why applications in engineering are very successful (Bassi et al. 2005; Zhang and Stanescu 2010; Moura et al. 2017; Manzanero et al. 2018a), where mainly weakly compressible turbulent flows in complex geometries are simulated.

Inspired by the success of DG in the engineering disciplines, astrophysicists are eager to apply DG to their problems. DG implementations with focus on astrophysical fluid dynamics and related applications are for instance presented in Mocz et al. (2014); Schaal et al. (2015); Zanotti et al. (2015); Teukolsky (2016); Guillet et al. (2019); Bauer et al. (2016); Kidder et al. (2017); Lombart and Laibe (2020). For example, Bauer et al. (2016) reports that a third order DG method requires significantly less grid resolution than a second order FV scheme to get comparable results in weakly compressible turbulence simulations. This corroborates the superior performance of DG in subsonic turbulence simulations. Moreover, Schaal (2016) showed that higher order DG can evolve ultra cold Keplerian discs on stable orbits over very long time periods, featuring the near zero angular

momentum dissipation. Zanotti et al. (2015) and Guillet et al. (2019) demonstrate that with proper shock capturing DG is capable to carry out a wide range of astrophysical test cases in magneto-hydrodynamical regimes.

However, the successful application of DG for highly compressible, supersonic turbulence is still in an early stage. Supersonic collisions of fluid streams produce shocks observable as sharp solution gradients demanding for very robust numerical schemes, hence the long history of FV schemes in the astrophysics community. While isolated very strong shocks can already be resolved with high acuity for DG, e.g., Zhang and Shu (2010, 2012); Dumbser et al. (2018), finding a high fidelity DG method for turbulent flow regimes teeming with shocks is still an open research question.

To our best knowledge, up to now no DG method has been implemented yet in a full-stack multi-physics framework and applied to full-scale astrophysics simulations.

1.3 Research Mission & Structure of the Thesis

Guided by the current state-of-the-art, we formulate specific research objectives with the principle mission to push forward the utilization of DG in astrophysics.

In Chapter 2 we introduce the basic aspects for hyperbolic systems of nonlinear conservation laws with special focus on the entropy principle. Chapter 3 offers background information about our astrophysical model and justifies our adaption of the ideal magneto-hydrodynamics equations. Included are discussions about the multi-physics aspects we encounter in our envisaged astrophysics simulations. In Chapter 4, we gradually proceed towards our final numerical scheme based on the insights from the first two chapters. First, the basic concepts of FV and DG methods are described while special attention is given to state-of-the-art shock capturing methods and entropy-based stabilization techniques for DG. Additionally, we already interweave numerical results with the aim to answer our first research objective.

1.) Can we successfully run a high order DG method in transonic compressible inviscid flow regimes?

The numerical setup, which we are using to push the several variants of DG methods to their limits, is adjusted such that it starts in a smooth, subsonic state, but can locally develop to challenging transonic flow configurations bringing most DG variants to a crash.

Based on those results, we propose a convex blending scheme and try to answer our second objective.

2.) Can we devise a shock-capturing scheme for high order DG, which is capable of resolving strong shocks in a robust and sharp manner while preserving high order accuracy in smooth, well-resolved flows?

This objective also involves the following questions. How can we ensure positivity of density or pressure and adhere to the divergence constraint in case of the MHD equations?

After we detailed our novel convex blending scheme, we introduce in Chapter 5 our prototype code NEMO and give an overview of technical aspects a typical CFD code entails. Taking a closer look at runtime behavior and scalability helps to assess the overall performance our implementations achieve compared to other codes.

The next step is to expose our DG scheme to multi-physics applications.

3.) Can we fully integrate a "hardened" DG method into a popular, actively maintained multi-physics framework for large-scale astrophysics applications, such as FLASH?

In Chapter 6, we present details of our implementation of the convex blending DG scheme in FLASH. After a solid range of stringent test cases, in which we show that our solver fully connects to all multi-physics aspects of the framework, we are finally ready for the last objective.

4.) Is our new DG method capable of successfully carrying out comprehensive multi-physics simulations?

Chapter 7 presents five exemplary astrophysics applications each with different multi-physics aspects and varying demands on the fluid solver. We also evaluate, which benefits we achieve compared to other long-established fluid solvers readily available in FLASH.

Finally, in Chapter 8 we summarize our findings and contemplate about promising directions follow-up research could aim at in the future.

1.4 Peer-reviewed Publications

Over the course of this research project the following two publications have been accepted in peer-reviewed journals with major recognition in the fields of CFD and astrophysics.

Johannes Markert, Gregor Gassner, and Stefanie Walch. (2020).
A Sub-Element Adaptive Shock Capturing Approach for Discontinuous Galerkin Methods. *Communications on Applied Mathematics and Computation*:

> In this paper, a new strategy for a sub-element based shock capturing for discontinuous Galerkin (DG) approximations is presented. The idea is to interpret a DG element as a collection of data and construct a hierarchy of low to high order discretizations on this set of data, including a first order finite volume scheme up to the full order DG scheme. The different DG discretizations are then blended according to sub-element troubled cell indicators, resulting in a final discretization that adaptively blends from low to high order within a single DG element. The goal is to retain as much high order accuracy as possible, even in simulations with very strong shocks, as e.g. presented in the Sedov test. The framework retains the locality of the standard DG scheme and is hence well suited for a combination with adaptive mesh refinement and parallel computing. The numerical tests demonstrate the sub-element adaptive behavior of the new shock capturing approach and its high accuracy.

Johannes Markert, Stefanie Walch, and Gregor Gassner. (2021).
A Discontinuous Galerkin Solver in the FLASH Multi-Physics Framework. *Monthly Notices of the Royal Astronomical Society, vol. 511, issue 3, pp. 4179-4200*:

> In this paper, we present a discontinuous Galerkin solver based on previous work by the authors for magneto-hydrodynamics in form of a new fluid solver module integrated into the established and well-known multi-physics simulation code FLASH. Our goal is to enable future research on the capabilities and potential advantages of discontinuous Galerkin methods for complex multi-physics simulations in astrophysical settings. We give specific details and adjustments of our implementation within the FLASH framework and present extensive validations and test cases, specifically its interaction with several other physics modules such as (self-)gravity and radiative transfer. We conclude that the new DG solver module in FLASH is ready for use in

astrophysics simulations and thus ready for assessments and investigations.

1.5 Open Source Codes

Over the course of this research project the following two open source codes have been developed. They are publicly accessible and free to use, modify and redistribute.

Johannes Markert.
Nemo - a modular CFD code for rapid prototyping.
github.com/jmark/nemo:

> NEMO is a lightweight, easy to understand (magneto)-hydrodynamics code in 2D/3D leveraging the robustness of 'Finite Volume' (FV) methods and the efficiency of nodal 'Discontinuous Galerkin Spectral Element Methods' (DGSEM). The code is open source, written in modern Fortran and specifically aimed at providing a modular and performant development platform for rapid prototyping new solvers for computational fluid dynamics. NEMO is capable of hybrid parallelization via OpenMP+MPI and supports adaptive mesh refinement on Cartesian grids via the open source library P4EST. NEMO served as the workhorse for all numerical results presented in Markert et al. (2021).

Johannes Markert.
DG for FLASH.
github.com/jmark/DG-for-FLASH:

> Implementation of the Discontinuous-Galerkin-Finite-Volume (DGFV) convex blending scheme for (ideal) magneto-hydrodynamics. The DGFV solver is provided as a FLASH module. In order to use the solver, its source files must be copied to `physics/Hydro/HydroMain/split/DGFV` inside the source tree of your own copy of FLASH. The details about the implementation and numerical validation of this solver module are documented in Markert et al. (2022).

Chapter 2

Hyperbolic Conservation Laws

2.1 Introduction

Continuum mechanics is a branch of physics that deals with the mechanical behavior of materials modeled as a continuous mass rather than as discrete particles. Instead of dealing with the physical properties of solids we focus on the modeling of fluids and gases. As a matter of fact, there are models combining both, solids and fluids, into one system of equations. One example is a first order hyperbolic formulation of continuum mechanics, called HPR model, proposed by Peshkov and Romenski (2016).

Generally, the continuum approach assumes that the investigated medium completely fills the space it occupies and is independent of any particular coordinate system in which it is observed. Modeling fluids in this way ignores the fact that in reality the material is made of atoms or molecules, and so is not continuous on micro-scales. However, on length scales much greater than that of inter-atomic distances, such models are in general sufficiently accurate. The domain - the space filled by the bulk material - is the continuum that can be continually sub-divided into infinitesimal volumes with properties being those of the investigated fluid.

Fundamental physical laws such as the conservation of mass, the conservation of momentum, both linear and angular, and the conservation of energy can be conveniently applied to such models paving the way to derive differential equations, which describe the physical behavior of such objects. The physical properties, e.g., density, pressure or velocity, are represented by continuous fields, which are mathematical objects that have the required

property of being independent of any coordinate system. These fields are transformed to specific coordinates as part of the discretization process.

There are two classical techniques describing the motion of material in the fluid. In the Lagrangian formulation the coordinates are pinned to fluid parcels co-moving through space and time. Therefore, the coordinates of both the fluid parcel and the attached fields do not change along their trajectory; they are time invariant. Movement is expressed by the spatial and temporal continuity between the starting position of the parcel and its position at the considered instant. However, since the reference points move with the fluid flow, it is difficult to know the state of the fluid at a given point in space and time. By contrast, in Eulerian formulation we fixate the coordinate system to a grid (or mesh) through which the material moves as time passes. This temporal change in the field variables can be described by partial derivatives. The velocity of all parcels at each point and instant defines the flow. Both Lagrangian and Eulerian approaches yield mathematically the same result, but the Eulerian formulation is often found more practical.

External forces are forces originating from sources outside of the domain acting on the body of the fluid. These forces arise from the presence of force fields, e.g., gravitational field or electromagnetic field, or from inertial forces when the whole body is in motion. Forces are specified by vector fields, which are assumed to act continuously over the entire domain. Since internal forces from the conservation laws and the external forces from the sources seek to balance each other, the combined model is called a balance law.

Considering the object of study as a closed system, the application of continuum mechanics requires respecting three fundamental physical principles: conservation of total mass, conservation of total momentum and conservation of total energy. This property is called primary conservation. However, most fluid models derived from the real world need additional information about the material under investigation. These material specific properties enter the model as constitutive relations. They connect, for example, the pressure with the density and energy via so-called equations-of-state (EOS). Constitutive relations generally give rise to further conserved quantities; summarized under the term secondary conservation.

Such a quantity from the realm of thermodynamics (the theory of non-equilibrium processes) is the entropy. The concept of entropy first originated in the 1850s in the works of the German physicist and mathematician Rudolf Clausius, where he introduced the

entropy principle characterizing the irreversibility of physical processes, giving the time a steadfast direction, i.e. the arrow of time. This principle is quantitatively described by the second law of thermodynamics. It states that the total entropy in a closed system only stays constant or increases (and never decreases) by any physical process happening within the boundaries of the system. Only exchange of heat and work with the exterior of the system might cause a decrease of entropy within the system, but at the expense of the outside environment.

Clausius (1856a) expanded on his previous work (Clausius 1856b) on the concept of uncompensated transformations (unkompensierte Verwandlungen), which, in our modern nomenclature, would be called the entropy production. He presented an expression for the entropy production $\Delta\mathcal{S}_{\text{phys.}} = 0$ (for a closed system) which reads

$$\Delta\mathcal{S}_{\text{phys.}} = \int \mathrm{d}\mathcal{S} - \int \frac{\mathrm{d}Q}{T} \quad \geq 0. \tag{2.1}$$

If the process is reversible ($\Delta\mathcal{S}_{\text{phys.}} = 0$), the total change in entropy becomes $\int \mathrm{d}\mathcal{S} = \int \mathrm{d}Q/T$, where T is the temperature and Q is the added heat. Hence, $\int \mathrm{d}Q/T$ is called the entropy flux from the outside into the system. If the process is irreversible, $\int \mathrm{d}\mathcal{S} > \int \mathrm{d}Q/T$, the positive difference is then the entropy added to the system. Division by Δt of (2.1) becomes in the infinitesimal limit

$$\frac{\mathrm{d}}{\mathrm{d}t}\Delta\mathcal{S}_{\text{phys.}} = \int \frac{\mathrm{d}}{\mathrm{d}t}\mathrm{d}\mathcal{S} - \int \frac{\mathrm{d}}{\mathrm{d}t}\frac{\mathrm{d}Q}{T} \quad \geq 0 \tag{2.2}$$

the non-negative entropy production rate. Again, the concept of entropy in thermodynamics as a quantity that can either stay constant or increases in isolated systems. However, in the numerics community, as a branch of applied mathematics, it is common practice to define the instant entropy in a system as the upper bound and let the entropy monotonically decrease while evolving in time. The physics does not change; the result is a mere change of signs in the entropy production rate (2.2). The "mathematical" entropy production rate $\frac{\mathrm{d}}{\mathrm{d}t}\Delta\mathcal{S}$ is then always non-positive:

$$\frac{\mathrm{d}}{\mathrm{d}t}\Delta\mathcal{S} = \int \mathrm{d}\mathcal{S} - \int \frac{\mathrm{d}Q}{T} \quad \leq 0. \tag{2.3}$$

From here on, we only refer to the mathematical convention of the entropy.

For the rest of this chapter, we give a brief overview of hyperbolic systems of conservation

laws. We first introduce the basic notations to describe the Cauchy problem for the generic system, define what hyperbolicity means and briefly sketch the connection of the weak formulation of first order hyperbolic partial differential equations (PDE) with the concept of entropy. The last section discusses Riemann problems, which are an indispensable building block for our numerical schemes.

2.2 Systems of Conservation Laws

In this section, we briefly introduce elementary properties of the class of first order hyperbolic PDEs. The selected aspects of these equations are essential for the investigation of the fluid flow and the implementation of numerical methods. The discretization techniques presented are strongly based on the underlying physics and mathematical properties of PDEs. In this thesis, we deal exclusively with hyperbolic PDEs, where hyperbolic conservation laws are a subset of.

In a sense, solutions of hyperbolic equations behave like waves. A disturbance in the initial data is not felt by every point in space at once. Relative to a fixed time coordinate, disturbances have a finite propagation speed. They travel along the characteristics of the equations. This feature qualitatively distinguishes hyperbolic equations from elliptic PDEs and parabolic PDEs. A perturbation in the solution of elliptic or parabolic equations is felt at once by all points in the domain. Heat conduction, for example, is modeled by parabolic PDEs and Newtonian gravity by elliptic PDEs. The equations describing the mechanics of compressible fluids reduce to hyperbolic systems, namely the Euler equations, when any effects of viscosity and heat conduction are neglected. In fact, the Euler equations or rather their extension to the ideal magneto-hydrodynamic equations is, what we are interested in as we discuss in Section 3.4.

Here, we restrict ourselves to some of the basics on hyperbolic PDEs and choose an rather informal way of presentation. The actual derivation of the governing equations is based on integral relations on control volumes (fluid parcels) and their boundaries. Let $\boldsymbol{u} : \Omega \times \mathbb{R}_+ \mapsto \mathbb{R}^m$ be a multi-variate function assigning to each point $\vec{x} = (x, y, z)^T \in \Omega$ in the Cartesian domain $\Omega \subseteq \mathbb{R}^3$ and to each point in time $t \in \mathbb{R}_+$ m conserved quantities. The rate of change of $\boldsymbol{u}$ in any control volume $\Omega_q \subseteq \Omega$ depends on the flux through the

boundary $\partial\Omega_q$ given by the conservation law in integral form

$$\int_{\Omega_q} \partial_t \boldsymbol{u}(\vec{x},t)\, d\vec{x} + \int_{\partial\Omega_q} \sum_{d=1}^{3} \boldsymbol{f}_d(\boldsymbol{u}(\vec{x},t))\, n_d \,\mathrm{d}(\partial\Omega_q) = 0 \tag{2.4}$$

where

$$\boldsymbol{f}_d : \mathbb{R}^m \mapsto \mathbb{R}^m, \quad d = 1,2,3, \tag{2.5}$$

are the smooth multi-variate flux functions for the Cartesian components x, y, z, respectively, and $\vec{n} = (n_1, n_2, n_3)^T$ is the outward normal unit vector at the boundary. If we assume $\boldsymbol{u}$ to be sufficiently smooth, we can apply the divergence theorem (2.4) to obtain the differential formulation

$$\partial_t \boldsymbol{u}(\vec{x},t) + \sum_{d}^{3} \partial_{x_d} \boldsymbol{f}_d(\boldsymbol{u}(\vec{x},t)) = 0, \quad \forall\, \Omega_q. \tag{2.6}$$

The flux functions $\boldsymbol{f}_d$ depend only on the conserved variable $\boldsymbol{u}$ which is the case for most physical fluid phenomena governed by hyperbolic conservation laws. The Cauchy problem for the system (2.6) requires the prescription of initial conditions

$$\boldsymbol{u}(\vec{x}, t=0) = \boldsymbol{u}_0(\vec{x}). \tag{2.7}$$

Next, we define the three flux Jacobians as

$$\boldsymbol{J}_d(\boldsymbol{u}) = \partial_{\boldsymbol{u}} \boldsymbol{f}_d(\boldsymbol{u}) \quad \in \mathbb{R}^{m\times m} \tag{2.8}$$

and say that the system (2.6) is hyperbolic if, for any $\boldsymbol{u} \in \Pi \subset \mathbb{R}^m$ and any $\alpha_d \in \mathbb{R}$, the linear combination $\sum_{d=1}^{3} \alpha_d \boldsymbol{J}_d(\boldsymbol{u})$ has m real eigenvalues $\lambda_m(\boldsymbol{u}) \in \mathbb{R}$ and m linearly independent eigenvectors $\boldsymbol{r}_1(\boldsymbol{u}), \ldots, \boldsymbol{r}_m(\boldsymbol{u})$. Π corresponds to an admissible set dictated by constraints on $\boldsymbol{u}$, for example the positivity of certain quantities like density, pressure or energy. If, in addition, these eigenvalues are distinct, viz. $\lambda_1(\boldsymbol{u}) < \ldots < \lambda_m(\boldsymbol{u})$, then the system is said to be strictly hyperbolic.

2.3 Weak Formulation

The strong nonlinearity of the equations and the lack of regularity of solutions, especially due to the absence of second order (parabolic) terms providing a smoothing effect, account

for most of the difficulties encountered in a rigorous mathematical analysis of such systems. Hyperbolic systems of conservation laws can develop discontinuities in finite time, even when the initial condition of the Cauchy problem is smooth. Thus, we can no longer talk about classical smooth solutions in space and must interpret the solutions for (2.6) in a weak sense. By integrating the inner product of the terms in (2.6) and a smooth test function $\boldsymbol{\phi} \in (C^{\infty}_{\text{compact}}(\mathbb{R}^3))^m$ of compact support in space, we transfer the spatial derivatives to the test function and get a weak form

$$\int_{\mathbb{R}^3} \left(\partial_t \boldsymbol{u}(\vec{x},t)\right) \boldsymbol{\phi}(\vec{x}) \, \mathrm{d}\vec{x} + \int_{\mathbb{R}^3} \sum_d^3 \boldsymbol{f}_d(\boldsymbol{u}(\vec{x},t)) \, \partial_x \boldsymbol{\phi}(\vec{x}) \, \mathrm{d}\vec{x} = 0. \tag{2.9}$$

The function $\boldsymbol{u}$ now only needs to be locally compact and is allowed to have discontinuities. In other words, we relaxed the smoothness conditions on the solution. We call $\boldsymbol{u}$ the weak solution (in space) to (2.6). As a matter of fact, we returned to the more fundamental integral form (2.4) involving integrals over control volumes and their boundaries. From a computational point of view there is another good reason for returning to the integral form. Discretized domains result in finite control volumes or computational cells or elements. Local application of the fundamental equations in these volumes naturally lead to FV schemes and discontinuous Galerkin methods which we introduce in Chapter 4.

Rankine-Hugoniot Jump Condition

By definition, every smooth (or classical) solution is a weak solution. However, not every discontinuity is physically admissible. Consider the surface of a discontinuity Γ moving with speed λ in the $\vec{x} - t$ space, and $\vec{n} \neq \vec{0}$ be its normal vector. When we denote by $\boldsymbol{u}^{\pm}(\boldsymbol{u},t) = \lim_{\epsilon \downarrow 0} \boldsymbol{u}(\vec{x} \pm \epsilon \, \vec{n}, t)$ the limits of $\boldsymbol{u}$ on either side of Γ then the weak solution $\boldsymbol{u}$ must satisfy the following relations

$$\lambda \left(\boldsymbol{u}^+ - \boldsymbol{u}^-\right) = \sum_d^3 \left(\boldsymbol{f}_d(\boldsymbol{u}^+) - \boldsymbol{f}_d(\boldsymbol{u}^-)\right) n_d. \tag{2.10}$$

The above relation, connecting the speed λ of propagation of the discontinuity and the limiting values $\boldsymbol{u}^+$ and $\boldsymbol{u}^-$ on the two sides of the discontinuity, is called Rankine-Hugoniot (RH) jump condition. In general, it is not possible to solve for the propagation speed λ besides smaller linearized systems of equations. Nevertheless, the jump relations are an extremely useful tool in the analysis of shocks which we further investigate in Section 3.5 about shocks in astrophysical settings.

2.4 Entropy Condition

Resorting to the large space of weak solutions, however, comes at the cost of non-uniqueness. In general a weak solution is not unique, and in many cases there is an infinite number of solutions. Hence, additional selection rules are necessary to pick a unique solution subjected to the laws of physics. Two important theorems lay the basis for the proper convergence of a numerical scheme to the correct physical solution of well-posed, linear hyperbolic conservation laws.

The equivalence theorem by Lax and Richtmyer (1956) is a fundamental theorem in the analysis of numerical solution of well-posed, linear initial value problems. A consistent numerical scheme is called convergent, if in the limit of infinitesimal discretization, the bounds on the discretization error is also infinitesimally small. Or in other words, a consistent numerical method converges to the "true solution" if and only if it is Lax-Richtmeyer stable (Richtmyer and Morton 1994). Stability in this context means that the norm of the matrix used in the recurrence relation of the numerical scheme is at most unity guaranteeing a decay of instabilities due to truncation errors of the discretization (LeVeque 2007, Chap. 9).

The second important theorem proven by Lax and Wendroff (1959) states that if the sequence of approximate solutions to a system of hyperbolic conservation laws generated by a conservative and consistent numerical scheme converges as the mesh parameter goes to zero, then the limit is a weak solution of the system. Furthermore, if the scheme satisfies a discrete entropy inequality, the limit is an entropy solution.

From the introduction of this chapter we know, that the concept of entropy was originally derived from the second law of thermodynamics. According to Harten (1983) the entropy principle can be generalized to any arbitrary system of hyperbolic conservation laws and is a selection rule for weak solutions modeling real physical processes.

We assume that the inviscid equations (2.6) are equipped with a strictly convex entropy function

$$\mathcal{S} : \mathbb{R}^m \mapsto \mathbb{R} \tag{2.11}$$

mapping m conservative variables to a real scalar and associated entropy fluxes

$$\mathcal{F}_d : \mathbb{R}^m \mapsto \mathbb{R} \tag{2.12}$$

such that

$$(\partial_{\boldsymbol{u}} \mathcal{F}_d(\boldsymbol{u}))^T = \boldsymbol{w}^T \partial_{\boldsymbol{u}} \boldsymbol{f}_d(\boldsymbol{u}) \tag{2.13}$$

where $\boldsymbol{w} = \partial_{\boldsymbol{u}} \mathcal{S}(\boldsymbol{u})$ is the vector of m entropy variables. Taking the scalar product of (2.6) with $\boldsymbol{w}$ results in the following additional conservation law

$$\partial_t \mathcal{S}(\boldsymbol{u}(\vec{x},t)) + \sum_{d=1}^{3} \partial_{x_d} \mathcal{F}_d(\boldsymbol{u}(\vec{x},t)) = 0 \tag{2.14}$$

which is satisfied for smooth solutions. The entropy condition states that weak solutions should satisfy the entropy inequality

$$\partial_t \mathcal{S}(\boldsymbol{u}(\vec{x},t)) + \sum_{d=1}^{3} \partial_{x_d} \mathcal{F}_d(\boldsymbol{u}(\vec{x},t)) \leq 0 \tag{2.15}$$

which we again understand in a weak sense

$$\int_{\mathbb{R}^3} \Big(\partial_t \mathcal{S}(\boldsymbol{u}(\vec{x},t))\Big)\, \phi(\vec{x}) + \sum_{d=1}^{3} \mathcal{F}_d(\boldsymbol{u}(\vec{x},t))\, \partial_{x_d}\phi(\vec{x}) \;\; \mathrm{d}\vec{x} \;\; \leq \;\; 0 \tag{2.16}$$

for all $\phi \in C^{\infty}_{\text{compact}}(\mathbb{R}^3)$, with $\phi > 0$. The solution $\boldsymbol{u}$ is called an entropy solution if it satisfies (2.16) for every convex entropy $\mathcal{S}$.

Since $\mathcal{S}$ is strictly convex, there exists a one-to-one mapping between $\boldsymbol{u}$ and $\boldsymbol{w}$, which allows the change of variables $\boldsymbol{u} = \boldsymbol{u}(\boldsymbol{w})$. Godunov (1961) and Mock (1980) proved a theorem which links the existence of convex entropy functions for the system (2.6) with the symmetrization of the system under the change of variable. The transformed system

$$(\partial_{\boldsymbol{w}} \boldsymbol{u})\, \partial_t \boldsymbol{w} + \sum_{d=1}^{3} \Big(\partial_{\boldsymbol{w}} \boldsymbol{f}_d(\boldsymbol{w})\Big)\Big(\partial_{x_d} \boldsymbol{w}\Big) = 0 \tag{2.17}$$

is said to be symmetrized by the change of variable $\boldsymbol{u} = \boldsymbol{u}(\boldsymbol{w})$, if the Jacobian $\partial_{\boldsymbol{w}} \boldsymbol{u}$ is symmetric positive definite and $\partial_{\boldsymbol{w}} \boldsymbol{f}_d$ are symmetric.

Furthermore, we define the entropy flux potentials $\theta_d(\boldsymbol{w})$ which are the duals of the entropy fluxes $\mathcal{F}_d(\boldsymbol{u})$ obtained by their Legendre transforms

$$\theta_d(\boldsymbol{w}) = \boldsymbol{w} \cdot \boldsymbol{f}_d(\boldsymbol{u}(\boldsymbol{w})) - \mathcal{F}_d(\boldsymbol{u}(\boldsymbol{w})). \tag{2.18}$$

Rearranging (2.18) gives explicit expressions for the entropy fluxes

$$\mathcal{F}_d : \boldsymbol{w} \mapsto \boldsymbol{w} \cdot \boldsymbol{f}_d(\boldsymbol{u}(\boldsymbol{w})) - \theta_d(\boldsymbol{u}(\boldsymbol{w})) \tag{2.19}$$

where the entropy fluxes $\mathcal{F}_d$ are functions of the entropy variables $\boldsymbol{w}$.

Vanishing Viscosity Solution

The inequality (2.15) with the entropy flux pairs in the definition of entropic solution can be understood as the time-arrow information that should be retained from the microscopic dissipative mechanisms that are neglected on the macroscopic level in our inviscid, hyperbolic system of conservation law (2.6). Since thermodynamics shows that entropy is produced by dissipative processes, we introduce a small dissipative term in the equations (2.6) and analyze the behavior of the limit of a sequence of solutions to these new set equations.

For the case of scalar conservation laws ($m = 1$), every strictly convex function can serve as an entropy function. The idea (Kružkov 1970) is to prove the existence and uniqueness of entropy solutions in the class of functions of bounded variation. The existence is shown by considering the solution u_ϵ of the parabolic problem

$$\partial_t u_\epsilon + \partial_x f(u_\epsilon) = \epsilon \, \partial_x^2 u_\epsilon, \quad \epsilon > 0. \tag{2.20}$$

The limit $\epsilon \to 0$ then converges to an entropy solution u of the original (inviscid) conservation law (Lu 2002). However, the proof relies heavily on local bounds on the total variation of the solution u_ϵ. A condition which is difficult to come by, especially for polynomial approximations of higher order as is done for DG methods. An alternative to the vanishing viscosity approach can be obtained by relaxing the bounded variation conditions on the perturbed solutions u_ϵ and using the method of compensated compactness (Murat 1978; Tartar 1979; Lu 2002), which expands the proof of entropy solutions to specialized, small systems of conservation laws (Lu 2002). Unfortunately, for larger systems of conservation laws, the situation is rather bleak. Apart from some partial results for one-dimensional systems (Bressan et al. 1997; Bianchini and Bressan 2005), no direct path to well-posedness for general systems have been found.

Nevertheless, entropy conditions do play an important role in providing global stability estimates. Formally integrating (2.15) in space and time and assuming suitable decay

conditions at the boundaries (or periodic boundary conditions), we recover an analog to (2.3)

$$\int_\Omega \mathcal{S}(\boldsymbol{u}, t)\,\mathrm{d}\vec{x} \leq \int_\Omega \mathcal{S}_0(\boldsymbol{u})\,\mathrm{d}\vec{x}. \tag{2.21}$$

The above bound on total entropy along with the convexity of $\mathcal{S}$ gives rise to a-priori estimates on the solution of (2.6) in suitable spaces of Lebesque-integrable functions (Dafermos 2005, Chap. 6) which is the only generic nonlinear estimate for systems of conservation laws available at present. There are no well-posedness results available for general multi-dimensional systems of conservation laws. Recently, Chiodaroli et al. (2015) investigated the non-uniqueness of entropy solutions for the isentropic, compressible Euler equations equipped with a very simple pressure law. They showed that in more than one space dimension, the current concepts of an admissible solution fail to yield uniqueness even under very strong assumptions on the initial data. This suggests that the classical notion of entropy solutions may not be the final answer to establish the existence and uniqueness of solutions for general systems.

Entropy Production Rate

In physics, entropy production arises from processes within the system, including chemical reactions, internal matter diffusion, internal heat transfer, and frictional effects such as viscosity occurring within the system from mechanical work transfer to or from the system. But also numerical schemes produce entropy via their inherent numerical diffusion.

Suppose we have a discretization of our conservation law (2.6) denoted by the RHS $\dot{\boldsymbol{u}}$ and by the same discretization mechanism we also get the residual $\dot{\mathcal{S}}$ as the entropy exchange over the boundaries of the sub-domain Ω_q, then we define the local entropy production rate $\dot{\Delta\mathcal{S}}$ of the discretization as

$$\dot{\Delta\mathcal{S}}_q(t) = \boldsymbol{w}(t) \cdot \dot{\boldsymbol{u}}(t)\Big|_{\Omega_q} - \dot{\mathcal{S}}(t)\Big|_{\partial\Omega_q}. \tag{2.22}$$

With (2.22) we measure the entropy production rate caused by the discretization. From this expression we cannot, however, learn if the produced amount of entropy is physically adequate. Only one strict rule applies. The entropy production rate (2.22) shall never become positive. We revisit this issue in Chapter 4.

2.5 Riemann Problem

The Riemann problem (Toro 1999; LeVeque 2002), named after German mathematician Bernhard Riemann, is a specific initial value problem composed of a system of conservation laws together with piecewise constant initial data containing a single discontinuity, or jump, in the domain of interest. The Riemann problem is very useful for the analysis of hyperbolic PDEs and is an import building block in the construction of numerical schemes.

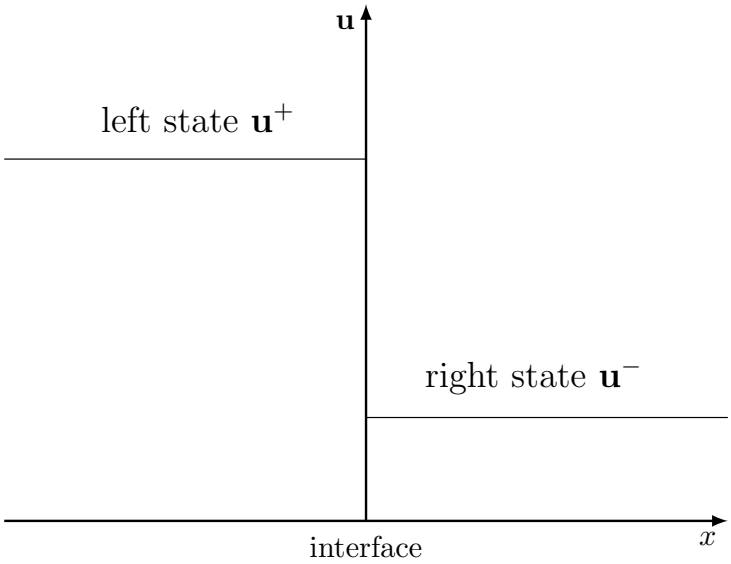

Figure 2.1: Schematic of the Riemann problem (2.23) in 1D with left $\boldsymbol{u}^+$ and right states $\boldsymbol{u}^-$ touching at the interface with the jump $\boldsymbol{u}^- - \boldsymbol{u}^+$.

We consider the following special type of Cauchy problem for a one-dimensional system of hyperbolic conservation laws

$$\partial_t \boldsymbol{u} + \partial_x \boldsymbol{f}(\boldsymbol{u}) = 0 \quad \text{and} \quad \boldsymbol{u}_0(x) = \begin{cases} \boldsymbol{u}^+, & x \leq 0 \\ \boldsymbol{u}^-, & x > 0 \end{cases} \tag{2.23}$$

where $\boldsymbol{u}^+$ and $\boldsymbol{u}^-$ are constant states respectively left and right of the interface. The setup is also depicted in Figure 2.1. As shown in Figure 2.2, the solution to this problem consists of m waves emanating from the origin, corresponding to one of the real eigenvalues (characteristics) λ_l of the flux Jacobian $\partial_{\boldsymbol{u}} \boldsymbol{f}(\boldsymbol{u})$. The solutions to (2.23) are self-similar and are of the form $\boldsymbol{u}(x,t) = \boldsymbol{u}(x/t)$ in the $x-t$ plane.

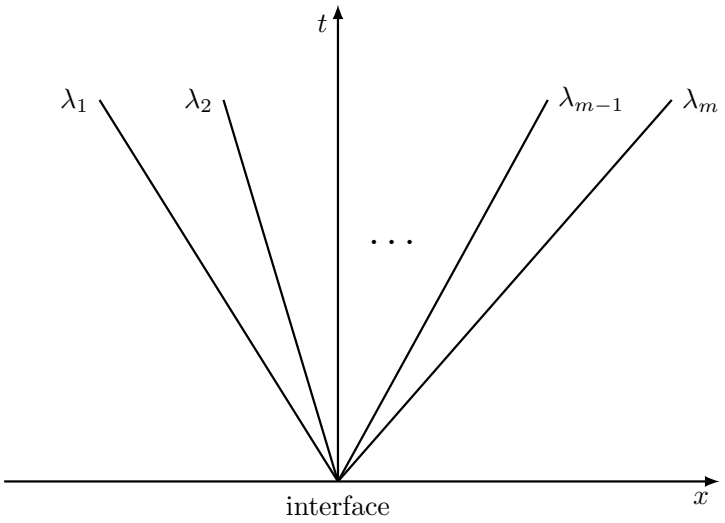

Figure 2.2: Riemann fan of the linearized version of the Cauchy problem (2.23) consisting of m characteristics spreading out into the domain at different speeds λ_i. For the nonlinear equations the characteristics generally show more complex patterns, which are difficult to visualize schematically. The characteristics can be curved, hence they can diverge away from or converge towards each other leading to the versatile flow dynamics discussed in the text.

In the following we consider that $\partial_{\boldsymbol{u}}\lambda_l(\boldsymbol{u})$ is the gradient of the scalar function $\boldsymbol{u} \mapsto \lambda_l(\boldsymbol{u})$ living in the m-dimensional phase space spanned by all possible vectors $\boldsymbol{u} = (u_1, \ldots, u_m)^T$. Thus, $\partial_{\boldsymbol{u}}\lambda_l(\boldsymbol{u}) \cdot \boldsymbol{r}_l(\boldsymbol{u})$ is the directional derivative of eigenvalue λ_l in the direction of the eigenvector $\boldsymbol{r}_l$. Each pair $\Big(\lambda_l(\boldsymbol{u}), \boldsymbol{r}_l(\boldsymbol{u})\Big)$ defines a characteristic field or λ_l-field. Such fields are linearly degenerate if $\partial_{\boldsymbol{u}}\lambda_l(\boldsymbol{u}) \cdot \boldsymbol{r}_l(\boldsymbol{u}) = 0$ for all $\boldsymbol{u} \in \Pi$ which means that the l-th eigenvalue λ_l is constant along each integral curve of the corresponding field of eigenvectors $\boldsymbol{r}_l$. If $\partial_{\boldsymbol{u}}\lambda_l(\boldsymbol{u}) \cdot \boldsymbol{r}_l(\boldsymbol{u}) > 0$ for all $\boldsymbol{u} \in \Pi$ then the system is genuinely nonlinear. Then the eigenvalue λ_l is strictly increasing along each such curve. In general, we will only consider hyperbolic conservation laws which have linearly degenerate or genuinely nonlinear characteristic fields.

With the above assumptions we are ruling out the possibility that along some integral curve of an eigenvector $\boldsymbol{r}_l$, the corresponding eigenvalue λ_l may partly increase and partly decrease, having several local maxima and minima. The solution of the Riemann problem then has a simple structure consisting of superpositions of m elementary waves: shocks, contact discontinuities and rarefactions (Lax 1973). This considerably simplifies the analysis and the construction of approximate Riemann solvers. In the following we briefly describe the three different forms of elementary waves.

The λ_l-wave encodes a shock wave if it corresponds to a genuinely nonlinear field and

connects two states $\boldsymbol{u}^+$ and $\boldsymbol{u}^-$ through a single jump discontinuity. The discontinuity moves with a speed v_l given by the RH condition (2.10). Furthermore, the Lax (1973) entropy condition holds, i.e.,

$$\lambda_l(\boldsymbol{u}^+) > v_l > \lambda_l(\boldsymbol{u}^-), \tag{2.24}$$

which is a result from the entropy condition (2.15). The characteristic lines $dx/dt = \lambda_l$ on both sides of the shock line $dx/dt = v_l$ run into the shock wave by crossing each other and causing a singularity in the solution. Nature resolves this contradiction by dissipation mechanisms and associated entropy production. Any proper numerical solver must mimic this dissipation mechanism in an entropy consistent way.

If λ_l corresponds to a linearly degenerate field, then a contact wave connects two states $\boldsymbol{u}^+$ and $\boldsymbol{u}^-$ through a single jump discontinuity. As in the case of the shock wave, the discontinuity moves with speed v_l given by the RH condition (2.10). It additionally satisfies the parallel characteristic condition

$$\lambda_l(\boldsymbol{u}^+) = v_l = \lambda_l(\boldsymbol{u}^-). \tag{2.25}$$

The characteristic lines on either side of the contact line $dx/dt = v_l$ run parallel to the jump moving unimpeded through phase space. In the case of the Euler equations, contact discontinuities are usually observed by a density jump moving slowly through the domain while the pressure is continuous.

The λ_l-wave corresponds to a rarefaction wave, if it connects the two states $\boldsymbol{u}^+$ and $\boldsymbol{u}^-$ through a smooth transition in a genuinely nonlinear field. The characteristics diverge from each other since

$$\lambda_l(\boldsymbol{u}^+) < \lambda_l(\boldsymbol{u}^-). \tag{2.26}$$

Using the knowledge of breaking down the rather complex dynamics at discontinuous interfaces into a spreading fan of characteristics allows to devise algorithms to numerically solve the Riemann problem (2.23).

Approximate Riemann Solvers

The exact solution of the Riemann problem (2.23) for nonlinear hyperbolic systems of conservation laws is generally very complex and can become computationally expensive. Godunov (1959) laid the foundation for a numerical scheme, called Godunov-type method,

to solve the Riemann problem (2.23) via so-called approximate Riemann solvers. This approach is computationally viable and if combined with proper reconstruction methods can be very robust, yield good numerical results, and has tolerable timestep restrictions. Section 4.5 about FV schemes elaborates on this idea.

If we recall that the entropy solution is the limit of viscous solutions to (2.21) and take a centered flux to which some viscosity (with the right sign) is added, we get a very simple expression of such a flux. It is given by

$$\boldsymbol{f}^* = \{\!\{\boldsymbol{f}\}\!\} - \frac{\lambda^{\max}}{2} [\![\boldsymbol{u}]\!] \tag{2.27}$$

with

$$\lambda^{\max} = \max\left\{ \max_l^m \left|\lambda_l(\boldsymbol{u}^+)\right|, \max_l^m \left|\lambda_l(\boldsymbol{u}^-)\right| \right\}$$

being the maximum characteristic wave speed of the Riemann fan at the interface. Here, we introduce the notation of mean and jump operators at interfaces between the left $(\cdot)^+$ and right $(\cdot)^-$:

$$\{\!\{(\cdot)\}\!\} = \frac{1}{2}\left((\cdot)^+ + (\cdot)^-\right) \quad \text{and} \quad [\![(\cdot)]\!] = (\cdot)^- - (\cdot)^+. \tag{2.28}$$

The so-called Rusanov (1961) flux (2.27), alternatively called Local Lax-Friedrichs flux (Lax 1954), is considered to be computationally cheap and very robust. Taking the viscosity parameter $\lambda^{\max}$ as the largest wave speed at the interface guarantees the stability of the scheme since it covers the whole Riemann fan depicted in Figure 2.2. Of course, this kind of stabilization can be very dissipative, excessively smearing out the numerical solution over time; especially when the jumps tend to be very high. FV schemes strongly depend on the effectiveness and efficiency of Riemann solvers to handle the jumps at cell edges, hence a strong development in this area began as these schemes became popular. More sophisticated approximate Riemann solvers for all kind of equation systems with different special features were developed by taking more information readily available from the Riemann fan into account: Osher (Engquist and Osher 1981), Roe (1981), Harten-Lax-van-Leer (Harten et al. 1983), to name a few.

Affordable Entropy Stable Riemann Solvers

Tadmor (1984) introduced the idea of an entropy conservative numerical flux $\boldsymbol{f}^{\#}$, which

satisfies the following shuffle condition

$$[\![\boldsymbol{w}]\!] \cdot \boldsymbol{f}^{\#} = [\![\theta]\!] \,. \tag{2.29}$$

An algorithm to compute such an entropy conservative flux exactly was given by Tadmor (1987) as

$$\boldsymbol{f}^{\#} = \int_0^1 \boldsymbol{f}\Big(\boldsymbol{w}^{+} + \alpha\, [\![\boldsymbol{w}]\!]\Big)\, \mathrm{d}\alpha \tag{2.30}$$

which is however cumbersome and too expensive for practical applications. Hence, a lot of effort has been put into the construction of affordable entropy conservative Riemann solvers by properly evaluating the jump conditions (2.10) such that they become entropy consistent. Generally, this way of deriving entropy conservative schemes is not unique and it is up to the skills of the inventor to find a good balance between complexity and affordability. In Section 3.4 we give the entropy conservative fluxes $\boldsymbol{f}^{\#}$ for the ideal MHD equations.

The entropy consistent fluxes have a central character and analogously to (2.27) an artificial dissipation term is added to get a guaranteed entropy stable numerical flux

$$\boldsymbol{f}^{*} = \boldsymbol{f}^{\#} - \frac{1}{2}\, \lambda^{\max}\, \boldsymbol{R}\, \boldsymbol{S}\, \boldsymbol{R}^{T}\, [\![\boldsymbol{w}]\!] \tag{2.31}$$

where $\boldsymbol{R}\,\boldsymbol{S}\,\boldsymbol{R}^{T}$ is a positive definite matrix that is guaranteed to cause a negative contribution to the entropy inequality. $\boldsymbol{R}$ is the matrix of characteristic right eigenvectors $\boldsymbol{r}_l$ and $\boldsymbol{S}$ is a diagonal scaling matrix specific to the PDE at hand. Derigs et al. (2017), for example, derives an entropy consistent dissipation operator for the ideal MHD equations. The operator is quite complex and computationally expensive. For our purposes we instead use the following Riemann solver:

$$\boldsymbol{f}^{*} = \boldsymbol{f}^{\#} - \frac{\lambda^{\max}}{2}\, [\![\boldsymbol{u}]\!] \,. \tag{2.32}$$

The numerical flux (2.32) has a very similar structure compared to the Rusanov flux (2.27) and is also robust and reasonably cheap. Even though it is not guaranteed entropy stable, the amount of dissipation produced by the jump operator is usually more than enough to practically provide entropy consistency. The price, of course, is high diffusion at steep gradients in the solution. However, we are interested in high order DG schemes. Such schemes tend to diminish jumps at element interfaces, the higher the polynomial

order is chosen. After all, with the exception of shocks, a fully resolved flow field should be nearly continuous. Consequently, the importance of the Riemann solver decreases with the scheme's increasing order. This was for example investigated by Rider and Lowrie (2002), Qiu et al. (2006) and Wheatley et al. (2010). We are therefore only interested in a simple, robust and computationally cheap Riemann solver that captures the physics reasonably well.

Chapter 3

Astrophysical Model

3.1 Introduction

The interstellar cycle, which takes place within galaxies, is fundamental for our Universe as it controls the formation of stars and therefore the evolution of galaxies. Yet given the broad range of scales in space and time as well as the plethora of physical processes involved, our understanding is still considered to be very incomplete.

Amongst many other physical phenomena, namely gravity, highly compressible turbulence, radiation, cosmic rays, and stellar feedback, magnetic fields are also contributing significantly to the evolution of the interstellar medium (ISM) and more specifically to the formation of stars. In fact, the magnetic energy in the ISM is of comparable order to other energies such as for example the kinetic energy.

Hence, a lot of research in computational astrophysics is dedicated to the role magnetic fields are playing in the formation and evolution of molecular clouds (MC). Within such clouds the formation of molecules (most commonly molecular hydrogen and to a minor extend helium) takes place. MCs are considered to be a major birth place of stars (stellar nursery).

MCs are usually embedded within larger cosmic objects such as galaxies and are heavily affected by their hosts via gravitational pull or cosmic winds. In order to get an idea of the typical size of MCs we take a look at Figure 3.1 which shows Gaia's all-sky view of our Galaxy, the Milky Way. Gaia is a space observatory of the European Space Agency (ESA), launched in 2013. The map shows a false color image of the stars and nebulae

seen by the observatory in each portion of the celestial sphere at around 2015 and then projected onto a rectangle.

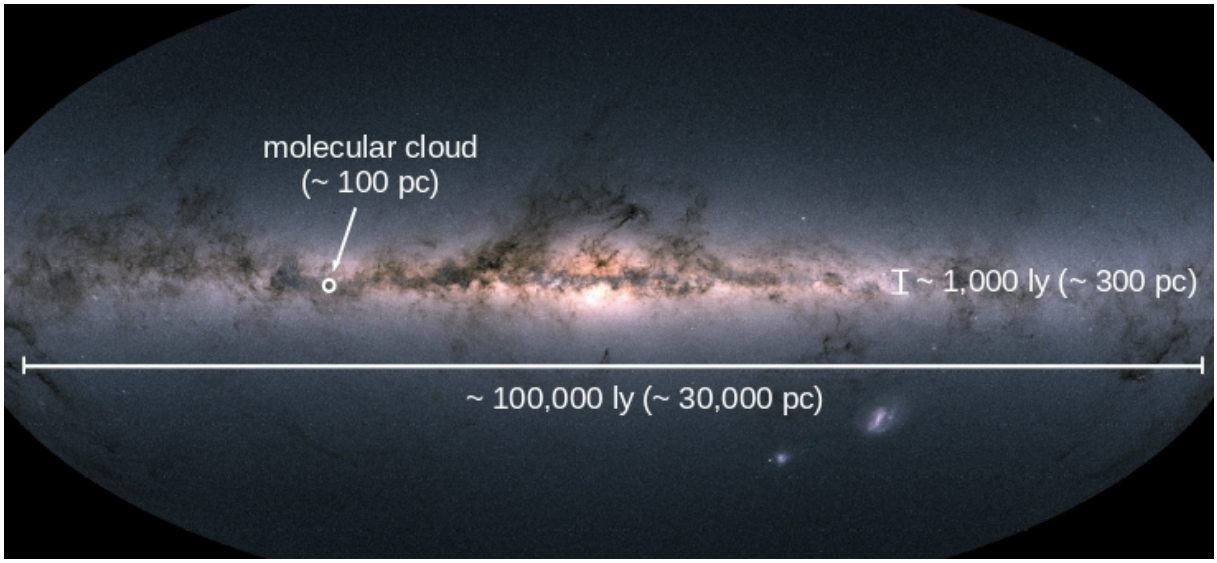

Figure 3.1: Lateral view of our Galaxy, the Milky Way, taken by Gaia, a space observatory of the ESA. The two bright objects in the lower right of the image are the Large and Small Magellanic Clouds, two dwarf galaxies orbiting the Galaxy. Image source: Moitinho et al. (2018).

The bright horizontal structure that dominates the image is called the galactic plane. It is a flattened, rotating disc that hosts most of the stars in our home galaxy. The bright center of the image, the galactic center, is very vivid and teeming with stars.

The darker, frayed regions visible all over the image correspond to foreground clouds of interstellar gas and dust. The light of stars located further away, behind the clouds, gets absorbed casting distinct shadows. These are the objects, which are considered stellar nurseries where new generations of stars are being born. However, the impression of the real size of these clouds is skewed since the observation was done within our Galaxy and not from afar as the image spuriously suggests. The clouds appear larger than the disk, because they are close (on galactic scales) to our solar system.

The diameter of our Galaxy is estimated to be around 100,000 light-years and has a thickness of around 1,000 light-years. MCs range from several hundred to thousands of light-years in diameter and really are objects of galactic proportions. Hence, they fit well within a galaxy, but are large enough to become the sole focus in the study of their inner dynamics. A light-year (ly) is a unit of length and is defined by the distance

light travels in one (Earth) year in empty space. It amounts to $\approx 9.45 \times 10^{12}$ km. For reference, the gravity well (Hill sphere) of our solar system is considered to be between 1 to 3 ly. More usually, the unit commonly used in professional astronomy is the parsec (pc), a portmanteau of "parallax of one second", which is about 3.26 ly. A pc is fixed by the distance at which one astronomical unit (average distance between Sun and Earth) subtends an angle of one second of arc in relation to a distant cosmic object visible in the sky. Or in other words, one pc amounts to 3.086×10^{13} km. That is roughly 206, 286 astronomical units, coincidentally the same size as the aforementioned Hill sphere of our solar system.

Turbulence is an ubiquitous phenomenon in astrophysics and many astronomical observations suggest that MCs are highly turbulent (Elmegreen and Scalo 2004; Scalo and Elmegreen 2004; Hennebelle and Falgarone 2012). Theoretical models suggest that together with gravity, supersonic turbulence is playing a major role in the evolution of MCs for example by creating strong density fluctuations, that in turn may serve as seed for the mass reservoir of future stars. Stellar winds from young clusters of stars and shock waves created by supernovae inject enormous amounts of energy into their surroundings and are considered to be the driving energy source for the observed turbulent motion in MCs.

A galactic year (also called cosmic year) is the duration of time required for our solar system to orbit once around the center of our Galaxy. That is about 230 million years. Turbulent dynamics in MCs, such as the turnover time, are clocked in tenths of millions of years, hence happen on similar time scale as the dynamics of their host galaxy.

Unthreading the different roles magnetic fields are playing is however quite difficult, since directly measuring them in astronomical observations remains a challenge. Magnetic fields behave not like a mere pressure and are highly non-isotropic in nature, and because observations do not allow us to easily vary the parameters as it is possible to do so in experiments on Earth. This however can be done in numerical simulations, where the influence of a specific parameter, like the magnetic intensity, can be modified and studied. The exact understanding on how magnetic fields affect turbulence in MCs is still an open research question.

The long, stringy structures in the aforementioned darker, frayed regions in Figure 3.1 indicate patches of large negative velocity divergence, marking zones of strong compression. Such shockwaves, or sometimes called shocklets within the context of supersonic

turbulence, are discontinuities in the flow moving at supersonic speed and rapidly compressing and heating up the colder ambient gas. The presence of shocklets is extremely challenging for computational models of compressible turbulent flows.

Since the majority of astrophysical models are set in empty space there is no demand for unstructured meshes describing complex geometries as it is common for example in industrial applications. Simple Cartesian boxes, or cubes, with outflow boundaries suffice in almost all cases. There is, however, a huge demand for resolution. Hence, adaptive mesh refinement strategies are absolute essential in order to cover the vast range of scales from whole galaxies sometimes even down to individual star systems.

In this chapter we introduce the interstellar medium, the substance our fluid model (continuum) is made of, discuss the gigantic scales in space and time we encounter, investigate the governing equations we are solving, analyze the role of shocks and give a brief overview on the role of chemistry, gravity and radiation physics in our simulations.

3.2 Interstellar Medium

In astrophysics, the ISM is the material and radiation that exist in the space between the star systems in a galaxy. This matter includes gas in ionic, atomic, and molecular form, as well as dust and cosmic rays. It permeates the entire interstellar space and its energy that occupies the same volume, in the form of electromagnetic radiation, is the interstellar radiation field.

The ISM is composed of multiple phases distinguished by whether components are ionic, atomic, or molecular as well as the temperature and density of the material. Specifically it consists, primarily, of hydrogen ($\approx 75\%$), followed by helium ($\approx 25\%$) with trace amounts of carbon, oxygen, and nitrogen. The hydrogen and helium are primarily a result of primordial nucleosynthesis, while the heavier elements in the ISM are mostly a result of enrichment in the process of stellar life cycles. The thermal pressures of the gas components are in rough equilibrium with one another. It is believed that most of the volume in the thin ($\sim 10^2\,\mathrm{pc}$) disk of our own Galaxy is filled by warm neutral medium and warm or hot ionized medium (Ferriere 2001) with typical temperatures up to $10^4\,\mathrm{K}$.

The interstellar gas is extremely dilute, with an average density of about 1 atom per cubic centimeter. For comparison, the air in Earth's atmosphere has a density of approximately 3×10^{19} molecules per cubic centimeter. Nonetheless, the mean free paths of the particles

are still short compared to the huge sizes of the regions occupied. The particles undergo many collisions before traversing a significant fraction of the region. The particle velocity distributions can be therefore considered Maxwellian and we can describe them by a gas kinetic temperature, which is usually assumed to be the same for all species of particles present. Even though the interstellar gas is very dilute, the amount of matter adds up over the vast distances on galactic scales.

Magnetic fields and turbulent motions also induce pressure into the ISM, and are typically more important, dynamically, than the thermal pressure. The quantity which characterizes this relationship is the so-called plasma-beta

$$\beta = \frac{p}{p_{\text{mag}}} \tag{3.1}$$

which is the ratio of thermal (resp. hydrodynamical) pressure p to magnetic pressure p_{mag}. Hence, a low β-flow is said to be dominated by magnetic forces. Since electrical conductivity in the ISM is extremely high, any resistive effects can be neglected.

A good understanding of ISM is of crucial importance in astrophysics precisely because of its intermediate role between stellar and galactic scales. Stars form within the densest regions of the ISM, which ultimately contribute to MCs and in turn replenish the ISM with matter and energy through proto-planetary nebulae, stellar winds, and supernovae. This interplay between stars and the ISM helps to determine the rate at which a galaxy depletes its gaseous content, and therefore its burning rate by forming stars.

Detailed descriptions of the ISM's properties can be found for example in Boyd et al. (2003) or Dyson and Williams (2020).

3.3 Compressible Euler Equations

We begin by considering the compressible Euler equations, which describe the fluid mechanics in the absence of any viscous forces. The equations are formulated based on fundamental principles of conservation of mass, momentum and energy, and can be shown to be hyperbolic in nature. We also discuss the entropy framework for this system, which will play a crucial role in constructing suitable entropy consistent numerical schemes. As a matter of fact, the compressible Euler equations are the hydrodynamical limit of the ideal MHD equations for zero magnetic fields $|\vec{B}| \to 0$. Hence, a proper understanding of the

Euler equations and their fluid dynamics forms a good basis for investigations of the complex dynamics the complete magneto-hydrodynamical regime reveals. The compressible Euler equations in 3D are defined as

$$\partial_t \begin{pmatrix} \rho \\ \rho\vec{v} \\ E \end{pmatrix} + \nabla \cdot \begin{pmatrix} \rho\vec{v} \\ \rho\,\vec{v} \otimes \vec{v} + p\,\mathbb{1} \\ (E+p)\,\vec{v} \end{pmatrix} = \mathbf{0}, \tag{3.2}$$

with the vector of conserved quantities $\boldsymbol{u} = (\rho, \rho\,\vec{v}, E)^T$, where ρ denotes the density, $\vec{v} = (v_1, v_2, v_3)^T$ the velocity, and quantity E is the total energy per unit volume

$$E = \rho\left(e + \frac{1}{2}\vec{v}^2\right). \tag{3.3}$$

The specific internal energy e is given by a caloric equation-of-state, $e = e(\rho, p)$. We choose the equation-of-state to be that of the ideal gas. Pressure, density and internal energy are related by

$$p = (\gamma - 1)\,\rho\,e \tag{3.4}$$

where $\gamma = c_p/c_v$ is the ratio of specific heats of the gas model for constant pressure c_p and constant volume c_v. Usually, we choose $\gamma = 5/3$ accounting for the mostly monoatomic components (protons) of the ISM. The gas temperature T is related to density and pressure by

$$p = R_{\text{specific}}\,\rho\,T \tag{3.5}$$

with $R_{\text{specific}} = c_p - c_v$ being the specific gas constant. The set of physically permissible states is given by

$$\Pi = \Big\{\text{permissible states}\Big\} = \Big\{\forall\,\boldsymbol{u} \;\Big|\; \rho > 0 \wedge p(\boldsymbol{u}) > 0\Big\}. \tag{3.6}$$

The first equation in (3.2) describes the conservation of mass, followed by three equations ensuring the conservation of the spatial components of momentum, while the final equation describes the conservation of total energy. If we bring the (3.2) into the form (2.6) we

succinctly write

$$\boldsymbol{u} = \begin{pmatrix} \rho \\ \rho\, v_1 \\ \rho\, v_2 \\ \rho\, v_3 \\ E \end{pmatrix} \quad \text{and} \quad \boldsymbol{f}_d(\boldsymbol{u}) \begin{pmatrix} \rho\, v_d \\ \rho\, v_d\, v_1 + p\, \delta_{d,1} \\ \rho\, v_d\, v_2 + p\, \delta_{d,2} \\ \rho\, v_d\, v_3 + p\, \delta_{d,3} \\ (E + p)\, v_d \end{pmatrix}, \tag{3.7}$$

where we introduce the Kronecker delta

$$\delta_{i,j} = \begin{cases} 1, & i = j \\ 0, & \text{otherwise.} \end{cases} \tag{3.8}$$

Eigenvalues

The five eigenvalues of the flux Jacobian (2.8) for the compressible Euler equations (3.2) in direction d are given by

$$\lambda_1^{(d)} = v_d - c, \quad \lambda_2^{(d)} = \lambda_3^{(d)} = \lambda_4^{(d)} = v_d, \quad \lambda_5^{(d)} = v_d + c \tag{3.9}$$

with sonic signal speed c of the medium given by

$$c = \sqrt{\gamma\, T}. \tag{3.10}$$

For water or air one usually calls it the speed of sound. Assuming the positivity of density and pressure, the eigenvalues are real and the corresponding eigenvectors are linearly-independent, thus making the system hyperbolic.

Sonic & Turbulent Mach number

We define the unitless sonic Mach number $\mathcal{M}$ of the flow as

$$\mathcal{M} = \frac{|\vec{v}|}{c} \tag{3.11}$$

named after the Austro-Czech physicists Ernst Mach. The sonic Mach number is used to distinguish various flow regimes: the flow is subsonic for $\mathcal{M} < 1$, supersonic for $\mathcal{M} > 1$ and transonic if the flow has both supersonic and subsonic regions. For $\mathcal{M} > 5$ the flow is under hypersonic regime. At very slow flow speeds the speed of sound is so much faster

that it plays a negligible role in the flow dynamics. Once the speed of the flow approaches the speed of sound, however, the sonic Mach number becomes all-important, and shock waves begin to emerge.

If we would characterize the local structure of the flow simply by computing a mean sonic Mach number $\mathcal{M}$, one might merely measure the bulk velocity of a fluid parcel moving unimpeded through the domain. In contrast to engineering applications, there are no obstacles the fluid can interact with since our astrophysics models are set in open space. Consequently, structures in the fluid can only emerge via self-interaction, which we identify as turbulence. In order to quantify the local "strength" of the turbulence, we compute the turbulent Mach number $\mathcal{T}_q$ within fluid parcel Ω_q as

$$\mathcal{T}_q = \frac{\sqrt{|\Omega_q|^{-1} \int_{\Omega_q} (\vec{v}(\vec{x}) - \langle \vec{v} \rangle_q)^2 \, \mathrm{d}\vec{x}}}{\langle c \rangle_q}, \tag{3.12}$$

where $\langle u \rangle_q$ is the average value of quantity u in the sub-domain Ω_q, i.e.

$$\langle u \rangle_q = |\Omega_q|^{-1} \int_{\Omega_q} u(\vec{x}) \, \mathrm{d}\vec{x}.$$

The turbulent Mach number $\mathcal{T}$ is a measure of the compressibility of the turbulent flow and in the CFD community it is common to characterize subsonic flows of $\mathcal{T} < 0.3$ as weakly compressible (or even incompressible). For example in Anderson Jr (2010), a relation for a calorically perfect gas between the Mach number $\mathcal{M}$ and the ratio of stagnant [1] density $\rho_{\text{stag.}}$ to static density $\rho_{\text{stat.}}$ is derived. We adapt the relation to the notion of turbulent Mach numbers and define

$$\left(\frac{\rho_{\text{stag.}}}{\rho_{\text{stat.}}} \right)_q = \left(1 + \frac{\gamma - 1}{2} \mathcal{T}_q^2 \right)^{1/(\gamma - 1)}. \tag{3.13}$$

It basically computes the change in density if a turbulent fluid parcel Ω_q would have been adiabatically put to rest. The kinetic energy would have been completely transformed into thermal energy entailing a change in temperature, pressure and density of the parcel. In Figure 3.2 equation (3.13) is plotted in relation to the turbulent Mach number $\mathcal{T}$ and various heat capacity ratios γ. As a rule of thumb, a fluid is deemed compressible for

[1] In aerodynamics, the terms "total pressure" and "total density" of a gas are commonly used when the flow is brought to rest isentropically. Since "total density" is already occupied in the context of multi-component fluids, we use the term "stagnant density" instead.

a density variation of more than 5 % demarked by the dashed gray line. As seen in Figure 3.2 all plotted curves cross this threshold at $\mathcal{T} \approx 0.3$.

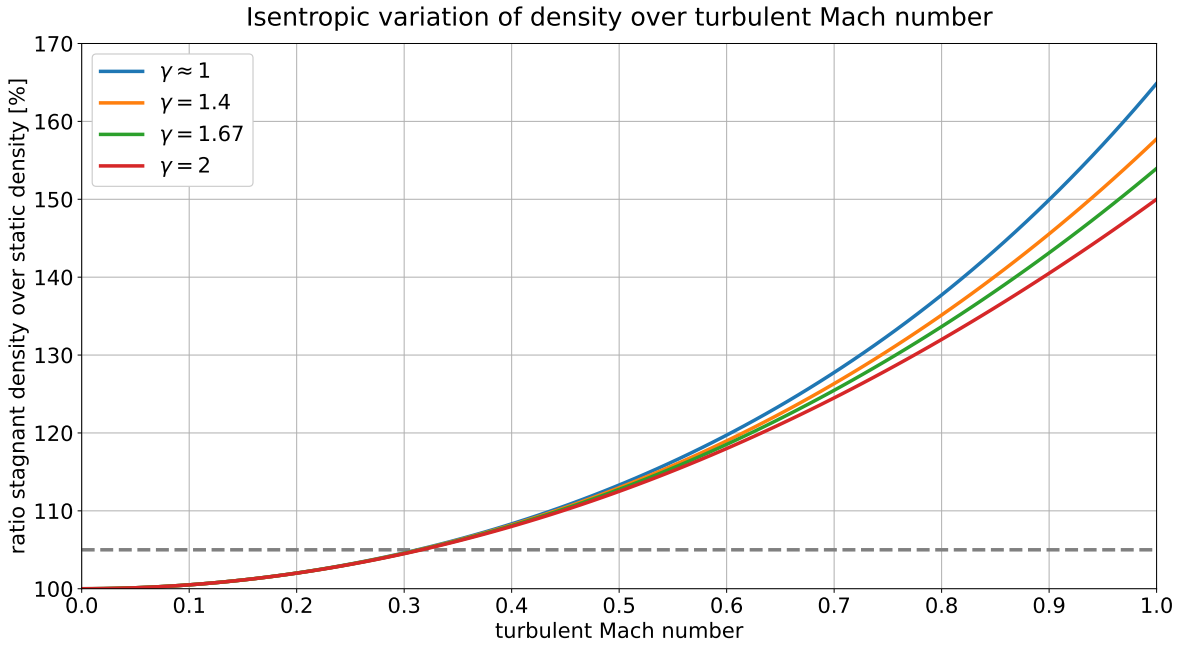

Figure 3.2: Ratio of stagnant density vs static density (3.13) plotted over turbulent Mach number $\mathcal{T}$ for different heat capacity ratios γ. A flow is considered compressible when the variation in density is above 5 % (gray dashed line).

Due to the nonlinearity of relation (3.13), the turbulent Mach numbers grows very fast beyond the threshold and quickly enters the regime of highly compressible turbulence. This regime already puts a lot of strain on numerical schemes even if the flow is still globally subsonic. Clearly, compressibility in the gas allows phenomena not observed in strictly incompressible turbulence, namely the production of random shocklets. They can form when turbulent fluctuations of large magnitude reinforce local conditions such that compression waves perpetually steepen the flow (Samtaney et al. 2001).

Entropy Variables

Harten et al. (1998) has shown that the compressible Euler equations (3.2) are equipped with a family of entropy-entropy flux pairs (2.11,2.12) of the form

$$\mathcal{S}(\boldsymbol{u}) = -\frac{\rho\, s}{\gamma - 1} \quad \text{and} \quad \mathcal{F}_d(\boldsymbol{u}) = \mathcal{S}(\boldsymbol{u})\, v_d \tag{3.14}$$

with physical entropy

$$s = \log(p) - \gamma \log(\rho). \tag{3.15}$$

The corresponding entropy variables $\boldsymbol{w}$ are given by

$$\boldsymbol{w}(\boldsymbol{u}) = \begin{pmatrix} \frac{\gamma - s}{\gamma - 1} - \hat{\beta}\, \vec{v}^2 \\ 2\,\hat{\beta}\, v_1 \\ 2\,\hat{\beta}\, v_2 \\ 2\,\hat{\beta}\, v_3 \\ -2\,\hat{\beta} \end{pmatrix} \quad \text{with} \quad \hat{\beta} = \frac{\rho}{2\,p}. \tag{3.16}$$

The entropy potential is given by

$$\theta_d = \rho\, v_d. \tag{3.17}$$

For reference, we also list the inverse of (3.16) mapping the entropy variables back to conservative variables.

$$\boldsymbol{u}(\boldsymbol{w}) = \begin{pmatrix} \hat{\rho} \\ -\hat{\rho}\, w_2/w_5 \\ -\hat{\rho}\, w_3/w_5 \\ -\hat{\rho}\, w_4/w_5 \\ \hat{\rho}\, \hat{e} \end{pmatrix} \quad \text{with} \quad \begin{gathered} \hat{s} = \gamma - (\gamma - 1)\left(w_1 - \frac{w_2^2 + w_3^2 + w_4^2}{w_5}\right), \\ \hat{\rho} = \left(-\frac{\exp(-\hat{s})}{w_5}\right)^{1/(\gamma-1)} \\ \text{and} \\ \hat{e} = -\frac{1}{(\gamma-1)\, w_5} + \frac{1}{2}\,\frac{w_2^2 + w_3^2 + w_4^2}{w_5^2}. \end{gathered} \tag{3.18}$$

3.4 Ideal Magneto-hydrodynamic Equations

Magneto-hydrodynamics (MHD) is the study of the magnetic properties and behavior of electrically conducting fluids and was initiated by Swedish physicist Hannes Alfvén. Examples of such magnetized fluids include plasmas, liquid metals, salt water, and electrolytes. The word MHD is derived from magneto - meaning magnetic field, hydro - meaning water, and dynamics - meaning movement. Ideal MHD is the most basic single-fluid model for determining the macroscopic equilibrium and stability properties of an inviscid and perfectly conducting fluid. In this thesis, we interchangeably use the terms plasma, fluid, gas, and medium for any kind of such a perfectly conducting matter.

The fundamental concept behind MHD is that magnetic fields can induce currents in moving plasmas, which in turn polarize the plasma and reciprocally change the magnetic field

topology themselves. The set of equations that describe ideal MHD are a combination of the compressible Euler equations (3.2) and Maxwell's equations of electromagnetism. These differential equations must be solved simultaneously, either analytically or numerically.

The basic requirement for the validity of ideal MHD is that plasma particles are collision dominated. This is the general principle for any fluid model (continuum assumption). If there are sufficient collisions, a given particle remains reasonably close to its neighboring particles during the time scales of interest. In this case the division of the plasma into small identifiable fluid parcels provides a good description of the physics. The particle distribution is therefore close to Maxwellian, that is an ideal gas (particle interaction only via elastic collisions) in thermodynamic equilibrium. The resistivity due to these collisions is also considered small. In particular, the typical magnetic diffusion times over any length and time scale is much larger than at microscopic levels. Clearly, ISM satisfies these requirements nicely, being an extremely thin, (nearly) perfectly conducting gas moving in cosmic length and time scales.

Considering the MHD of a compressible, non-viscous and perfectly conducting fluid it is our goal to numerically solve the ideal MHD equations in their conserved formulation:

$$\partial_t \begin{pmatrix} \rho \\ \rho\,\vec{v} \\ E \\ \vec{B} \end{pmatrix} + \nabla \cdot \begin{pmatrix} \rho\,\vec{v} \\ \rho\,\vec{v} \otimes \vec{v} + P\,\mathbb{1} - \vec{B} \otimes \vec{B} \\ (E+P)\vec{v} - (\vec{v}\cdot\vec{B})\vec{B} \\ \vec{B} \otimes \vec{v} - \vec{v} \otimes \vec{B} \end{pmatrix} = 0, \tag{3.19}$$

where ρ is the density, $\vec{v} = (v_1, v_2, v_3)^T$ is the velocity, E is the total energy and $\vec{B} = (B_1, B_2, B_3)^T$ is the magnetic field vector. For the sake of brevity, we assumed the magnetic permeability to be $\mu_0 := 1$ and dropped any factors from the equations. $\mathbb{1}$ represents the 3×3 identity matrix and the total pressure P is the sum of thermal and magnetic pressure:

$$P = p + \frac{1}{2}\vec{B}^{\,2}. \tag{3.20}$$

The total energy E is related to the thermal pressure p via the equation-of-state

$$p = (\gamma - 1)\Big(E - \frac{\rho}{2}\vec{v}^{\,2} - \frac{1}{2}\vec{B}^{\,2}\Big), \tag{3.21}$$

where γ as the ratio of heat capacities for constant volume and pressure. For ISM one

usually sets $\gamma = 5/3$ modeling a mono-atomic, ideal gas.

Additionally, Maxwell's equations impose $\nabla \cdot \vec{B} = 0$ on all physical realizations of the magnetic field, throughout the physical domain Ω and at all times t. While the induction equation guarantees that an initially divergence-free magnetic field will remain so in time, truncation errors in numerical schemes can lead to generation of nonzero numerical divergence and accumulation over time. These errors can trigger a nonlinear instability in the MHD equations and cause blowup of the numerical solution (Brackbill and Barnes 1980; Tóth 2000; Kemm 2013). In addition, even if the numerical divergence stays bounded, divergence errors can still result in spurious perturbations to the flow, such as acceleration of the fluid along the magnetic field lines.

Divergence Control

The issue of divergence control has received a lot of attention in the context of FD and FV MHD codes, resulting in the development of multiple techniques. Projection methods (Brackbill and Barnes 1980) project the magnetic field onto a globally divergence-free representation after each evolution step. The main drawback of this technique is that it requires solving a global elliptic Poisson problem at each projection operation, which is expensive and less scalable for distributed computing. Constrained transport (CT), another family of methods, keeps the magnetic field divergence-free up to machine precision via a careful numerical discretization and update scheme for the induction equation (Evans and Hawley 1988; Ryu et al. 1998; Balsara and Spicer 1999; Gardiner and Stone 2005). The method has been very popular with FD and FV grid codes in astrophysics such as Zeus-2D (Stone and Norman 1992), RAMSES (Fromang et al. 2006), ENZO (Collins et al. 2010) and FLASH (Lee 2013). The advantage is its suitability for second order mesh methods, exact divergence control, and lack of any tunable parameter in the scheme. CT has also been extended to higher order methods such as WENO (Balsara et al. 2009) and also DG, involving either dual discretizations (Li et al. 2011; Balsara and Käppeli 2017; Zhao and Tang 2017) or updating a vector potential with its own higher order DG discretization (Rossmanith 2013). The main drawback of CT for DG is its implementation complexity and cost, requiring significantly more operations and storage to update the magnetic field in a divergence-free way.

For this work, we adopt two established divergence control techniques, which allow working with cell-centered discretizations while preserving the hyperbolic character of the

equations: the Powell scheme, based on the addition of a non-conservative source term to the ideal MHD equations, and hyperbolic divergence cleaning, which dynamically disperses the numerical divergence at speed c_H using an additional scalar field Ψ. The extended ideal MHD equations (Derigs et al. 2018) then read

$$\partial_t \begin{pmatrix} \rho \\ \rho\,\vec{v} \\ E \\ \vec{B} \\ \Psi \end{pmatrix} + \nabla \cdot \begin{pmatrix} \rho\,\vec{v} \\ \rho\,\vec{v}\otimes\vec{v} + P\,\mathbb{1} - \vec{B}\otimes\vec{B} \\ (E+P)\vec{v} - (\vec{v}\cdot\vec{B} - c_\mathrm{H}\,\Psi)\vec{B} \\ \vec{B}\otimes\vec{v} - \vec{v}\otimes\vec{B} + c_\mathrm{H}\,\psi\,\mathbb{1} \\ c_\mathrm{H}\,\vec{B} \end{pmatrix} = \underbrace{-\,(\nabla\cdot\vec{B}) \begin{pmatrix} 0 \\ \vec{B} \\ \vec{v}\cdot\vec{B} \\ \vec{v} \\ 0 \end{pmatrix}}_{\boldsymbol{\Upsilon}^\mathrm{Powell}} \underbrace{-\,(\nabla\Psi)\cdot \begin{pmatrix} \vec{0} \\ 0 \\ \vec{v}\,\Psi \\ 0 \\ \vec{v} \end{pmatrix}}_{\boldsymbol{\Upsilon}^\mathrm{GLM}} \tag{3.22}$$

Powell Source Term

Following the theoretical groundwork by Godunov (1972), Powell et al. (1999) pointed out that the system (3.19) is not Galilean invariant and does not formally conserve entropy. He proposed to add a specific source term proportional to $\nabla \cdot B$ in order to symmetrize the hyperbolic system. The so-called Powell term can be obtained from deriving the local form of the system (3.19) based on integral conservation laws (Powell et al. 1999) or from requiring entropy stability (Godunov 1972; Chandrashekar and Klingenberg 2016; Winters and Gassner 2016; Derigs et al. 2018).

We write the Powell source terms in (3.22) as

$$\boldsymbol{\Upsilon}^\mathrm{Powell} = \left(\partial_x B_1 + \partial_y B_2 + \partial_z B_3\right) \boldsymbol{\Phi}^\mathrm{Powell} \quad \text{with} \tag{3.23}$$

$$\boldsymbol{\Phi}^\mathrm{Powell} = \left(0,\ B_1, B_2, B_3,\ \vec{v}\cdot\vec{B},\ v_1, v_2, v_3,\ 0\right)^T. \tag{3.24}$$

The Powell method can be easily adapted to Eulerian grid codes without setting or tuning any free parameters. It has been successfully implemented and tested in astrophysical MHD codes equipped with a FV scheme and adaptive mesh refinement (AMR) (Derigs et al. 2016). For DG it was adopted by Warburton and Karniadakis (1999); Bohm et al. (2018) for viscous and resistive MHD flows.

However, the Powell method has two limitations. Firstly, it does not fully eliminate

the divergence error, as it advects it away with the flow. Consequently, it can result in local accumulation of numerical divergence in the case of standing shocks (Balsara and Spicer 1999; Tóth 2000). Secondly, the source term is not strictly conservative anymore since it will locally inject conserved quantities in the presence of numerical divergence errors causing deviations in the jump conditions across shock fronts. We investigate these issues in the numerical results sections and argue that the combination with hyperbolic divergence cleaning, discussed in the next paragraph, remedies these issues and leads to acceptable results.

Hyperbolic Divergence Cleaning

Munz et al. (2007) coupled the divergence constraint for the electric field with the induction equation by introducing a generalized Lagrangian multiplier (GLM) Ψ as an additional field. As in Dedner et al. (2002) we apply this technique to ideal MHD in order to account for the divergence-free condition $\nabla \cdot \vec{B} = 0$ by adding the GLM Ψ as another conservative state variable, which we call hyperbolic divergence correction field. This new field couples to the divergence of the magnetic field through a modified induction equation (Derigs et al. 2018):

$$\partial_t \vec{B} + \nabla \cdot (\vec{B} \otimes \vec{v} - \vec{v} \otimes \vec{B}) + \nabla \Psi = 0.$$

The field Ψ evolves via the dynamical equation

$$\partial_t \Psi + c_{\mathrm{H}} \nabla \cdot \vec{B} + \vec{v} \nabla \Psi + \Psi = 0$$

resulting into a coupled GLM-MHD system which makes the fluctuations of Ψ propagate away from their sources at speed $c_{\mathrm{H}} > 0$ while damping them with the damping speed $c_{\mathrm{p}} > 0$ at time-scales $\propto c_{\mathrm{H}}^{-1}$.

We write the GLM source terms in (3.22) as

$$\boldsymbol{\Upsilon}^{\mathrm{GLM}} = \left(v_1\, \partial_x \Psi + v_2\, \partial_y \Psi + v_3\, \partial_z \Psi\right) \cdot \boldsymbol{\Phi}^{\mathrm{GLM}} + c_{\mathrm{p}}\, \boldsymbol{\Phi}^{\mathrm{damp}} \tag{3.25}$$

with GLM vectors

$$\boldsymbol{\Phi}^{\mathrm{GLM}} = \left(0,\ \vec{0},\ \Psi,\ \vec{0},\ 1\right)^T \quad \text{and} \tag{3.26}$$

$$\boldsymbol{\Phi}^{\mathrm{damp}} = \left(0,\ \vec{0},\ 0,\ \vec{0},\ \Psi\right)^T. \tag{3.27}$$

For disappearing divergence errors the correction field $\Psi \to 0$ and the GLM contributions vanish. Thus, the GLM modifications to the ideal MHD model are consistent and restore the continuous limit.

For efficient divergence propagation without compromising stability we set c_H to be the maximum magneto-sonic wave speed (3.35) (defined further down below) which is present in the entire physical domain Ω:

$$c_\mathrm{H} = \max_{\Omega} \lambda^{\max}. \tag{3.28}$$

According to Dedner et al. (2002) a good choice for the damping speed, c_p, is given by

$$c_\mathrm{p} = \frac{c_\mathrm{H}}{0.18}.$$

The GLM method is straightforward to implement in existing schemes and it has also been adopted by a number of MHD codes, e.g., Gaburov and Nitadori (2011); Mignone et al. (2012); Dumbser and Loubère (2016); Bohm et al. (2018); Rueda-Ramírez et al. (2021).

Ideal GLM-MHD Equations

To summarize, we want to numerically solve a three-dimensional, hyperbolic balance law describing the ideal generalized Lagrange multiplier magneto-hydrodynamics (iGLM-MHD), i.e.

$$\partial_t \boldsymbol{u} + \sum_d^3 \partial_{x_d} \boldsymbol{f}_d(\boldsymbol{u}) = -\left(\boldsymbol{\Upsilon}^{\mathrm{Powell}} + \boldsymbol{\Upsilon}^{\mathrm{GLM}}\right). \tag{3.29}$$

The vector of conservative state variables and the flux in direction d read as

$$\boldsymbol{u} = \begin{pmatrix} \rho \\ \rho\, v_1 \\ \rho\, v_2 \\ \rho\, v_3 \\ E \\ B_1 \\ B_2 \\ B_3 \\ \Psi \end{pmatrix} \quad \text{and} \quad \boldsymbol{f}_d(\boldsymbol{u}) = \begin{pmatrix} \rho\, v_d \\ \rho\, v_d\, v_1 - B_d\, B_1 + P\, \delta_{d,1} \\ \rho\, v_d\, v_2 - B_d\, B_2 + P\, \delta_{d,2} \\ \rho\, v_d\, v_3 - B_d\, B_3 + P\, \delta_{d,3} \\ \left(\frac{\rho}{2}\,\vec{v}^{\,2} + \frac{\gamma p}{\gamma - 1} + \vec{B}^{\,2}\right) v_d - \left(\vec{v}\cdot\vec{B} - c_{\mathrm{H}}\Psi\right) B_d \\ v_d\, B_1 - v_1\, B_d + c_{\mathrm{H}}\, \Psi\, \delta_{d,1} \\ v_d\, B_2 - v_2\, B_d + c_{\mathrm{H}}\, \Psi\, \delta_{d,2} \\ v_d\, B_3 - v_3\, B_d + c_{\mathrm{H}}\, \Psi\, \delta_{d,3} \\ c_{\mathrm{H}}\, B_d \end{pmatrix} \tag{3.30}$$

with state variables density ρ, velocity $\vec{v} = (v_1, v_2, v_3)^T$, magnetic field vector $\vec{B} = (B_1, B_2, B_3)^T$, and hyperbolic divergence correction field Ψ. The total energy E is related to the thermal pressure p via the equation of state

$$p = (\gamma - 1)\left(E - \frac{\rho}{2}\vec{v}^{\,2} - \frac{1}{2}\vec{B}^{\,2} - \frac{1}{2}\Psi^2\right), \tag{3.31}$$

where γ is the ratio of heat capacities for constant volume and pressure. For ISM one usually sets $\gamma = 5/3$ modeling a mono-atomic, ideal gas.

The set of permissible states is defined by

$$\Pi = \left\{\text{permissible states}\right\} = \left\{\forall\, \boldsymbol{u} \;\middle|\; \rho > 0 \wedge p(\boldsymbol{u}) > 0\right\}. \tag{3.32}$$

In simulations, we strictly enforce positive densities and positive thermal pressures throughout the whole domain and at all times.

Eigenvalues

The nine eigenvalues of the system in direction d are

$$\lambda^{(d)}_{\pm\mathrm{fast}} = v_d \pm c^{(d)}_{\mathrm{fast}},\ \lambda^{(d)}_{\pm\mathrm{slow}} = v_d \pm c^{(d)}_{\mathrm{slow}},\ \lambda^{(d)}_{\pm\mathrm{A}} = v_d \pm c^{(d)}_{\mathrm{A}},\ \lambda^{(d)}_{\pm\mathrm{H}} = v_d \pm c_{\mathrm{H}},\ \lambda^{(d)}_{\mathrm{E}} = v_d \tag{3.33}$$

with

$$c_{\mathrm{A}}^{(d)} = |b_d|, \quad c_{\mathrm{fast,slow}}^{(d)} = \sqrt{\frac{1}{2}\left(c^2 + \vec{b}^2 \pm \sqrt{(c^2+\vec{b}^2)^2 - (2\,c\,b_d)^2}\right)}, \quad \vec{b} = \frac{\vec{B}}{\sqrt{\rho}} \tag{3.34}$$

and c being the sonic signal speed (3.10). c_{fast} and c_{slow} are called fast and slow magneto-sonic wave speeds, respectively, and c_{A} is the signal speed of the three Alfvén waves $\vec{b} = (b_1, b_2, b_3)^T$. The full eigensystem of the system (3.29) as well as its detailed derivation is documented in Derigs et al. (2018).

A crucial step part of our numerical treatment of iGLM-MHD is to compute the maximum eigenvalue of the system. It encodes the maximum wave speed involved in the solution and helps to find a good estimation for an acceptable timestep in case of explicit time integration. It is calculated by the upper bound of all involved fast magneto-sonic wave speeds (3.34) in all directions d:

$$\lambda^{\max} = \max_{d=1}^{3} |v_d| + c_{\mathrm{fast}}^{(d)}. \tag{3.35}$$

Note, in the hydrodynamic limit, $|\vec{B}| \to 0$, equation (3.35) reduces to

$$\lambda^{\max} = \max_{d=1}^{3} |v_d| + c.$$

which is the maximum wave speed estimate for the compressible Euler equations (3.2).

Alfvén Mach Number

The unitless Alfvén Mach number $\mathcal{A}$ is analogously to the sonic Mach number $\mathcal{M}$ defined as the ratio of the plasma speed to the speed of Alfvén waves:

$$\mathcal{A} = \frac{|\vec{v}|}{|\vec{b}|}. \tag{3.36}$$

The Alfvén Mach number is an important quantity in characterizing shock waves under the MHD flow regime, which we investigate further in Section 3.5.

Entropy Variables

According to Derigs et al. (2018), the ideal GLM-MHD equations are equipped with a family of entropy-entropy flux pairs (2.11,2.12) of the form

$$\mathcal{S}(\boldsymbol{u}) = -\frac{\rho\, s}{\gamma - 1} \quad \text{and} \quad \mathcal{F}_d(\boldsymbol{u}) = \mathcal{S}(\boldsymbol{u})\, v_d \quad \text{with} \quad s = \log(p) - \gamma\, \log(\rho). \tag{3.37}$$

The corresponding entropy variables $\boldsymbol{v}$ are given by

$$\boldsymbol{w}(\boldsymbol{u}) = \begin{pmatrix} \frac{\gamma - s}{\gamma - 1} - \hat{\beta}\, \vec{v}^{\,2} \\ 2\,\hat{\beta}\, v_1 \\ 2\,\hat{\beta}\, v_2 \\ 2\,\hat{\beta}\, v_3 \\ -2\,\hat{\beta} \\ 2\,\hat{\beta}\, B_1 \\ 2\,\hat{\beta}\, B_2 \\ 2\,\hat{\beta}\, B_3 \\ 2\,\hat{\beta}\, \Psi \end{pmatrix} \quad \text{with} \quad \hat{\beta} = \frac{\rho}{2\,p}. \tag{3.38}$$

Note, that $\hat{\beta}$ is not be confused with the plasma-beta β in (3.1). The entropy potential is given by

$$\theta_d = \rho\, v_d + \hat{\beta}\, v_d\, \vec{B}^{\,2} + 2\,\hat{\beta}\, c_{\mathrm{H}}\, \Psi\, B_d. \tag{3.39}$$

For completeness, we list the inverse of (3.16) mapping the entropy variables back to conservative variables.

$$\boldsymbol{u}(\boldsymbol{w}) = \begin{pmatrix} \hat{\rho} \\ -\hat{\rho}\, w_2/w_5 \\ -\hat{\rho}\, w_3/w_5 \\ -\hat{\rho}\, w_4/w_5 \\ \hat{\rho}\, \hat{e} \\ -\hat{\rho}\, w_6/w_5 \\ -\hat{\rho}\, w_7/w_5 \\ -\hat{\rho}\, w_8/w_5 \\ -\hat{\rho}\, w_9/w_5 \end{pmatrix} \quad \text{with} \quad \begin{gathered} \hat{s} = \gamma - (\gamma - 1)\, \left(w_1 - \tfrac{w_2^2 + w_3^2 + w_4^2}{w_5} \right), \\ \hat{\rho} = \left(-\tfrac{\exp(-\hat{s})}{w_5} \right)^{1/(\gamma - 1)} \\ \text{and} \\ \hat{e} = -\tfrac{1}{(\gamma - 1)\, w_5} + \tfrac{1}{2}\, \tfrac{w_2^2 + w_3^2 + w_4^2}{w_5^2} + \tfrac{1}{2}\, \tfrac{w_6^2 + w_7^2 + w_8^2 + w_9^2}{\hat{\rho}\, w_5^2}. \end{gathered} \tag{3.40}$$

Affordable Entropy Conservative Riemann Solver

In Section 2.5 we introduce the concept of affordable Riemann solvers. Derigs et al. (2018) devised an kinetic energy preserving and entropy conservative (KEPEC) Riemann solver for the ideal GLM-MHD equations. It reads

$$\boldsymbol{f}_d^{\#}(\boldsymbol{u}^+, \boldsymbol{u}^-) = \begin{pmatrix} \{\!\{\rho\}\!\}_{\ln} \, \{\!\{v_d\}\!\} \\ \{\!\{\rho\}\!\}_{\ln} \, \{\!\{v_d\}\!\} \, \{\!\{v_1\}\!\} - \{\!\{B_d\}\!\} \, \{\!\{B_1\}\!\} + P^{\#} \, \delta_{d,1} \\ \{\!\{\rho\}\!\}_{\ln} \, \{\!\{v_d\}\!\} \, \{\!\{v_2\}\!\} - \{\!\{B_d\}\!\} \, \{\!\{B_2\}\!\} + P^{\#} \, \delta_{d,2} \\ \{\!\{\rho\}\!\}_{\ln} \, \{\!\{v_d\}\!\} \, \{\!\{v_3\}\!\} - \{\!\{B_d\}\!\} \, \{\!\{B_3\}\!\} + P^{\#} \, \delta_{d,3} \\ (f_5^{\#})_d \\ \{\!\{v_d\}\!\} \, \{\!\{B_1\}\!\} - \{\!\{v_1\}\!\} \, \{\!\{B_d\}\!\} + c_{\mathrm{H}} \, \{\!\{\Psi\}\!\} \, \delta_{d,1} \\ \{\!\{v_d\}\!\} \, \{\!\{B_2\}\!\} - \{\!\{v_2\}\!\} \, \{\!\{B_d\}\!\} + c_{\mathrm{H}} \, \{\!\{\Psi\}\!\} \, \delta_{d,2} \\ \{\!\{v_d\}\!\} \, \{\!\{B_3\}\!\} - \{\!\{v_3\}\!\} \, \{\!\{B_d\}\!\} + c_{\mathrm{H}} \, \{\!\{\Psi\}\!\} \, \delta_{d,3} \\ c_{\mathrm{H}} \, \{\!\{B_d\}\!\} \end{pmatrix} \tag{3.41}$$

with

$$P^{\#} = \frac{\{\!\{\rho\}\!\}}{2 \, \{\!\{\hat{\beta}\}\!\}} + \frac{1}{2} \sum_{d=1}^{3} \{\!\{B_d^2\}\!\}$$

and

$$(f_5^{\#})_d \;=\; (f_1^{\#})_d \left(\left(2\,(\gamma-1) \, \{\!\{\hat{\beta}\}\!\}_{\ln} \right)^{-1} - \frac{1}{2} \sum_{e=1}^{3} \{\!\{v_e^2\}\!\} \right) + \sum_{e=1}^{3} (f_{e+1}^{\#})_d \, \{\!\{v_d\}\!\} +$$
$$\sum_{e=1}^{3} (f_{e+5}^{\#})_d \, \{\!\{B_e\}\!\} + (f_9^{\#})_d \, \{\!\{\Psi\}\!\} + \frac{1}{2} \sum_{e=1}^{3} \{\!\{v_d \, B_e^2\}\!\} + \{\!\{B_d\}\!\} \left(\sum_{e=1}^{3} \{\!\{v_e \, B_e\}\!\} \right) - c_{\mathrm{H}} \, \{\!\{B_d \, \Psi\}\!\} \, .$$

$\{\!\{(\cdot)\}\!\}_{\ln} = \frac{[\![(\cdot)]\!]}{[\![\ln(\cdot)]\!]}$ is the logarithmic mean. A numerically stable procedure to compute the logarithmic mean is for example described by Ismail and Roe (2009). The numerical, central-like flux $f^{\#}$ is consistent with the physical flux, that is $\boldsymbol{f}_d^*(\boldsymbol{u}, \boldsymbol{u}) = \boldsymbol{f}_d$, and, together with the discretization of the non-conservative terms in (3.29), conserves the discrete entropy by construction. The discretization of (3.23) and (3.25) are given by (4.20) and (4.21) in Section 4.5. Note, that the non-conservative terms vanish when the left/right states are identical, reflecting convergence to the continuous case, where the divergence of the magnetic field vanishes. In the hydrodynamical limit, $|\vec{B}| \to 0$, the

numerical flux (3.41) reduces to the KEPEC flux for the compressible Euler equations first described by Chandrashekar (2013).

3.5 Shocks

From a theoretical point of view, (magneto-)hydrodynamic shocks are a manifestation of the nonlinearity of the plasma evolution equations (3.19), hence giving rise to nonlinear waves and discontinuities. Most importantly, supersonic waves can steepen into shock waves. That is, sound waves are unable to propagate ahead of the disturbance rapidly accumulating to a highly compressed, hot, and very narrow transition layer separating the heated post-shock region in fluid state $\boldsymbol{u}^+$ from the cooler pre-shock region in fluid state $\boldsymbol{u}^-$. From the perspective of the co-moving rest frame, colder fluid particles stream in at supersonic speed v^- from the right and get abruptly decelerated by colliding with the slower gas particles within the shock. During the collision the incoming gas gets compressed and heated, thereby maintaining a discontinuous shock surface. This surface, or shock front, separates two distinct hydrodynamic states. Figure 3.3 shows a sketch of such a shockwave in the co-moving rest frame moving alongside the shock front.

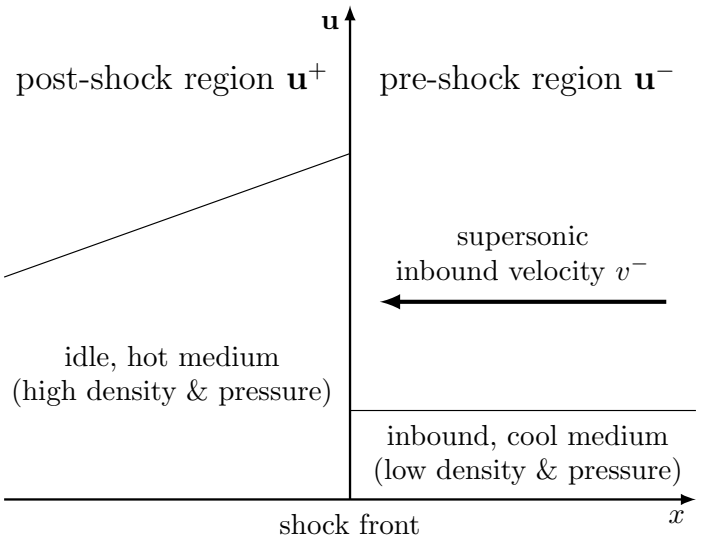

Figure 3.3: Sketch of a shockwave in the co-moving reference frame moving at same speed as the shock front. On the right side is the untouched cool medium in pre-shock state $\boldsymbol{u}^-$ and on the left side is the heated medium in post-shock state $\boldsymbol{u}^+$. Due to cooling effects, the heated fluid quickly cools shortly after the shock has swept through as is indicated by the tilted state profile in the post-shock region.

We indicate the pre-shock fluid properties with the minus sign (−), and post-shock prop-

erties with the plus sign (+). Behind the shock, the gas has raised density ($\rho^+ > \rho^-$) and pressure ($p^+ > p^-$) and is slowed down to subsonic speeds ($v^+ < v^-$). Moreover, the shock irreversibly converts kinetic energy of the pre-shock fluid into thermal energy, and massively increases the entropy of the gas. In other words, energy gets dissipated at high rates throughout the process. Together with the pre-shock sound speed, we know the Mach number of the shock $\mathcal{M} = |v^-|/c^-$. Since the inflowing fluid stream moves supersonically and information propagates in the gas at the speed of sound, it implies that, until the moment of collision, the pre-shock gas is not affected by the shock.

So far, we ignored any implications on the shock in the presence of magnetic fields. There is a class of shocks in the MHD regime, which differs significantly from the "classical" hydrodynamical shocks, where magnetic fields are weak or even absent. In many astrophysical environments the mean free path of the particles is larger than the transition layer of the shock. Consequently, any abrupt deceleration of particles do not originate from particle collisions. Instead, the presence of magnetic fields causes the rapid plasma deceleration by means of electromagnetic resistivity. This type of shock is called collisionless shock and is solely governed by magnetic forces. To complicate matters even further, we recall that information in plasmas is carried via three different waves, namely, fast (or compressional) Alfvén waves, intermediate (or shear) Alfvén waves, and slow (or magneto-sonic) waves. Hence, plasmas governed by MHD equations support a multitude of different types of shocks, corresponding to the huge configuration space of disturbances each traveling faster than one of the aforementioned waves. We have already mentioned the two extreme cases.

For our following investigations, we simplify our perspective on MHD shocks and state that a shock propagating through a MHD fluid produces a significant alteration in plasma properties after going through the shock. In nature, shocks are not perfectly narrow discontinuities and their thickness of the front is determined by the balance between convective and dissipative effects. Hence, they rather consist of broadened transition layers with a scale of the order of the mean free path of the gas particles. Which is in most cases still very narrow compared to the feasible resolution capacities of the simulation code. In such layers, the energy dissipation occurs by means of viscous effects and heat conduction, which are, however, absent in our chosen fluid model of ideal MHD equations (3.19). Basically, the actual physics inside the shock transition layer is not covered in the model. Nonetheless, the effect on the gas by the shock and any associated energy dissipation can be uniquely characterized by the Rankine-Hugoniot jump conditions (2.10)

and by the magneto-hydrodynamic quantities of the pre- and post-shock states. Since the shock is assumed to be sufficiently narrow, these relations become independent of any detailed structure in the transition layer. In this sense, a quantitatively correct treatment of shocks can be achieved even with our idealized fluid model.

Our primary goal is to devise proper shock capturing methods for high order DG schemes. In the following, we derive the jump relations for a strong ($\mathcal{M} \gg 1$), narrow, planar MHD shock in co-moving rest frame (steady-state) and, by that, get insights into which flow parameters are good candidates for indicating a shock. With these assumptions, the Rankine-Hugoniot jump relations (2.10) for the ideal MHD equations (3.19) are

$$\begin{aligned}
\llbracket \rho\, \vec{v} \cdot \vec{n} \rrbracket &= 0 \\
\Big\llbracket \rho\, \vec{v} (\vec{v} \cdot \vec{n}) + (p + \vec{B}^2/2)\vec{n} - (\vec{B} \cdot \vec{n})\vec{B} \Big\rrbracket &= 0 \\
\Big\llbracket \Big(\rho\, e + \rho\, \vec{v}^2/2 + \vec{B}^2/2 + p + \vec{B}^2/2\Big)(\vec{v} \cdot \vec{n}) - (\vec{B} \cdot \vec{n})(\vec{B} \cdot \vec{v}) \Big\rrbracket &= 0 \\
\Big\llbracket \vec{n} \times (\vec{v} \times \vec{B}) \Big\rrbracket &= 0 \\
\Big\llbracket \vec{B} \cdot \vec{n} \Big\rrbracket &= 0
\end{aligned} \tag{3.42}$$

using the succinct bracket notation (2.28) for the fluid state jumps at discontinuities.

Our very fast moving shock, moving faster than the speed of sound c and any of the Alfvén waves c_A, is characterized by the following flow parameters

$$\vec{v}^{\,+} = (v^+, 0, 0)^T,\ \vec{B}^{\,+} = (0, B^+, 0) \quad \text{and} \quad \vec{v}^{\,-} = (v^-, 0, 0)^T,\ \vec{B}^{\,-} = (0, B^-, 0). \tag{3.43}$$

Inserting into the jump conditions (3.42) gives the following relations (Draine and McKee 1993)

$$\frac{\rho^+}{\rho^-} = \frac{2\,(\gamma+1)}{D + (D^2 + 4(\gamma-1)(2-\gamma)\mathcal{A}^{-2})^{1/2}}, \tag{3.44}$$

$$\frac{B^+}{B^-} = \frac{\rho^+}{\rho^-}, \quad \frac{v^+}{v^-} = \frac{\rho^-}{\rho^+} \tag{3.45}$$

$$\frac{p^+}{p^-} = 1 + \gamma\,\mathcal{M}^2 \left(1 - \frac{\rho^-}{\rho^+}\right) + \beta^{-1} \left(1 - \left(\frac{\rho^+}{\rho^-}\right)^2\right) \tag{3.46}$$

with $D = (\gamma - 1) + 2\,\mathcal{M}^{-2} + \gamma\,\mathcal{A}^{-2}$. Above relations depend on three parameters: sonic

Mach number $\mathcal{M}$, Alfvén Mach number $\mathcal{A}$ and plasma-beta β. We can eliminate one parameter by expressing the Alfvén Mach number with the other two, i.e.

$$\mathcal{A} = \mathcal{M} \left(\frac{2}{\gamma \beta} \right)^{-1/2}. \tag{3.47}$$

When we eliminate the magnetic field by letting $|\vec{B}| \to 0$ we get the relations for a purely hydrodynamic shock, since $\beta^{-1} \to 0$. The jump ratio for density then reads

$$\lim_{|\vec{B}| \to 0} \frac{\rho^-}{\rho^+} = \frac{\gamma + 1}{\gamma - 1 + 2\,\mathcal{M}^{-2}}. \tag{3.48}$$

leading to the well known insight that the maximum density jump ratio for very strong shocks ($\mathcal{M} \gg 1$) is bounded and only depends on γ:

$$\lim_{\mathcal{M} \to \infty} \frac{\rho^-}{\rho^+} = \frac{\gamma + 1}{\gamma - 1}. \tag{3.49}$$

For $\gamma = 5/3$ the maximum density jump ratio is 4 and for $\gamma = 1.4$ we get a ratio of 6. In the isothermal limit $\gamma \to 1$ the ratio goes to infinity, which implies that in isothermal regimes the proper handling of shocks by a numerical scheme can become extremely tricky.

In Figure 3.4 we plot the jump ratios for $\gamma = 5/3$ over the sonic Mach number $\mathcal{M}$ for the physical quantities density, pressure, temperature and entropy. We compare the behavior in the hydrodynamic limit $\beta \to \infty$ with an exemplary magnetic pressure dominated regime of $\beta = 2/5$. The jumps for temperature and entropy are calculated from (3.44), (3.5) and (3.15). They read

$$\frac{T^+}{T^-} = \gamma \frac{p^+}{p^-} \frac{\rho^-}{\rho^+} \quad \text{and} \quad \frac{s^+}{s^-} = \frac{p^+}{p^-} \left(\frac{\rho^-}{\rho^+} \right)^{\gamma}.$$

The second law of thermodynamics requires shocks to be compressive: $\frac{\rho^+}{\rho^-} > 1$. As a matter of fact, this is a general rule, which applies to all types of MHD shocks (Boyd et al. 2003). This constraint is observable in Figure 3.4 for the density jump in the magnetic regime ($\beta = 2/5$, blue dashed line). The physically permissible shock regime starts not until sonic Mach number $\mathcal{M} > 2$ when the density jump ratio gets larger than one. Apparently, for weaker shocks, we get less compression for strong magnetic fields, which in turn is a consequence of the additional resistance to compression provided by

the magnetic pressure. In other words, magnetically dominated shocks are somewhat mitigated. Hence, shock capturing schemes that work well for hydrodynamical regimes can be expected to also work well for MHD flows.

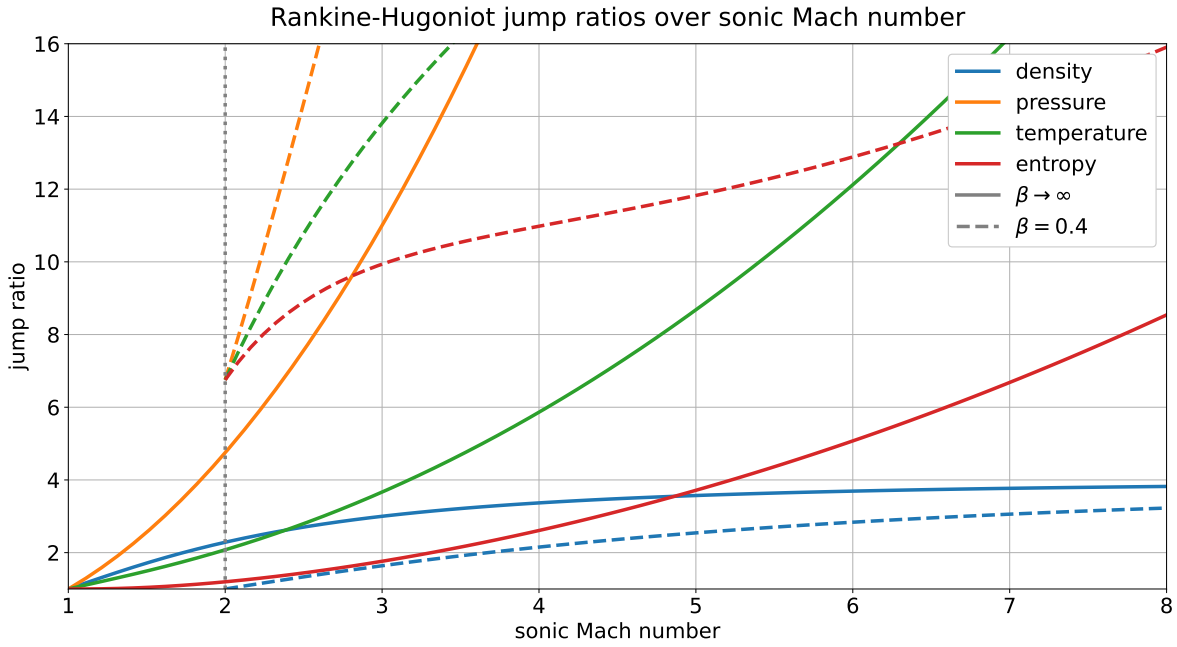

Figure 3.4: Rankine-Hugoniot jump ratios subject to the Mach number for the sonic regime ($\beta \to \infty$, solid lines) and a magnetic pressure dominated regime ($\beta = 2/5$, dashed lines) for various physical quantities and $\gamma = 5/3$. For the latter regime the physically feasible domain begins at Mach numbers $\mathcal{M} > 2$ (demarked by the dotted, gray vertical line).

We can also derive an explicit formula by inserting (3.47) into the density jump relation in (3.42) and setting $\frac{\rho^+}{\rho^-} = 1$. We get a "critical" sonic Mach number

$$\mathcal{M}_{\text{crit.}} = \sqrt{\frac{2}{\gamma\,\beta} + 1} \tag{3.50}$$

which distinguishes the subsonic regime ($\mathcal{M} < \mathcal{M}_{\text{crit.}}$) from the supersonic regime ($\mathcal{M} > \mathcal{M}_{\text{crit.}}$) in strongly magnetized plasmas ($\beta \ll 1$).

In this work, we are interested in detecting shocks in astrophysical settings. Hence, we have to develop algorithms that can specifically detect strong shocks. Figure 3.4 indicates which hydrodynamic quantities might be useful for achieving our goal. Clearly, the density jump flattens for high Mach numbers and is therefore not sensitive in the regime of strong shocks. Moreover, the entropy jump is also not suitable, since it is flat for lower Mach

numbers ($\mathcal{M} < 5$), especially in the hydrodynamic limit. On the other hand, the pressure and the temperature jumps are sensitive over the whole Mach number range, and represent good candidates for measuring the Mach number in simulations. For our shock capturing schemes, we rely on the pressure jumps being the most sensitive shock indicator for both, Euler and MHD regimes.

3.6 Multi-species Fluids

The ability to track the exact composition of a fluid or gas is of central importance in astrophysical simulations as they include detailed chemical reaction chains (chemical networks) to treat heating, cooling, as well as the formation and destruction of chemical compounds in order to mimic the behavior of ISM (Walch et al. 2015; Gatto et al. 2015; Glover and Clark 2014). In our model, the individual species or mass fractions $\sigma_s \in [0, 1]$ move with the same velocity $\vec{v}$ as the total, or sometimes called mixture, density ρ. The sum of all n_{spec} mass fractions maintains the total density at all times, i.e. $\sum_s^{n_{\text{spec}}} \rho\, \sigma_s = \rho$.

The chemical evolution of the gas consists of simplified chemical reaction networks that track the fraction of the gas, and the formation and destruction of, for example, H_2 and CO due to ionization and re-combination processes. Another commonly used chemical reaction network for modeling ISM, called NL97 (Nelson and Langer 1997), comprises of five different species: H,H^+,H_2,CO, and C^+. Clearly, NL97 models a partially charged fluid and magnetic forces act only on the charged particles and not on the neutral particles. An aspect, we do not account for in our fluid model of ideal MHD equations (3.22). However, if collisions between charged and neutral particles occur frequently, then we still have a good approximation. In this case, collisions will rapidly redistribute momentum between the charged and the neutral components of the plasma, and the end result is the same as if the magnetic forces acted on both types of fluids.

Species Evolution Equations

The chemical species are generally not in chemical equilibrium and therefore have to be solved by continuity equations of the form (Glover and Mac Low 2007; Micic et al. 2012)

$$\partial_t(\rho\,\sigma_s) + \nabla \cdot (\rho\,\vec{v}\,\sigma_s) = \mathrm{C}_s(\rho, T, \ldots) - \mathrm{D}_s(\rho, T, \ldots), \quad s = 1, \ldots, n_{\text{spec}}$$

with C_s and D_s representing the creation and destruction of species s due to chemical reactions. The conversion of a species generally depends on the density, temperature and on the abundances of the other chemical species. Therefore, a set of coupled PDEs for the mass densities of the different chemical species have to be solved. In practice, the problem can be made substantially easier to handle by splitting the chemical source and sink terms from the advection terms (operator splitting). With this approach, the continuity equations simplify to

$$\partial_t(\rho\,\sigma_s) + \nabla\cdot(\rho\,\vec{v}\,\sigma_s) = 0, \quad s = 1,\ldots,n_{\mathrm{spec}}. \tag{3.51}$$

Changes in the chemical composition of the gas resulting from chemical reactions are then computed in a separate chemistry step, during which the following set of coupled ordinary differential equations (ODEs) are solved:

$$\frac{\mathrm{d}}{\mathrm{d}t}(\rho\,\sigma_s) = \mathrm{C}_s(\rho, T, \ldots) - \mathrm{D}_s(\rho, T, \ldots), \quad s = 1,\ldots,n_{\mathrm{spec}}.$$

If the ODEs above require of much shorter timesteps than the (magneto-)hydrodynamical timestep, then sub-cycling is used within the chemistry step, hence avoiding the need to constrain the global timestep.

Multi-species Equations-of-state

The multi-species fluid model is usually generalized with a variable heat capacity ratio γ by adopting a weighted mean over all species (Murawski 2002):

$$\gamma = \frac{\sum_s^n c_s^{\mathrm{pres.}}\,\sigma_s}{\sum_s^n c_s^{\mathrm{vol.}}\,\sigma_s} \tag{3.52}$$

with the heat capacities for constant pressure $c_s^{\mathrm{pres.}}$ and constant volume $c_s^{\mathrm{vol.}}$ of each individual species. Gouasmi et al. (2020) derived an entropy consistent numerical flux for such a multi-species fluid modeled by weighted heat capacity ratios as in (3.52). The flux has a very similar structure to (3.41) and uses proper averaging of left and right states in order to ensure entropy consistency.

In this work, we neither construct nor implement any chemistry related numerical solvers. Hence, we rely on external software or modules, as part of the simulation framework FLASH, to handle this specific aspect of the simulation and we consider it as a “black

box". In our case, the standard FLASH framework readily offers inbuilt multi-species support with units taking care of the correct equation-of-state calculations (3.52). Our responsibility is to properly solve the advection PDEs (3.51) and to offer the total density, temperature and the vector of abundances to the chemistry code and get back a new temperature and a set of new abundances. In other words, the chemistry step is in full control of the EOS (3.4) and oblique to the hydrodynamics scheme. Consequently, any global entropy balancing via careful construction of entropy consistent numerical fluxes as done by Gouasmi et al. (2020) turns untenable. The only clear insight we can have is the entropy contribution by the hydrodynamics scheme, which we focus on in this thesis.

Numerical Flux

The simplest and most direct way of solving (3.51) is to make the Rusanov flux (2.27) ansatz

$$\boldsymbol{f}^{*,\mathrm{spec}} = \{\!\{\rho v \boldsymbol{\sigma}\}\!\} - \frac{1}{2}\lambda^{\max}\,[\![\rho\boldsymbol{\sigma}]\!]\,. \tag{3.53}$$

However, we have observed in our simulations, that chemical reaction networks can deplete a species to zero abundance at any point in time. Hence, numerical species fluxes with a structure similar to (3.53) break down for zero species densities and are also not suitable for our envisaged simulations. Furthermore, it is very important to avoid any negative abundances and at the same time to maintain the total density balance. One solution to these tricky constraints is to resort to a robust, fail-safe mass tracer approach. It reads

$$\boldsymbol{f}^{*,\mathrm{trace}} = \begin{cases} \{\!\{\rho v\}\!\}\ \boldsymbol{\sigma}^{+}, & \{\!\{\rho v\}\!\} > 0 \\ \{\!\{\rho v\}\!\}\ \boldsymbol{\sigma}^{-}, & \{\!\{\rho v\}\!\} < 0 \\ 0, & \text{otherwise}, \end{cases} \tag{3.54}$$

which we adopt in our implementation.

Mass Tracer Fields

Next to multi-component fluids, we also support mass tracer fields (also called mass scalars), which are advected analogously to (3.51). The implementation of mass tracer fields allows the use of any number of such fields, which makes it a flexible tool for tracing different mass quantities according to individual requirements. For example, a mass tracer field could be used to follow the distribution of metals in the ISM with virtually no additional costs.

An interesting simulation involving the modeling of a NL97 chemical network is presented in Section 7.5. A magnetized, turbulent molecular cloud gets shredded by a hot, supersonic galactic wind triggering the production of molecular hydrogen and carbon-monoxide.

3.7 Gravity

Next to turbulence, magnetic fields, and feedback from supernovae, molecular clouds are also shaped by gravitational forces. Over time an initial, relatively smooth distribution of matter, for example ISM, will collapse to form clumps of higher density, creating a hierarchy of condensed structures such as clusters of galaxies, stellar groups, stars and planets. Stars form when filaments and clumps inside MCs collapse under their own gravity. The compression caused by the collapse raises the temperature until thermonuclear fusion kick-starts, at which point the collapse gradually comes to a halt as the outward pushing thermal pressure counteracts the inward pulling gravitational force. The newborn star then exists in a state of dynamic equilibrium. Once all its energy sources are exhausted, the aged star will, depending on its size, collapse further until it reaches a new equilibrium state or explodes as a supernova.

The densest parts of the filaments or clumps are called molecular cores and their massive cores (dense molecular cores) can accumulate densities up to 10^6 particles per cubic centimeters. Astronomers can observe such cores by tracing the content of carbon monoxide (CO) while dense molecular cores are rich in ammonia (NH_3). Since the concentration of dust within molecular cores is sufficient to block light from background stars, they appear as dark nebulae. It is important to note, that MCs embedded in galaxies are not only molded by internal gravitational interactions, but also by the ambient gravitational field imposed on by the encompassing galaxy (Zuckerman and Evans 1974; Falgarone et al. 1991, 2008).

Gravity Source Term

The inclusion of gravity in the governing equations introduces a force into the right-hand-side (RHS) of the momentum equations

$$\partial_t(\rho\vec{v}) + \rho\,\nabla\phi = 0, \tag{3.55}$$

and the energy equation

$$\partial_t E + \rho \vec{v} \cdot \nabla \phi = 0. \tag{3.56}$$

The gravitational potential ϕ satisfies the Poisson's equation

$$\nabla^2 \phi = 2\pi G \rho$$

with gravitational constant G and mass distribution ρ in the computational domain Ω. The gradient of ϕ is then the gravitational acceleration $\nabla\phi = \vec{g} = (g_1, g_2, g_3)^T$ at each point in Ω. In this work, we do not solve the Poisson's equation ourselves and resort to external software or solver modules as part of the simulation framework FLASH. We expand on available gravity solvers in Section 6.3.4. External gravitational fields are usually given by explicitly defining an acceleration vector $\vec{g}_{\text{ext}}$.

Hydrostatic Balance

There is a class of problems, e.g., atmospheric flows and stellar structure simulations, where steady-state solution to systems of stratified objects in hydrostatic balance are important. The flow is static and the gravitational force is balanced by the pressure forces, i.e.

$$\nabla p = -\rho \nabla \phi \quad \text{and} \quad \vec{v} = \vec{0}. \tag{3.57}$$

Above ODE possesses non-trivial stationary solutions, which its precise balancing is not easy to achieve by numerical schemes. Conventional numerical schemes in which the gravitational source term may be discretized in a consistent manner are not able to preserve such stationary solutions especially on coarse meshes. This leads to erroneous numerical solutions especially when trying to compute small perturbations around the stationary solution necessitating the need for very fine meshes. However, in practical 3D simulations it may not be feasible to use very fine meshes. Moreover, even a very high order accurate scheme can lead to wrong prediction of small perturbations or even becomes unstable and crashes if the scheme is not well-balanced (Xing and Shu 2013). Hence, the balancing has to be achieved at the numerical level instead in order to maintain the stationary solution.

By contrast, simulations of turbulent MCs are highly un-steady and dynamic and do not resolve the internal structure of newly formed stars. Hence, well-balancedness of the numerical scheme is not a crucial requirement. In our implementation, we do not account for exact well-balancedness on numerical level. Nonetheless, Section 7.3 shows

that our code is capable of simulating a differentially rotating disk in initially hydrostatic equilibrium threaded with a toroidal magnetic field. Due to magneto-rotationally driven instabilities disk material gets ejected into the surrounding space similar to the solar flares emerging at the surface of our Sun.

Entropy Consistency

Recent findings by Waruszewski et al. (2021) have led to DG methods, which couple the gravity source terms in an entropy consistent manner. They recast the Euler equations with gravity in a more general, non-conservative balance law and construct arbitrary order flux differencing DG schemes. They also present an entropy conserving numerical flux for the Euler equations with gravity and also show how to extend the numerical flux in order to ensure entropy stability. Since the findings just came out at the time of writing, they could not enter in our research efforts and are left for future work.

Free-fall Time

Gravitational collapse is the contraction of an astronomical object under the influence of its own gravity (self-gravity). More and more matter is drawn inward towards the gravitational center till a hydrostatic balance is reached. Gravitational collapse is considered to be a fundamental mechanism for structure formation in our Universe. The free-fall time is the characteristic time that would take a body to collapse under its own gravitational attraction if no other forces existed to oppose the collapse. As such, it plays a fundamental role in setting the timescale for a wide variety of astrophysical processes in MCs. If we consider the infall of a spherically-symmetric, uniform distribution of mass M and radius R we get a density ρ of

$$\rho = \frac{3\,M}{4\,\pi\,R^3}.$$

According to Newton's law of gravity, the acceleration of gravity at any given distance R from the sphere's center depends only upon the total mass contained within R. Consequently, the free-fall time of a massless particle at R can be expressed solely in terms of the total mass M interior to it. In terms of the average density of the shell, the free-fall time t_{ff} then reads

$$t_{\mathrm{ff}} = \sqrt{\frac{3\,\pi}{32\,G\,\rho}} \propto \frac{1}{\sqrt{\rho}}. \tag{3.58}$$

In Figure 3.5 we show a log-log plot of the free-fall time (3.58) highlighting the huge range of density and timescales involved in molecular cloud models. Note that, still today no numerical code provides enough resolution in order to reach hydrostatic equilibrium (stars and planets), which is even far off the ranges shown in Figure 3.5. When there are no counteracting forces in such simulations once the gravitational collapse has begun, at some point in time a crash of the simulation due to insurmountable high densities and pressures is inevitable. There are two choices to tackle this issue. Either, one is only interested in the evolution close to the gravitational collapse and then stops the simulation at the right time or one resorts to so called sink-particle methods (Bate et al. 1995; Krumholz et al. 2004; Hubber et al. 2013; Bleuler and Teyssier 2014), which remove the excess mass at gravitational cores and inserts proxies respectively sink particles instead. In our simulations, we adopt the first approach.

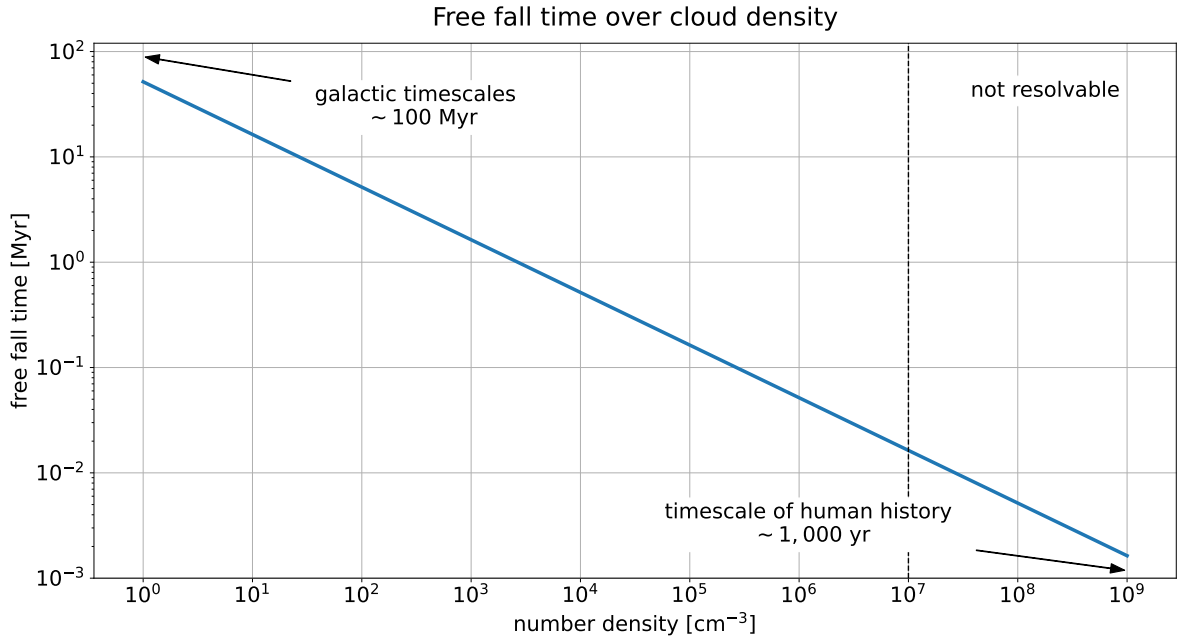

Figure 3.5: Free fall time t_{ff} of a molecular cloud at density ρ according to (3.58). We consider a range from 1 particle per ccm (ISM) to extremely dense molecular cores of over 10^9 particles per ccm. The free fall time ranges from tenths of millions of years (galactic timescales) down to extremely short period of a several thousand years. Structures with densities above 10^7 particles per ccm (dashed, vertical line) are considered not resolvable by our codes.

3.8 Radiative Transfer

MCs cool mainly by emitting electromagnetic radiation (photons), since they do not usually conduct or convect heat very efficiently. The mechanism by which this radiation occurs is mostly initiated by an excitation of an atomic, ionic, or molecular transition during a collision. In this excitation, gas particles gain their energy from the kinetic energy of the collision processes, and after some time, the excited system radiates this energy away in form of a photon escaping from the cloud. The gas loses kinetic energy, and thus it cools.

Since atomic hydrogen is the most abundant element in ISM, excitation of transitions in atomic hydrogen should be an efficient cooling mechanism. However, the transitions are so energetic (more than 10 eV above the ground state) that only at high temperatures (above 10^4 K) this mechanism starts to become effective. In other words, warm neutral media (WNM) is unable to cool efficiently below 10^4 K and stay warm over long time periods in the absence of other cooling mechanisms. An interesting consequence of this fact is, that ISM, which primarily consists of WNM, is in hydrostatic equilibrium and does not collapse under its own weight.

Jean's Criterion

Warm gas prevents clouds from collapsing since its internal pressure, or thermal energy, is strong enough to balance the gravitational forces. The cloud is in hydrostatic balance (3.57) and does not collapse, and thus does not form stars for temperatures above 10^4 K. However, if the gas is dense enough or cool enough, the cloud collapses under its own weight. This relation is quantified by the Jeans instability, which formulates the criterion for a cloud of gas to collapse through equating pressure forces to gravity:

$$L_{\mathrm{J}} = \sqrt{\frac{15\, k_{\mathrm{B}}\, T}{4\pi\, G\, \mu\, m_{\mathrm{p}}\, \rho}} \quad \propto \sqrt{\frac{T}{\rho}}. \tag{3.59}$$

Here, k_{B}, μ, and m_{p} are Boltzmannn's constant, the mean molecular weight and the proton mass. The so-called Jeans' length L_{J} is the critical length scale of the cloud below which it collapses under given temperature T and density ρ.

Note the similarity of formula (3.59) with the one for the free-fall time (3.58) considering that for the latter we assumed no counteracting forces to gravity. If we plot the

Jeans' length over the cloud density for a constant temperature $T = 10^4$ K, as we did in Figure 3.6, we get an estimate for the expected length scales a typical molecular cloud simulation has to cover. The y-axis in Figure 3.6 is scaled in powers of two, which allows to directly infer the necessary levels of refinement (in case of octree-based meshes) in order to properly resolve the small-scale structures in the simulation. This plot makes very clear why resolution requirements for typical molecular cloud simulations span ten refinement levels or even more.

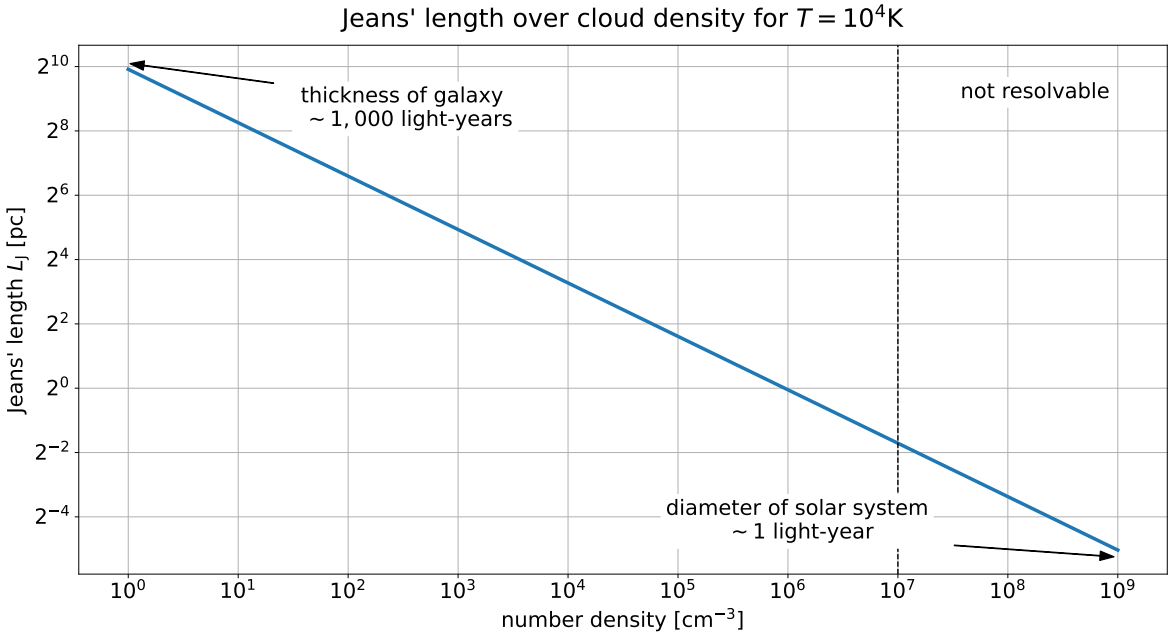

Figure 3.6: Jeans' length L_J of a molecular cloud at density ρ and constant temperature $T = 10^4$ K according to (3.59). We consider a range from 1 particle per ccm (ISM) to very dense molecular cores of over 10^9 particles per ccm. The Jeans' length is plotted in $\log_2$-scale and ranges from 2^{10} pc = 1024 pc down to 2^{-4} pc = 0.0625 pc. Structures with densities above 10^7 particles per ccm (dashed, vertical line) are considered not resolvable by our codes.

Stellar Feedback

There are various processes that heat the ISM to even millions of degrees. Cosmic rays are an efficient heating source able to penetrate in the depths of MC. They transfer energy to gas through both ionization and excitation. Ultraviolet radiation emitted by hot stars can remove electrons from dust grains. The photon is absorbed by the dust grain, and some of its energy kicks out an electron from the grain's surface. The remainder of the photon's energy gives the ejected electron kinetic energy which in turn heats the gas

through collisions with other particles. Molecular hydrogen (H_2) forms on the surface of dust grains, where the surface acts as a catalyst. This process is exothermal, heating the dust grain and surrounding gas. Moreover, transient events such as stellar winds, explosions of supernovae, and compressional and frictional heating (shocks) substantially increase the temperature of the gas. Obviously, perpetual heating prevents the ISM from cooling and disrupts the formation of stars. Hence, a good understanding of all heating mechanisms present in MCs is a crucial cornerstone towards a complete theory of star formation. In this work, we particularly focus on two feedback mechanisms influencing the dynamics in their host clouds: expanding HII-regions and supernovae.

Expanding HII Regions

If a massive ($>$ 10 solar mass) star is created in a molecular cloud, it radiates intense ultraviolet radiation into the cloud resulting in the creation of an expanding photo-ionized region, called an HII region. Such regions absorb all photons with energies larger than 13.6 eV which is exactly the ionization potential of hydrogen atoms. The photo-ionized HII region is bounded by an ionization front, which separates the ionized from the neutral gas driving a strong shock into the surrounding, neutral ISM. The supersonically expanding HII regions not only inhibit further star formation, but can also even destroy the host cloud (Dale and Bonnell 2011; Walch et al. 2012; Geen et al. 2015). Since it is assumed that our Milky Way Galaxy maintains a steady number of stars over galactic timescales ($\propto$ 100 Myr), MCs must be perpetually created at a rate that compensates the destruction by such massive stars.

For simulation codes, the simulation of expanding HII regions driven by stellar radiation involves a challenging combination of fluid dynamics, radiative transfer, micro-physical heating, cooling, ionization and recombination. In Section 7.4 we present the results of a simulation involving the growth of a HII bubble into turbulent ISM due to radiative feedback from a massive star.

Supernovae

A supernova (SN) is a powerful and bright stellar explosion. This transient astronomical event occurs during the last evolutionary stages of a massive star. The original object, called the progenitor, either collapses to a neutron star or black hole, or is completely destroyed. The optical luminosity (brightness) of a SN at its peak can be comparable to

that of an entire galaxy before fading over several weeks or months.

SN can expel several solar masses of material at speeds up to several percent of the speed of light. This drives an expanding shock wave into the surrounding ISM, sweeping up an expanding shell of gas and dust observed as a supernova remnant (SNR). These events are a major source of heavier elements in the ISM such as oxygen and iron. The expanding shock waves of SN can either trigger or inhibit the formation of a new generation of stars.

So-called type Ia category SN produce fairly consistent peak luminosities because of fixed critical masses at which white dwarfs will explode. Their consistent peak luminosities allow them to be used as standard candles in order to estimate the distance between Earth and their host galaxies. In Section 7.2 we simulate a specific specimen of an Ia SN identified as SN 1572. It became visible in the night sky in the year 1572 and was first described by Tycho Brahe a renowned Danish astronomer.

3.9 Final Remarks

In this chapter, we gave a brief overview of the astrophysical model we are employing in the simulations shown in this work. We introduced the governing equations, investigated the underlying fluid model, discussed multi-physics aspects like multi-component fluids, gravity, radiative transfer and derived the scales in space and time characteristic for state-of-the-art molecular cloud simulations.

Guided by our investigations about the employed astrophysical model, we deduce the following bucket list of numerical ingredients a “perfect” fluid solver optimally would provide to deliver an accurate and physically faithful approximation of the model.

- High (order) accuracy in smooth, well-resolved flow regions.
- Robust and non-oscillatory (monotone) handling of (very) strong shocks.
- Stable computation of under-resolved flow conditions, especially under near-vacuum conditions.
- Besides conservation of primary quantities (mass, linear momentum, energy), conservation of derived (secondary) properties such as kinetic energy, entropy, and vorticity.
- Adherence to additional constraints specified by the model, e.g. zero divergence in

the magnetic field.

- Solid entropy stability under all possible low and high Mach flow states. Clean and rigorous consideration of a wide range of EOS (isothermal, polytropic, barotropic, etc.), multi-species models, chemical networks, and source terms, e.g., gravity and radiation pressure.
- Conservative and positivity-preserving advection of multi-species flows. Some abundances can be zero.
- Maintaining flow configurations in hydrostatic balance, i.e. well-balancedness.
- Performant and scalable implementations allowing simulations with large dynamical ranges in space and time.

Clearly, the amalgamation of above specifications is a tough challenge considering the extremely wide range of claimed properties to consider. Moreover, some properties are actually mutually exclusive; for example, high order accuracy and monotone resolution of flow discontinuities. Zero abundances and proper entropy consistency are problematic considering, for example, that logarithmic means of the left and right densities are a crucial building blocks in entropy consistent numerical fluxes (Gouasmi et al. 2020). Furthermore, fixation on a special EOS for the entropy analysis or the choice of a specific gravitational field configuration for hydrostatic balance inhibits the versatility of any fluid solver; one of the design premises in our research efforts.

Throughout this thesis, we devise and discuss our own solutions and reference to existing approaches in the literature individually addressing every aspect in the list. However, a "clean" and rigorous incorporation of all the aspects into one solver framework is far beyond the scope of this thesis. Instead, we relax our requirements and take "compromises", which we document and discuss in this work.

Chapter 4

Numerical Scheme

4.1 Introduction

In this chapter, we provide the formulation of Finite Volume (FV) schemes and discontinuous Galerkin (DG) methods for systems of conservation laws (2.6) on Cartesian grids. Based on both schemes, we then devise a convex blending scheme which aims to combine the robustness of FV with the accuracy of DG.

The computational domain $\Omega \in \mathbb{R}^3$ is discretized using $Q \in \mathbb{N}$ non-overlapping control volumes $\Omega_q \subset \Omega$ on which we impose a discrete version of the conservation law at hand. In this thesis, we use the following notation interchangeably.

$$\vec{x} = (x_1, x_2, x_3)^T = (x, y, z)^T.$$

When we assume a Cartesian grid and subdivide the physical domain Ω into Q blocks of size

$$\Delta\vec{x}_q = (\Delta x_q, \Delta y_q, \Delta z_q)^T.$$

The FV method is a classical scheme for solving the hydrodynamic equations of fluid dynamics with the reputation of being robust and easy to implement while achieving at least second order accuracy. FV methods represent the solution $\boldsymbol{u}(\vec{x}, t)$ as mean values in space

$$\overline{\boldsymbol{u}}_i(t) = \frac{1}{|\mathcal{V}_i|} \int_{\mathcal{V}_i} \boldsymbol{u}(\vec{x}, t)\, \mathrm{d}\vec{x} \tag{4.1}$$

pinned at the center $\vec{\mu}_i \in \Omega$ of delimited (finite), non-overlapping volumes $\mathcal{V}_i \subset \Omega$ which we name cells. Each quantity that represents a cell-averaged (or face-averaged) mean value is written with a bar on top. The mean fluid values are evolved by solving the fluid dynamics equations in an integral, or weak, form. The strength of this method lies in the capability of capturing shocks and discontinuities and their strict conservation of mass, linear momentum, and total energy within the computational domain.

On the other hand, DG methods approximate their solution as piecewise-continuous polynomials living within non-overlapping adjacent elements Ω_q

$$\boldsymbol{u}(\vec{x},t) \approx \boldsymbol{p}_q(\vec{x},t) = \sum_{ijk=0}^{n-1} \widetilde{\boldsymbol{c}}_{q,ijk}(t)\, x^i\, y^j\, z^k \tag{4.2}$$

with n^3 time-dependent polynomial coefficients $\widetilde{\boldsymbol{c}}_{q,ijk}(t)$ per element. Finding the right polynomial coefficients to the exact solution $\boldsymbol{u}(\vec{x},t)$ is called interpolation problem and always has a unique solution according to the interpolation theorem provided the input data sits on pairwise distinct points, called interpolation nodes, in space. The polynomial approximation of the exact solution has a couple of advantages. First, one can easily differentiate and integrate in space within the boundaries of element Ω_q with high accuracy:

$$\partial_x\, \boldsymbol{u}(\vec{x},t) \approx \partial_x \boldsymbol{p}_q(\vec{x},t) = \sum_{i=1,jk=0}^{n-1} \widetilde{\boldsymbol{c}}_{q,ijk}(t)\, x^{i-1}\, y^j\, z^k \tag{4.3}$$

and

$$|\Omega_q|\, \overline{\boldsymbol{u}}_q(t) \approx \int_{\Omega_q} \boldsymbol{p}_q(\vec{x},t)\, \mathrm{d}\vec{x} = \sum_{ijk=0}^{n-1} \widetilde{\boldsymbol{c}}_{q,ijk}(t) \int_{\Omega_q} x^i\, y^j\, z^k\, \mathrm{d}\vec{x}. \tag{4.4}$$

Obviously, the derivatives resp. integrals of the monomials in (4.3) and (4.4) are easy to evaluate. However, in cases where the exact solution is highly irregular, for example it contains jump discontinuities or is very erratic, then the polynomial approximation of the data can turn out to be very problematic. The polynomial turns very oscillatory and may over- or undershoot (over- or underestimate) the input data considerably. This problem is called Gibbs phenomenon. It reflects the difficulty inherent in approximating a discontinuous function by a finite series of continuous polynomials.

The rest of this chapter is structured as follows. We first discuss the two main sources of numerical instabilities, shocks and aliasing. After introducing the basics of our employed time integration methods, we detail the building blocks of FV and DG schemes. The

last part of the chapter is devoted to our newly developed convex blending scheme. A common theme threading throughout this chapter is the special focus on different entropy consistency/stability approaches and their influence on the robustness of the numerical scheme. After a method has been introduced we immediately present some numerical results and assess their "performance". This, so we hope, gives a comprehensible chain of reasoning about our final choice of our employed numerical scheme. The numerical results shown in this chapter have been obtained with our prototyping code Nemo, which we discuss in the subsequent Chapter 5.

4.2 Shock Capturing

We know that for general nonlinear systems of hyperbolic conservation laws, discontinuities in terms of shocks may develop in finite time regardless on the smoothness of the initial data, as already investigated in Section 3.5. Godunov (1959)'s famous theorem states that numerical schemes with formal accuracy higher than first order exhibit large spurious oscillations near discontinuities in the solution (Wesseling 2009). Consequently, the absence of monotonicity preservation in the evolving solution might lead to catastrophic numerical instabilities or even nonphysical solution states, e.g. negative density or pressure.

Oscillations in the numerical solution $\overline{\boldsymbol{u}}$ at a point in time t_n can be measured by the total variation, which is given by

$$\mathrm{TV}(\overline{\boldsymbol{u}}(t_n)) = \sum_i |\overline{\boldsymbol{u}}_{i+1}(t_n) - \overline{\boldsymbol{u}}_i(t_n)|. \tag{4.5}$$

For second order schemes, like FV with appropriate reconstruction, several techniques have been developed to alleviate the oscillations in the numerical solution near discontinuities. The class of total variation diminishing (TVD) limiters have proven to be very robust without overly compromising the accuracy in smooth flow regions. Harten (1997) showed that a numerical scheme, which is TVD, is also monotonicity preserving - a key feature to properly handle discontinuities. A numerical scheme is said to be TVD if the total variation does not increase with time:

$$\mathrm{TV}(\boldsymbol{u}(t_{n+1})) \leq \mathrm{TV}(\boldsymbol{u}(t_n)). \tag{4.6}$$

The most popular TVD limiter for FV schemes is the minmod limiter (Roe 1986) which we further discuss in Section 4.5.6.

Stabilizing high order DG methods at shock-driven discontinuities, on the other hand, is a tricky problem and up-to-now no perfect method has been found. Methods for high order numerical schemes that intervene at shocks and regularize the numerical solution are called shock capturing methods. A shock capturing method for DG usually consists of two main parts. In the first part, elements that contain oscillating polynomials have to be identified and then be treated in the second part to reduce the oscillations or even, as a last resort, completely replace the solution with an oscillation-free solution coming, for example, from a second order TVD scheme. Clearly, the shock detection step is most crucial for the accuracy of the scheme, as well as the computational efficiency.

In general, there is no golden rule for shock capturing in a high order context. In the numerics community many ideas and approaches have been developed for oscillation control since interest in these methods emerged. What follows is a brief discussion of the most popular shock capturing approaches for DG schemes.

Slope Limiting / Low-Pass Filtering

For a high order scheme, the higher frequency modes of the solution polynomial can be limited directly, as was done by Biswas et al. (1994); Krivodonova (2007); Cockburn and Shu (1998); Cockburn et al. (1990); Kuzmin (2012). The approach can be seen as a generalization of the minmod limiter, sequentially applied to the higher modes or moments. Moe et al. (2015) devised a limiter based on ideas by Barth and Jespersen (1989) and the maximum principle preserving (MPP) framework Zhang and Shu (2011). All higher modes of the polynomial are reduced such that the limited solution fits within the minimum and maximum bounds of the solutions in the direct neighborhood of the troubled element. The well-known positivity preserving (or rather restoring) limiter for DG by Zhang and Shu (2012) limits the moments such, that the solution is lifted into the convex set of physically permissible states.

Filtering, on the other hand, is inspired by methods in signal processing, in which noisy data and ringing are smoothed a posteriori. Panourgias and Ekaterinaris (2015) applied nonlinear filters locally within the DG elements based on previous work by Yee and Sjögreen (2006, 2007) for finite differences schemes in MHD flow regimes. Similar to mode limiters, filters directly modify selective DOF and provide some sort of shock capturing

by removing or damping the oscillating solution polynomial. The filter framework by Hesthaven and Kirby (2008) can also be utilized for shock capturing purposes. More recent developments in the area of solution filters for DG are presented for example in Bohm et al. (2019) and Zala et al. (2021) where, unfortunately, mass conservation is not naturally assured.

Slope limiting and filtering within the DG context have limitations, especially when it comes to strong shocks, as they arise in astrophysical applications. Such methods eventually reduce the solution to first order, destroying the accuracy of the overall computation. The shock resolution capability is not based on the n^{th} order polynomial resolution $\sim \frac{\Delta x}{n}$, but only on the whole element size Δx which gets larger and larger with higher polynomial degree $n-1$. Considering that a FV discretization with adequate reconstruction typically resolves a shock within about 2-3 cells, the shock width for high order DG schemes with large elements can become very wide. Numerical experiments with some of these limiters and for very strong shock simulations led to severe numerical artifacts in the solution. For example, the shock front was resolved by a checkerboard like pattern due to completely flattened DG elements. Tedious parameter-tuning sometimes could alleviate the issue for specific setups, but did not generalize to a broad range of simulations.

Artificial Viscosity / Artificial Diffusion

Sub-element resolution of a shock that scales proportional to $\frac{\Delta x}{n}$ can be achieved with artificial viscosity. The idea is to locally add viscosity scaled by the resolution length as well as the polynomial degree, such that shocks can be captured locally in single DG elements. The discontinuity is ground into a sharp, but smooth profile such that high order polynomials can resolve it. Using explicit artificial viscosity for shock capturing dates back to VonNeumann and Richtmyer (1950) and is now a popular approach for DG methods (Persson and Peraire 2006; Yu and Yan 2013; Zingan et al. 2013; Abbassi et al. 2014). Some form of troubled cell indicator is introduced to not only flag the troubled element that contains the shock but also to determine the amount of necessary viscosity, e.g., based on the amount of energy in the highest polynomial modes (Persson and Peraire 2006) or the local entropy production (Zingan et al. 2013). Counterintuitively, shocks can be more effectively captured within a single DG element if the polynomial order is sufficiently high enough. Persson and Peraire (2006) showed that less viscosity is needed the higher the polynomial degree of the DG discretization, which in turn leads to sharper shock profiles. In general, the method is completely local in case the oscillation detection

process only uses local data within the DG element. This approach is not only well-suited for DG, but has also proven to give reasonable results for a broad variety of shock tests.

A pressing issue of artificial viscosity, however, is that a high amount of artificial viscosity is needed for very strong shocks. This makes the overall discretization very stiff with a very small explicit time step restriction. Local time stepping (Gassner et al. 2015), specialized many stage Runge-Kutta schemes with optimized coefficients for strongly dissipative operators (Klöckner et al. 2011) or implicit-explicit Runge-Kutta schemes for stiff relaxation problems (Jin 1995) might be useful in order to solve this problem. Moreover, the absence of any explicit dissipation mechanism in our fluid model (3.29), which we could utilize to induce artificial diffusion. The aforementioned issues would introduce a layer of complexity into the scheme, which we did not deem feasible for our project.

Substitution / Switching

Shock capturing approaches such as the (Hermite) Weighted Essentially Non-Oscillating ((H)WENO) limiters (Zhu and Qiu 2009; Qiu and Zhu 2010; Guo et al. 2015) replace the DG polynomial by an oscillation free reconstruction using data from neighboring elements. These WENO-type (Liu et al. 1994) limiters use the ideas of the Lagrangian (WENO) and Hermitian (Hermite WENO) interpolation, as well as the construction of interpolants via linear combination of a selection of the smoothest solution reconstructions. Similar to the element based slope limiters above, (H)WENO limiters are effective for low order DG discretizations only, as its accuracy strongly degrades with increasing polynomial order n, Despite, using a formally high order reconstruction, its leading discretization parameter is still Δx and not $\frac{\Delta x}{n}$ and hence the shock width still scales with Δx.

An alternative switching approach that achieves sub-element resolution at shocks is to replace the flagged DG elements by FV subcells. When switching to a FV method, it is assumed that the inherent numerical dissipation of the FV scheme is enough to clear all oscillations at even the strongest shocks. Second order TVD (Sonntag and Munz 2014; Vilar 2019) or appropriate high order reconstructions (Zanotti et al. 2015) allow a sharp shock resolution while keeping the DOF in switched DG elements constant. In Dumbser et al. (2014); Dumbser and Loubere (2016), the accuracy of the FV method is further enhanced via a respective increase in local grid resolution by increasing the DOF in troubled DG elements.

If, however, the whole high order DG element accidentally switches to the subcell FV

scheme in smooth parts of the solution, accuracy might strongly degrade. The FV scheme, even on the fine subcell grid, might wash away any fine structure details that the high order DG method simulated. From this point of view, a high order subcell FV scheme is highly desired in this context. A downside of any high order FV scheme is the non-locality of the data dependence due to the wide reconstruction stencils. This is especially cumbersome close to other high order DG elements, where the reconstruction stencils reach into neighboring elements and thus fundamentally change the data dependency footprint of the resulting implementation. This, consequently has a derogatory impact on the implementation complexity and the parallelization of the code.

Smooth Transition / Convex Blending

Instead of hard switching between different schemes, a smooth linear combination, which we coin "blending", aims to achieve the highest possible accuracy in every DG element. Assuming that the subcell low order FV approximation has enough inherent dissipation to capture shocks in an oscillation free manner, the goal is to give the FV discretization sufficient weight near shocks while smoothly transition back to higher order DG away from the shock. Hennemann et al. (2021); Rueda-Ramírez et al. (2021, 2022b) constructed entropy stable, convex blending FV-DG methods for compressible Euler and MHD.

In contrast to most shock capturing approaches for DG, where one indicator is used for the whole DG element, Markert et al. (2021) introduces multiple sub-element indicators that are adaptively adjusted inside the DG element. Similar to WENO schemes, where adaptive stencils are applied in order to adjust the smoothness of the approximation, "multi-level blending" aims to convex combine multiple orders of sub-element DG schemes within a top-level element according to smoothness selection rules. To achieve this, the DG element is firstly interpreted as a collection of available mean value data. Local reconstructions allows to define a hierarchy of approximation spaces, from pure piecewise constant approximations (subcell FV) up to a smooth global high order polynomial (DG element), with all piecewise polynomial combinations (sub-element DG) in between.

This shock capturing approach for high order DG has shown to be robust under very strong astrophysical shock conditions, while retaining the beneficial properties of DG as much as possible and keeping physical quantities, like density and pressure, positive. Most importantly, however, the method of Markert et al. (2021) is directly compatible with FLASH's block-based datastructure, and serves as a baseline approach for our choice

of shock capturing scheme described in Section 4.7.

4.3 Anti-Aliasing

High order DG methods are appreciated for their spectral like properties and their very low dispersion and dissipation errors (e.g. Ainsworth 2004; Gassner and Kopriva 2011; Dobrev et al. 2012; He et al. 2020). Low numerical dissipation is important to reduce artificial damping and heating, in addition, low dispersion errors are equally important as it guarantees high accuracy for wave propagation and interaction.

But these beneficial properties come at the cost of being prone to another source of numerical instabilities, namely aliasing. Aliasing is caused by under-resolution of e.g. turbulent vortical structures and can lead to instabilities that may even crash the code. As a cure, de-aliasing mechanisms are introduced in the DG methodology based on e.g. filtering (Kenevsky 2006; Hesthaven and Warburton 2008), polynomial de-aliasing (Kopriva 2017; Spiegel et al. 2015; Kopriva et al. 2019) or analytical integration (Kirby and Karniadakis 2003; Gassner and Beck 2013), split forms of the nonlinear terms that for instance preserve kinetic energy (Winters et al. 2018; Gassner et al. 2016a; Gassner 2014; Flad and Gassner 2017), and entropy (Carpenter et al. 2014; Wintermeyer et al. 2018; Pazner and Persson 2019; Chen and Shu 2017; Parsani et al. 2016; Murman et al. 2016; Chan 2018; Parsani et al. 2015; Liu et al. 2018a; Bohm et al. 2018; Gassner et al. 2018; Gassner 2013; Friedrich et al. 2018).

Abgrall (2018) develops a general framework to construct numerical schemes satisfying additional conservation relations, such as entropy conservation. Algebraic (anti-) diffusion-like terms correct the entropy error erroneously produced by the numerical scheme at each timestep, making it another interesting approach to achieve entropy stability for DG.

Entropy stable DG methods aim to produce physically meaningful approximations, which is especially desired in the presence of turbulence and near shocks. Entropy stability is seen as a promising candidate to cope with the instability issues of high order DG methods due to under-resolved sub-sonic flow structures without ruining the low dispersion and dissipation properties of DG. Second order TVD FV schemes, on the other hand, induce generally enough dissipation to overcome instabilities from under-resolved flow structures; however at the price of smearing out the numerical solution over time.

In astrophysical simulations, irregular solutions are a common issue, since by today's stan-

dards most numerical models are gravely under-resolved considering the vast amount of scales. Hence, our numerical scheme must handle under-resolvment well for our envisaged applications.

4.4 Time Integration

To evolve the fluid dynamics equations in time t and inside spatial sub-domain Ω_q, we integrate (2.6) from one point in time t_0 to another point in time t_1

$$\int_{t_0}^{t_1} \int_{\Omega_q} \partial_t \boldsymbol{u}(\vec{x},t)\, \mathrm{d}\vec{x}\, \mathrm{d}t = -\int_{t_0}^{t_1} \int_{\Omega_q} \sum_d^3 \partial_{x_d} \boldsymbol{f}_d(\boldsymbol{u}(\vec{x},t))\, \mathrm{d}\vec{x}\, \mathrm{d}t, \tag{4.7}$$

and evaluate the left-hand-side of above equation exactly. We write

$$\boldsymbol{u}(t_1)\big|_{\Omega_q} = \boldsymbol{u}(t_0)\big|_{\Omega_q} + \int_{t_0}^{t_1} \dot{\boldsymbol{u}}(t)\big|_{\Omega_q}\, \mathrm{d}t \tag{4.8}$$

where

$$\dot{\boldsymbol{u}}(t)\big|_{\Omega_q} = -\int_{\Omega_q} \sum_d^3 \partial_{x_d} \boldsymbol{f}_d(\boldsymbol{u}(\vec{x},t))\, \mathrm{d}\vec{x} \tag{4.9}$$

is the solution of the spatial fluxes, the right-hand-side, within sub-domain Ω_q. The numerical integration of the right-hand-side is covered in Section 4.5 and following. The numerical solution of the time integral in (4.8) is done with appropriate time integration methods.

For this purpose, there are many numerical procedures available in the literature like explicit, implicit or implicit-explicit time integrators. Whereas the latter two are well-suited for steady problems with possibly larger time steps (Kopriva and Jimenez 2013), an explicit solver is a more preferable choice for our applications, since the solutions of our ideal GLM-MHD equations (3.22) are compressible and advection dominated. Additionally, explicit timestepping methods are straightforward to implement and data-local in nature aiding the development of efficient codes for highly parallelized computing. We note, that investigations on other suitable time integrators for DG methods is ongoing research such as local time stepping methods (Gassner et al. 2011; Winters and Kopriva 2014) or stable space-time DG methods (Friedrich et al. 2019).

The simplest and most straightforward method to numerically integrate (4.8) is the explicit, first order Euler method. Given a timestep $\Delta t = t_1 - t_0$, the solution is evolved

from t_0 to t_1 by

$$\boldsymbol{u}(t_1) = \mathrm{E}(\boldsymbol{u}, t_0) \tag{4.10}$$

with

$$\mathrm{E}(\boldsymbol{u}, t) = \boldsymbol{u} + \Delta t \; \dot{\boldsymbol{u}}(t). \tag{4.11}$$

Euler timestepping however has very poor accuracy and is not stable for higher order DG methods, since the spatial operator spectra of DG (Krivodonova and Qin 2013) lie not entirely within the stability region of the explicit Euler method.

In our work, we use the third order, four stages, low-storage and strong-stability preserving Runge-Kutta scheme (SSP-RK(4,3)) devised by Spiteri and Ruuth (2002). Runge-Kutta schemes with strong-stability preserving property are composed of linear combinations of first order Euler steps. It enhances stability by better suppressing oscillations in the solution (Gottlieb et al. 2001) and ensures positivity of the solution provided the spatial discretization is also constructed to preserve positivity. For reference, we will list the SSP-RK(4,3) scheme, which has proven to be a very good compromise between accuracy, stability and performance:

$$\begin{aligned}
\boldsymbol{u}_{(1)} &= \frac{1}{2}\,\boldsymbol{u}_n + \frac{1}{2}\,\mathrm{E}(\boldsymbol{u}_n, t_n) \\
\boldsymbol{u}_{(2)} &= \frac{1}{2}\,\boldsymbol{u}_{(1)} + \frac{1}{2}\,\mathrm{E}(\boldsymbol{u}_{(1)}, t_n + \Delta t/2) \\
\boldsymbol{u}_{(3)} &= \frac{2}{3}\,\boldsymbol{u}_n + \frac{1}{6}\,\boldsymbol{u}_{(2)} + \frac{1}{6}\,\mathrm{E}(\boldsymbol{u}_{(2)}, t_n + \Delta t) \\
\boldsymbol{u}_{n+1} &= \frac{1}{2}\,\boldsymbol{u}_{(3)} + \frac{1}{2}\,\mathrm{E}(\boldsymbol{u}_{(3)}, t_n + \Delta t/2).
\end{aligned} \tag{4.12}$$

For the convergence tests, we resort to the very accurate fourth order, five stages and low-storage LS-RK(5,4) by Carpenter and Kennedy (2000) in order to rule out any significant error contributions by the time discretization.

A numerical scheme converges if it is stable and consistent. Consistency means that the local approximation of the solution is correct up to the targeted approximation order, and stability indicates that the error made in an individual timestep does not grow exponentially fast. A necessary condition for stability of an explicit scheme is the limitation of oscillations, which we already discussed in the previous section, another one is the Courant-Friedrichs-Lewy (CFL) timestep condition (Courant et al. 1928).

From Section 2.5 we know that hyperbolic conservations laws have a finite speed of signal propagation, and thus the solution at a point $(\vec{x}, t)$ has a finite domain of dependence obtained by back-tracing the characteristic lines passing through this point. The CFL condition ensures that the numerical domain of dependence contains the true domain of dependence. This is a crucial (necessary) condition used to prove the stability of numerical schemes. Rigorous estimates of CFL conditions are generally proved for the linearized problem, which act as guiding principles to choose the timestep for the nonlinear problem (Warburton and Hagstrom 2008; Toulorge and Desmet 2011; Chalmers et al. 2014).

To calculate a stable timestep an estimate of the maximum eigenvalue $\lambda^{\max}$ and an associated minimum distance measure $|\Delta\vec{x}|$ in the whole domain is applied. Then the maximal timestep is estimated by the CFL condition

$$\Delta t := \frac{CFL}{d} \min_{\Omega} \frac{|\Delta\vec{x}|}{\lambda^{\max}}, \tag{4.13}$$

where dimension $d = 3$ and the Courant-Friedrichs-Lewy constant $CFL \in (0, 1)$.

4.5 Finite Volume Scheme

In this section we describe the details about the formulation of finite volume schemes for systems of hyperbolic equation laws. For such methods, the computational domain Ω is discretized using non-overlapping control volumes, or cells, on which a discrete version of the conservation law is posed on each cell. For two and higher dimensional problems, the integral of the flux on the boundary of the cells also needs to be approximated using suitable quadrature rules. High order FV methods are obtained by using a high order quadrature formula and by appropriate reconstruction, e.g., the piece-wise parabolic method (PPM) (Colella and Woodward 1984), of solution values at the boundary quadrature points using neighboring cell-average values. In this thesis, we want to construct robust second order TVD FV schemes on Cartesian meshes, hence the midpoint rule applied at cell faces is sufficient.

Godunov proposed a method for constructing FV schemes, which are entropy stable, where at each interface between two adjacent cells, a local Riemann problem is formulated and solved exactly with the left and right states given by the cell average values $\overline{\boldsymbol{u}}_i$ and $\overline{\boldsymbol{u}}_{i+1}$ respectively. For the Godunov scheme, the flux integrals are evaluated exactly by using the solution to the local Riemann problem (2.23) centered at the interface $i + \frac{1}{2}$. From

Section 2.5 we know that the solution to our Riemann problems have self-similar structures allowing the interface fluxes to be calculated with approximate Riemann solvers. Such Riemann solvers efficiently compute an approximate solution for the Riemann problem, which is preferred to the expensive evaluation of the exact fluxes.

A second order FV scheme can be obtained by calculating a slope for every fluid quantity and every cell, based on the fluid states of neighboring cells. With this information one obtains a linear reconstruction of the solution inside the cell, and most importantly, at the cell boundaries. If these improved solutions at the cell boundaries enter the Riemann solver, a scheme with quadratic convergence in space is achieved. That is, the global error of the solution decreases quadratically with decreasing cell size.

After the reconstruction step, the numerical solution is evolved by calculating the fluxes and updating the mean conserved quantities. The last step is also known as averaging, since the reconstructed solution is not further taken into account and we are only interested in the new means at the new timestep. The steps discussed above can be summarized as a Reconstruct-Evolve-Average (REA) scheme. The hydrodynamical system is continuously evolved in time by consecutive REA steps. The generalization to two or three spatial dimensions can be accomplished straight-forwardly, either by a dimensional splitting, e.g., Strang (1968) splitting, or by applying the fluxes in the different directions simultaneously. The latter are called unsplit methods.

4.5.1 Building Blocks

For our convex blending scheme, described later in Section 4.7, it is convenient to group the cells into collective datastructures called blocks each containing $N \times N \times N$ cells. We associate these blocks with our control volumes $\Omega_q \in \Omega$ introduced at the beginning of this chapter. Each block Ω_q with midpoint $\vec{x}_q = (x_q, y_q, z_q)^T$ and size $\Delta\vec{x}_q = (\Delta x_q, \Delta y_q, \Delta z_q)^T$ is mapped to the reference space $[-1/2, 1/2]$ as

$$\vec{x}(\chi) = \vec{x}_q + \vec{\chi}\,\Delta\vec{x}_q\,, \quad \vec{\chi} \in \left[-\frac{1}{2}, \frac{1}{2}\right]^3 . \tag{4.14}$$

The $N \times N \times N$ regular subcells in block Ω_q have a size of

$$\left(\frac{\Delta x_q}{N}, \frac{\Delta y_q}{N}, \frac{\Delta z_q}{N}\right)^T =: \frac{\Delta\vec{x}_q}{N}$$

and bundle the cell-averaged fluid states $\overline{\boldsymbol{u}}(\vec{x},t)$ at subcells $\mathcal{V}_{q,\vec{i}}$ with centers $\vec{\mu}_{q,\vec{i}}$ and corners $\vec{\mu}_{\vec{i}\pm\frac{1}{2}}$ in reference space given by

$$\vec{\mu}_{q,\vec{i}} = -\frac{1}{2} + \frac{\vec{i}}{N} - \frac{1}{2\,N} \quad \text{and} \quad \vec{\mu}_{\vec{i}\pm\frac{1}{2}} = \vec{\mu}_{\vec{i}} \pm \frac{1}{2\,N}. \tag{4.15}$$

Here, we use the multi-index notation

$$\vec{i} = (i,j,k)^T, \quad i,j,k = 1,\dots,N. \tag{4.16}$$

For the sake of brevity, we do not declare the range of indices i,j,k in each equation from here on. When we translate (4.1) into our new notation for block-structured FV we get

$$\overline{\boldsymbol{u}}_{q,\vec{i}}(t) = \frac{N^3}{\Delta x_q\,\Delta y_q\,\Delta z_q} \int_{\vec{\mu}_{\vec{i}-\frac{1}{2}}}^{\vec{\mu}_{\vec{i}+\frac{1}{2}}} \boldsymbol{u}(\vec{x},t)\,\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z, \tag{4.17}$$

which denotes the cell-averaged fluid states of cell $\mathcal{V}_{q,\vec{i}}$ within block Ω_q at time t.

Semi-discrete Form

We extend the general hyperbolic conversation law (2.6) in differential form with a source term $\boldsymbol{\Upsilon}(\vec{x},t)$ as

$$\partial_t\,\boldsymbol{u}(\vec{x},t) = -\sum_d^3 \partial_{x_d}\,\boldsymbol{f}_d(\boldsymbol{u}(\vec{x},t)) - \boldsymbol{\Upsilon}(\vec{x},t).$$

and apply the usual FV machinery in order get to a semi-discrete form, which can be integrated in time. First, we apply the mean value operation (4.1) and use the divergence theorem in order to express the fluid dynamics as a balance between influx and outflux through the closed surface $\partial\mathcal{V}_{q,\vec{i}}$ of cell $\mathcal{V}_{q,\vec{i}}$, i.e.

$$\begin{aligned}
|\mathcal{V}_{q,\vec{i}}|\,\dot{\overline{\boldsymbol{u}}}_{q,\vec{i}}(t) &= -\int_{\mathcal{V}_{q,\vec{i}}} \sum_d^3 \partial_{x_d}\,\boldsymbol{f}_d(\boldsymbol{u}(\vec{x},t))\,\mathrm{d}\vec{x} - \int_{\mathcal{V}_{q,\vec{i}}} \boldsymbol{\Upsilon}(\vec{x},t)\,\mathrm{d}\vec{x} \\
&= -\oint_{\partial\mathcal{V}_{q,\vec{i}}} \sum_d^3 \boldsymbol{f}_d(\boldsymbol{u}(\vec{x},t))\,(\vec{n}_{q,\vec{i}})_d\,\mathrm{d}(\partial\mathcal{V}_{q,\vec{i}}) - \int_{\mathcal{V}_{q,\vec{i}}} \boldsymbol{\Upsilon}(\vec{x},t)\,\mathrm{d}\vec{x}
\end{aligned}$$

with $\vec{n}_{q,\vec{i}} = (n_1,n_2,n_3)^T$ being the outward facing unit normal of the cell's surface. Following standard procedure (Toro 1999), we discretize above surface integral and get for a

cell $\mathcal{V}_{q,\vec{i}}$ in block Ω_q

$$\begin{aligned}\dot{\overline{\boldsymbol{u}}}_{q,\vec{i}} = &- \frac{N}{\Delta x_q}\Big((\overline{\boldsymbol{f}}_1^*)_{q,i+\frac{1}{2}jk} - (\overline{\boldsymbol{f}}_1^*)_{q,i-\frac{1}{2}jk}\Big) \\ &- \frac{N}{\Delta y_q}\Big((\overline{\boldsymbol{f}}_2^*)_{q,ij+\frac{1}{2}k} - (\overline{\boldsymbol{f}}_2^*)_{q,ij-\frac{1}{2}k}\Big) \\ &- \frac{N}{\Delta z_q}\Big((\overline{\boldsymbol{f}}_3^*)_{q,ijk+\frac{1}{2}} - (\overline{\boldsymbol{f}}_3^*)_{q,ijk-\frac{1}{2}}\Big) + \dot{\overline{\boldsymbol{\Upsilon}}}_{q,\vec{i}}\end{aligned} \tag{4.18}$$

where

$$\begin{aligned}(\overline{\boldsymbol{f}}_1^*)_{q,i+\frac{1}{2}jk} &= \frac{N^2}{\Delta y_q\,\Delta z_q}\,\boldsymbol{f}_1^*\Big(\overline{\boldsymbol{u}}^+_{q,i+1jk}, \overline{\boldsymbol{u}}^-_{q,ijk}\Big), \\ (\overline{\boldsymbol{f}}_2^*)_{q,ij+\frac{1}{2}k} &= \frac{N^2}{\Delta z_q\,\Delta x_q}\,\boldsymbol{f}_2^*\Big(\overline{\boldsymbol{u}}^+_{q,ij+1k}, \overline{\boldsymbol{u}}^-_{q,ijk}\Big) \quad \text{and} \\ (\overline{\boldsymbol{f}}_3^*)_{q,ijk+\frac{1}{2}} &= \frac{N^2}{\Delta x_q\,\Delta y_q}\,\boldsymbol{f}_3^*\Big(\overline{\boldsymbol{u}}^+_{q,ijk+1}, \overline{\boldsymbol{u}}^-_{q,ijk}\Big)\end{aligned} \tag{4.19}$$

denote the face-averaged (midpoint rule), consistent numerical two point fluxes solving the Riemann problem at the discontinuous interfaces between adjacent cells in x-, y- and z-direction, respectively. Note, that the FV scheme (4.18) was derived without any reconstruction step at the boundaries. Nevertheless, if we use the affordable, entropy conservative Riemann solver (3.41) we already have a second order scheme suitable for smooth flows (Chandrashekar 2013).

Non-conservative Soure Terms

Since we are solving the ideal GLM-MHD equations (3.29), we also have to consider the non-conservative source terms. Here, we briefly introduce the discretization of the source terms according to the FV machinery we applied above. We have to discetrize the Powell term $\overline{\boldsymbol{\Upsilon}}^{\text{Powell}}_{q,\vec{i}}$ and the GLM term $\overline{\boldsymbol{\Upsilon}}^{\text{GLM}}_{q,\vec{i}}$. The gradients in the Powell term (3.23) are approximated with central finite differencing over cell $\mathcal{V}_{q,\vec{i}}$. The Powell term reads

$$\dot{\overline{\boldsymbol{\Upsilon}}}^{\text{Powell}}_{q,\vec{i}} = \sum_d^3 \frac{N}{(\Delta x_d)_q}\Big(\,\{\!\{B_d\}\!\}_{q,\vec{i}+\vec{o}_d} - \{\!\{B_d\}\!\}_{q,\vec{i}-\vec{o}_d}\Big)\,\boldsymbol{\Phi}^{\text{Powell}}\Big(\overline{\boldsymbol{u}}_{q,\vec{i}}\Big), \tag{4.20}$$

where the source vector is calculated by inserting the mean values into expression (3.24). We introduce $\vec{o}_d = \frac{1}{2}(\delta_{d,1}, \delta_{d,2}, \delta_{d,3})^T$ to be the offset index vector, which allows for a succinct notation. The discretization of the GLM term (3.25) is done analogously and

reads

$$\dot{\overline{\boldsymbol{\Upsilon}}}^{\mathrm{GLM}}_{q,\vec{i}} = \sum_{d}^{3} \overline{v}_d \, \frac{N}{(\Delta x_d)_q} \Big(\{\!\{\Psi\}\!\}_{q,\vec{i}+\vec{o}_d} - \{\!\{\Psi\}\!\}_{q,\vec{i}-\vec{o}_d} \Big) \, \boldsymbol{\Phi}^{\mathrm{GLM}}\Big(\overline{\boldsymbol{u}}_{q,\vec{i}}\Big) + c_{\mathrm{p}} \, \boldsymbol{\Phi}^{\mathrm{damp}}\Big(\overline{\boldsymbol{u}}_{q,\vec{i}}\Big). \quad (4.21)$$

4.5.2 CFL Condition

If we recall that in the evolve step of the FV method a time-independent flux is calculated for every interface, it is clear that the timestep size of the scheme has to be restricted in order to achieve stability. Every flux is computed by solving the Riemann problem across the corresponding interface, and the timestep Δt has to be chosen small enough, such that the solutions of the Riemann problems at opposing interfaces do not interact. The Riemann problem gives rise to different waves and the appropriate timestep can be expressed in terms of the CFL condition. The condition ensures that the distance traveled by the fastest wave of the Riemann solution in time Δt is less than width of the cell.

To calculate a stable timestep the maximum eigenvalue estimate (3.35) for ideal GLM-MHD is evaluated on all mean values $\overline{\boldsymbol{u}}_{q,\vec{i}}$ in each block Ω_q. In three dimensions it reads as

$$\overline{\lambda}^{\max}_{q} = \max_{ijk=1}^{N} \lambda^{\max}\Big(\overline{\boldsymbol{u}}_{q,ijk}\Big). \quad (4.22)$$

Then the maximal timestep is estimated by the CFL condition (4.13) as

$$\Delta t := \frac{CFL}{d} \min_{q} \frac{\min(\Delta x_q, \Delta y_q, \Delta z_q)}{N \, \overline{\lambda}^{\max}_{q}}, \quad (4.23)$$

where $d = 3, CFL \in (0, 1)$ and N is the number of mean values in each direction of the block q. Furthermore, we calculate the global hyperbolic correction speed (3.28) as

$$\overline{c_{\mathrm{H}}} = \max_{q} \max_{d=1}^{3} \max_{ijk=1}^{N} c_d\Big(\overline{\boldsymbol{u}}_{q,ijk}\Big). \quad (4.24)$$

If the CFL number is set to $CFL = 0.5$, the maximum distance the fastest wave can travel is halfway through the cell, and an interaction between two approaching waves is effectively prevented. However, one can also choose a slightly larger value, since the fluxes in the FV method are independent of time over one timestep, and the fastest wave can possibly reach and influence the opposing interface only for $CFL = 1$. If not otherwise specified, we set $CFL = 0.8$ for all simulations presented here.

4.5.3 Experimental Order of Convergence

To study the convergence properties of the FV implementation, we compute the numerical errors, that is the norm of the difference of the numerical solution u_{num} from the reference solution u_{ref} with mean values. In this work, we look at the ∞-norm, i.e.

$$\left|\left|\overline{u}_{\text{num}} - \overline{u}_{\text{ref}}\right|\right|_{\infty} = \max_{q=1}^{Q} \max_{ijk=1}^{N} \left|\overline{u}_{\text{num},q,ijk} - \overline{u}_{\text{ref},q,ijk}\right| \tag{4.25}$$

and the 2-norm, i.e.

$$\left|\left|\overline{u}_{\text{num}} - \overline{u}_{\text{ref}}\right|\right|_{2} = \left[|\Omega|^{-1} \sum_{q=1}^{Q} \sum_{ijk=1}^{N} \frac{V_q}{N^3} \left|\overline{u}_{\text{num},q,ijk} - \overline{u}_{\text{ref},q,ijk}\right|^2 \right]^{\frac{1}{2}}, \tag{4.26}$$

where $V_q = \Delta x_q \, \Delta y_q \, \Delta z_q$ is the volume of block q and $|\Omega|$ is the volume of the computational domain.

For the exact solution, we use the MHD vortex problem derived in Balsara (2004). It is suitable for accuracy testing, because it consists of a smoothly varying and dynamically stable flow configuration that carries out nontrivial motion in the computational domain. The problem is set up on a two-dimensional domain given by $\Omega = [-10, 10]^2$. with periodic boundaries. An unperturbed MHD flow with $\rho, v_1, v_2, p = 1$ is initialized on the computational domain while the other state variables are set to zero. The ratio of specific heats is given by $\gamma = 5/3$. The vortex is initialized at the center of the computational domain by way of fluctuations in the velocity and magnetic fields given by

$$\begin{aligned} (\delta v_1, \delta v_2) &= \frac{5}{2\,\pi}\, e^{(1-r^2)/2}\, (-y, x) \\ (\delta B_1, \delta B_2) &= \frac{1}{2\,\pi}\, e^{(1-r^2)/2}\, (-y, x) \\ \delta p &= \left(\left(\frac{5}{2\,\pi}\right)^2 (1 - r^2) - \frac{1}{2} \left(\frac{1}{2\,\pi}\right)^2 \right) e^{1-r^2}\, (-y, x) \end{aligned} \tag{4.27}$$

with radius $r^2 = x^2 + y^2$.

Prior to the subsequent discussions, we first verify the experimental order of convergence (EOC) of our implementation of the 2D version (4.18) and the Riemann solver (3.41) (EC FV). For time integration we use the LS-RK(5,4) scheme. We look at the total pressure (3.20) and calculate the ∞-norm and 2-norm according to (4.25) and (4.26). Since the

total pressure involves all state variables it is a good quantity to measure the overall convergence of the code. The results in Table 4.1 confirm the expected EOC of 2.

Table 4.1: EOC of total pressure P of the MHD vortex problem (4.27) run by the second order EC FV scheme.

DOF	$\|\|P\|\|_\infty$	$\|\|P\|\|_2$	EOC_∞	EOC_2
16^2	1.379e-00	3.241e-00	n/a	n/a
32^2	1.356e-00	1.658e-00	0.02	0.97
64^2	4.023e-01	3.599e-01	1.75	2.20
128^2	9.434e-02	8.888e-02	2.09	2.02
256^2	2.364e-02	2.183e-02	2.00	2.03
512^2	5.962e-03	5.453e-03	1.99	2.00
1024^2	1.488e-03	1.363e-03	2.00	2.00

4.5.4 Entropy Conservation

The EC Riemann solver for MHD (3.41) promises to conserve entropy up to machine precision. For our stress test of the entropy conservation property we devised our own 2D MHD Kelvin-Helmholtz (KHI) instability simulation. This setup has proven to be useful for our purposes since it starts with a smooth well resolved initial state and gradually develops vortical, turbulent-like structure with increasingly smaller scales. Although, the setup is subsonic, it locally develops transonic regions, which are very challenging for most schemes in our investigation.

The weakly-magnetized 2D MHD KHI setup is initialized in a squared and periodic domain $\Omega = [-1, 1]^2$ and reads

$$\begin{aligned}
\rho_0(\vec{x}) &= \frac{1}{2} + \frac{3}{4}\Big(\tanh(15\,(y+\frac{1}{2})) - \tanh(15\,(y-\frac{1}{2}))\Big) \\
(v_1)_0(\vec{x}) &= \frac{1}{2}\Big(\tanh(15\,(y+\frac{1}{2})) - \tanh(15\,(y-\frac{1}{2})) + 1\Big) \\
(v_2)_0(\vec{x}) &= \frac{1}{10}\,\sin(2\,\pi\,x) \\
(v_3)_0 = 0,\ p_0 = 1,\ \vec{B}_0 &= (10^{-2}, 0, 0)^T, \Psi_0 = 0.
\end{aligned} \tag{4.28}$$

The initial conditions of the setup are also plotted in Figure 4.1.

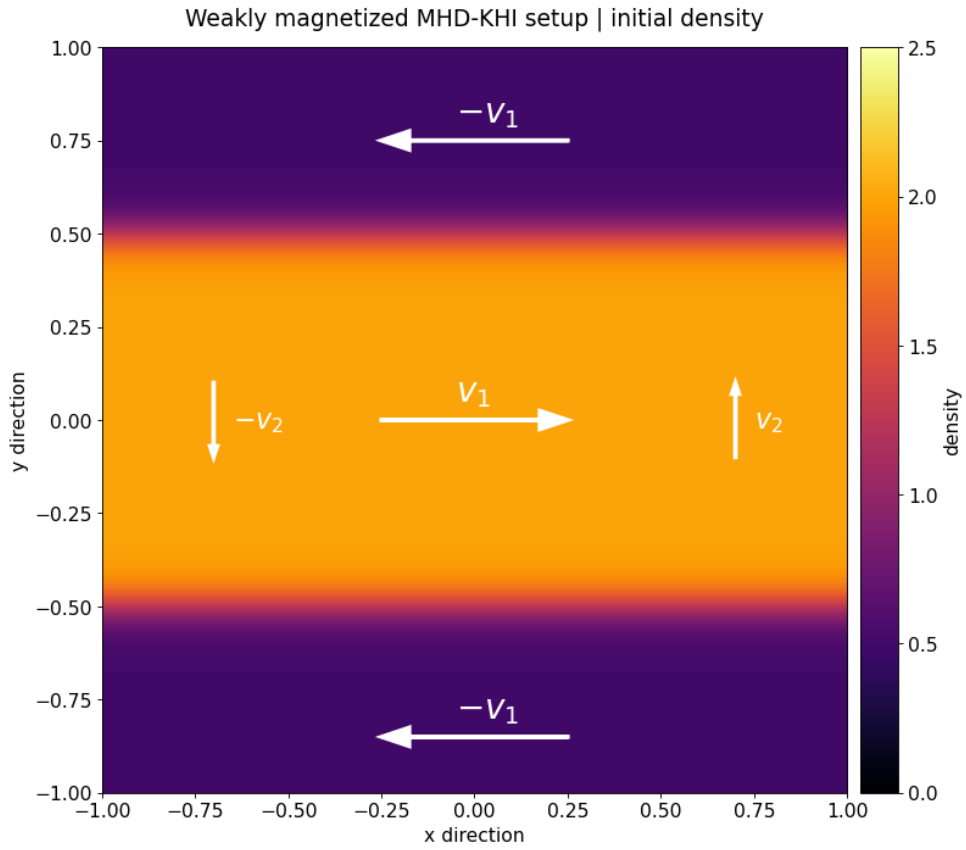

Figure 4.1: Initial density of the 2D MHD KHI simulation with a weak constant magnetic field $B_1 = 10^{-2}$ in x-direction.

We compute the setup (4.28) with the second order EC FV scheme and with two different timestepping schemes, the third order SSP-RK(4,3) and the fourth order LS-RK(5,4) method. The damping terms are inactivated ($c_\mathrm{p} = 0$). We let the simulations run on a uniform grid of 512^2 cells until the point $t \approx 3.7$, when the code crashes due to negative densities and pressures. This happens with both time integration methods and at the same point in the simulation. The density contours close before the crash are shown in Figure 4.2.

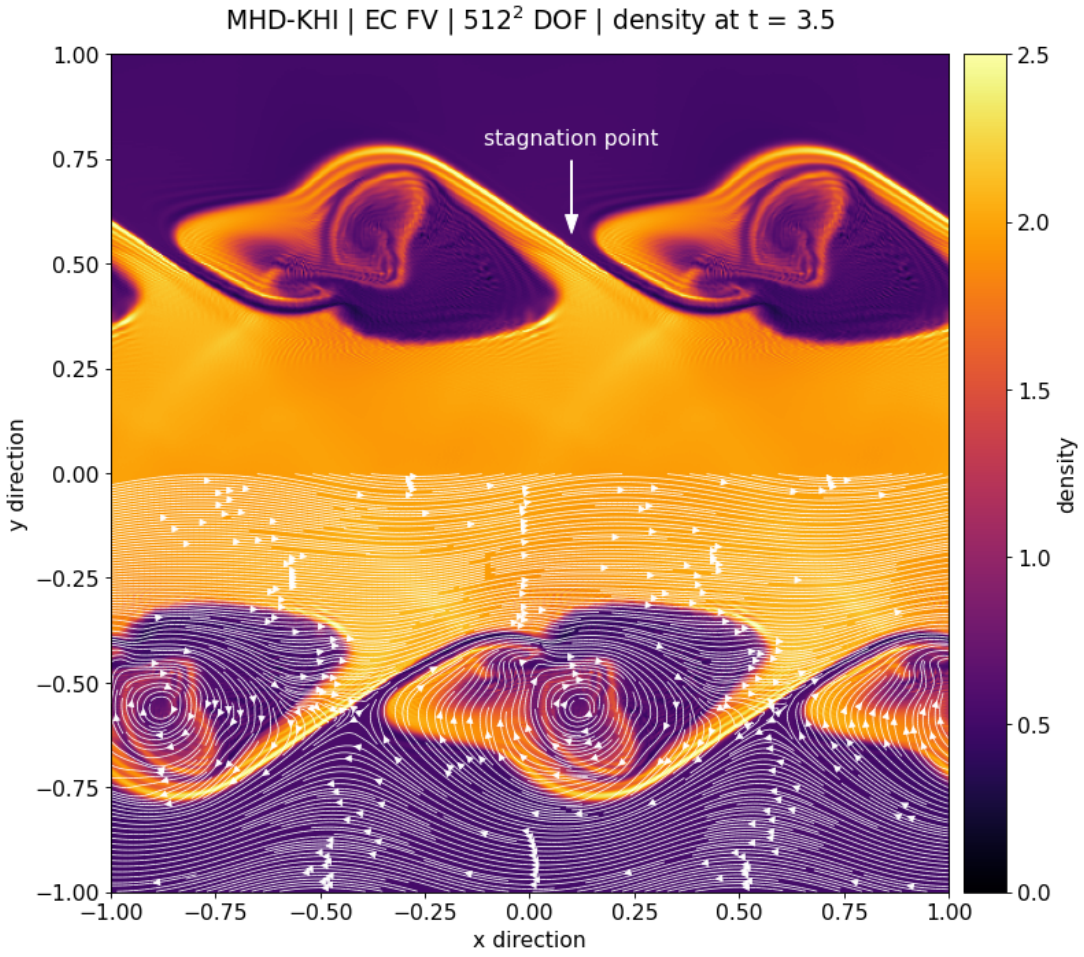

Figure 4.2: Contour plot of the density of the weakly magnetized 2D MHD KHI setup run by EC FV and SSP-RK(4,3) close before the crash. The result with LS-RK(5,4) looks the same. In the lower half, the streamlines (white) of the velocity field are overlayed.

The setup starts with perfectly smooth initial conditions, but quickly develops pronounced density gradients, while small scale structures inside the main vortices start to emerge. Moreover, lots of acoustics (spurious density oscillations) are observable. The formerly constant magnet field has curled up, which increases the magnetic pressure alongside the density slopes visible in Figure 4.3. Shown there are the contours of the inverse plasma-beta β^{-1} with overlayed magnetic field lines in white. The distinct red areas are regions of increased magnetic pressure counteracting the still dominating hydrodynamical pressure.

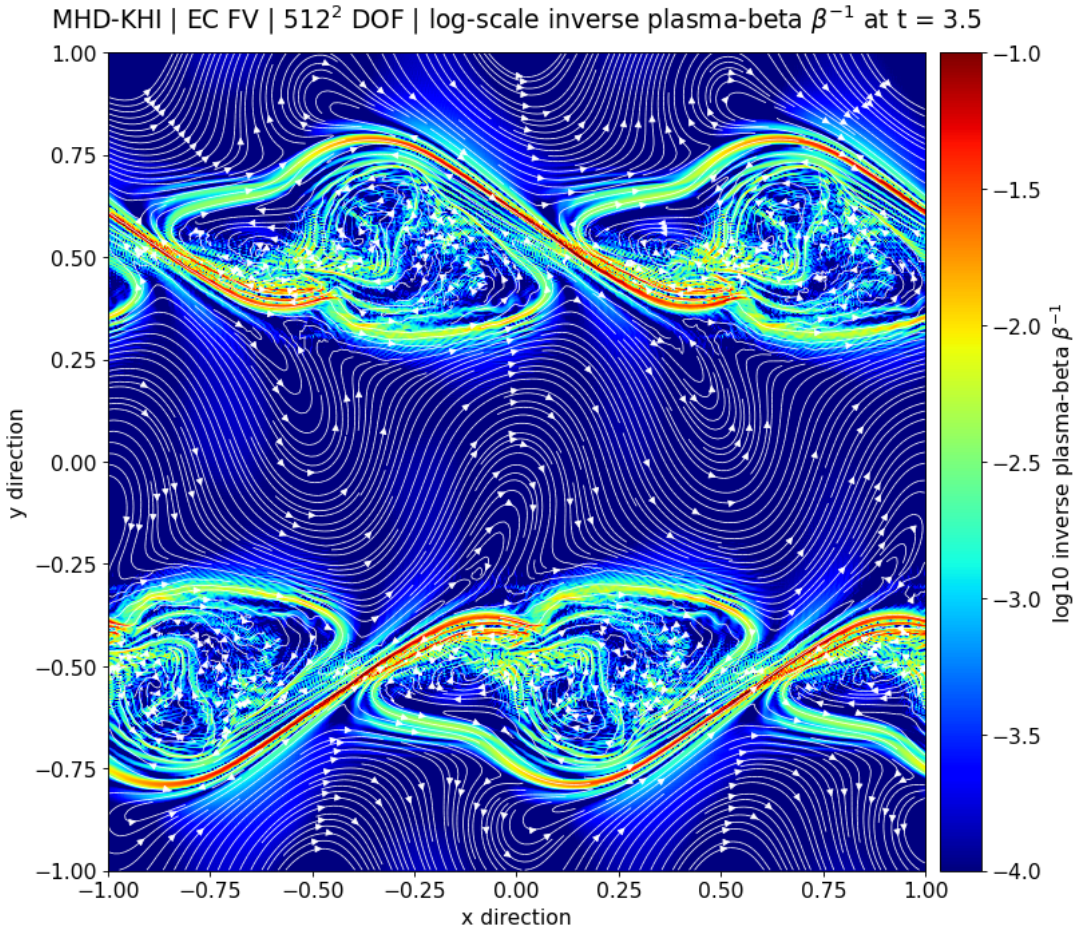

Figure 4.3: Contour plots the inverse plasma-beta β^{-1} of the weakly magnetized MHD KHI setup run by the entropy conservative FV and SSP-RK(4,3) close before the crash. The result with LS-RK(5,4) looks the same. The streamlines (white) of the magnetic field are overlayed.

The setup consists of four stagnation points, for example at $(x, y) \approx (0.125, 0.5)$, which are clearly recognizable by the tightly squeezed and lengthened magnetic field lines and the buildup of magnetic pressure. Stagnation points pose a challenge for numerical schemes, since they are considered to be a strong source of numerical instabilities. Numerical schemes have to inject the right amount of dissipation in order to stabilize the simulation and "survive" this stressing period long enough till the critical point has been dissolved. Another source of instability are highly compressive, transonic regions. In the upper half in Figure 4.4 we plot the sonic Mach number (3.11) and in the lower half we plot the turbulent Mach number (3.12). Clearly, at $(x, y) \approx (0.6, 0.4)$ there is a transonic region (dark red dot) of high compression indicated by the mirrored turbulent Mach number plot. Since the EC FV scheme has no mechanism to induce enough numerical dissipation,

it cannot handle these situations and eventually crashes.

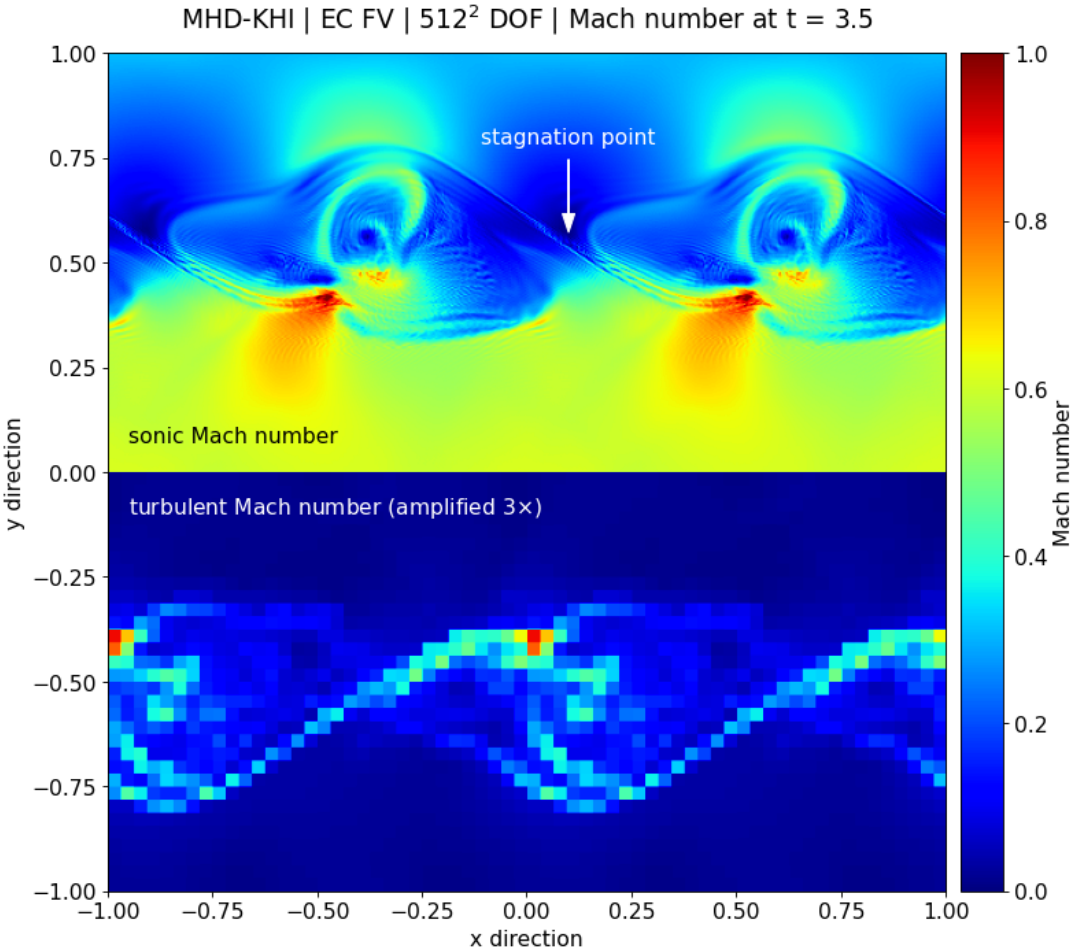

Figure 4.4: Contour plots the Mach number of the weakly magnetized MHD-KHI setup run by the EC FV and SSP-RK(4,3) close before the crash. The upper half depicts the sonic Mach number $\mathcal{M}$ according to (3.11) and the lower half depicts the turbulent Mach number $\mathcal{T}_q$ per block Ω_q according (3.12). The turbulent Mach number $\mathcal{T}_q$ is amplified by a factor of 3 in order to match the color range of the upper half. The bright red blocks in the lower half correspond to $\mathcal{T}_q \approx 0.3$ and the blocks in shades of cyan correspond to $\mathcal{T}_q \approx 0.15$.

In Figure 4.5 we show how the total entropy (integrated over the whole computational domain Ω) of the EC FV develops over time. Instead of the total entropy we plot the relative entropy drift over time, which we define as

$$\mathcal{S}_{\text{drift}}(t) = \frac{\int_\Omega \mathcal{S}(\vec{x}, t)\,\mathrm{d}\vec{x} - \int_\Omega \mathcal{S}(\vec{x}, 0)\,\mathrm{d}\vec{x}}{\int_\Omega \mathcal{S}(\vec{x}, 0)\,\mathrm{d}\vec{x}}. \tag{4.29}$$

It seems the (numerically) exact conservation of the total entropy is not confirmed in

Figure 4.5. Both timestepping schemes considerably drift away from zero at some point. The entropy error even seems to explode till the code crashes at $t \approx 3.7$ for both RK methods. In order to rule out bugs in the RK implementations we also plotted the relative mass drift in the same plot (scale is on the right y-axis), computed analogously to (4.29). The fourth order RK scheme conserves the mass perfectly, while there is still a mass drift for the third order RK scheme.

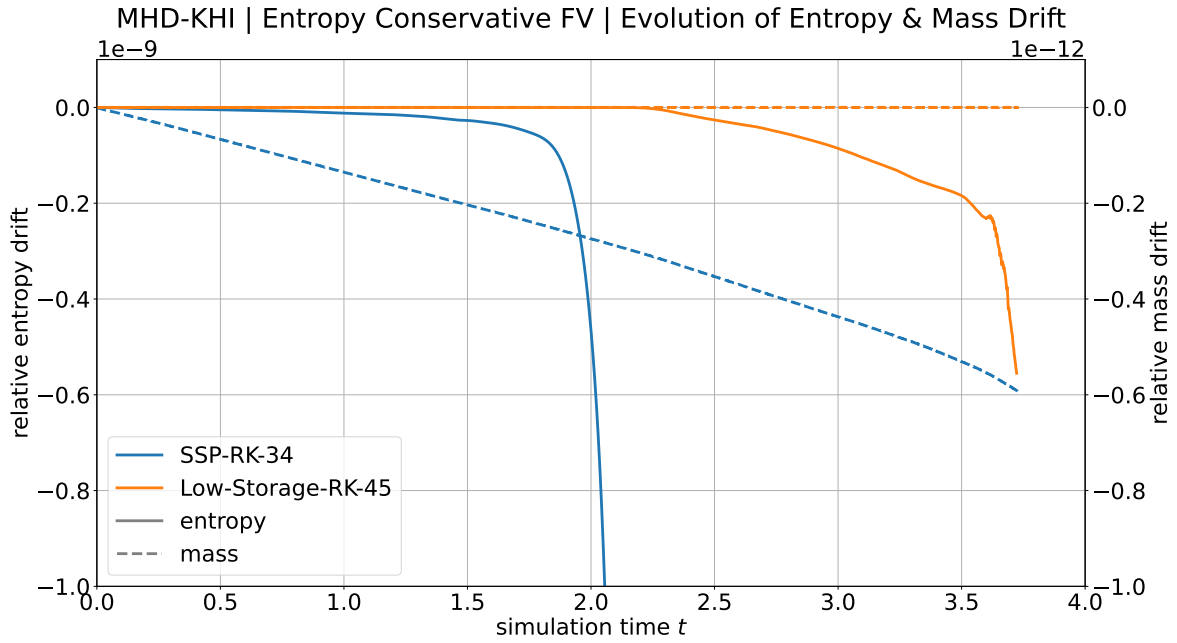

Figure 4.5: Evolution of the global entropy drift (4.29) and mass drift of the weakly magnetized MHD-KHI setup run by the EC FV method with the third order SSP RK scheme and the fourth order Low-Storage RK scheme. Both simulations crash at $t \approx 3.7$ visible by a rapid surge of entropy drift caused by a breakdown of the numerics. Apparently, the SSP RK scheme has a noticeably linear mass drift losing mass at a rate of $\Delta\text{mass}/\Delta t \approx -1.3 \times 10^{-10}$.

We want to note, that we specifically chose this setup since it pushes the numerical schemes to their limits; even break the numerics at some point in order to investigate their stability properties. We interpret volatile entropy production as a sign of instability, followed by a loss of convergence to any sensible solution. Since, the derivations about entropy conservation has been done on semi-discrete level only, any time integration related errors are not taking into account. Entropy conservative schemes for the full discretization, both in space and time, e.g. Friedrich et al. (2019), are implicit in time and not suitable for our simulations. Nevertheless, the entropy error introduced through explicit time integration should be negligible compared to the entropy production by any numerical or physical

dissipation mechanism present in our simulations.

4.5.5 Cell Entropy Production Rate

In Figure 4.5 we see that timestepping methods introduce errors in the total entropy, which accumulate over time. However, we are specifically interested in reliably measuring the entropy contribution by the spatial discretization, preferably even on a local level.

We discretize the conservation law of the entropy (2.14) by the FV machinery and get

$$\dot{\mathcal{S}}_{q,\vec{i}} = -\sum_{d}^{3} \frac{N}{(\Delta x_d)_q} \left((\overline{\boldsymbol{\mathcal{F}}}^*_d)_{q,\vec{i}+\vec{o}_d} - (\overline{\boldsymbol{\mathcal{F}}}^*_d)_{q,\vec{i}-\vec{o}_d} \right). \tag{4.30}$$

The surface entropy flux $\mathcal{F}_d$ is constructed from expression (2.19) and reads

$$(\boldsymbol{\mathcal{F}}^*_d)_{q,\vec{i}+\vec{o}_d} = \left(\{\!\{\boldsymbol{w}\}\!\} \cdot \boldsymbol{f}^*_d - \{\!\{\theta_d\}\!\} + \left\{\!\!\left\{ \boldsymbol{w} \cdot \boldsymbol{\Upsilon}^{\mathrm{MHD}}_d \right\}\!\!\right\}^{\dagger} \right)_{q,\vec{i}+\vec{o}_d} \tag{4.31}$$

Note that, we already incorporated the peculiarity of the non-conservative source terms $\boldsymbol{\Upsilon}^{\mathrm{MHD}}_d$ introduced by the ideal GLM-MHD equations (3.29) in the cell entropy surface flux (4.31). The entropy flux of the non-conservative source terms across the cell interface is evaluated as follows

$$\left\{\!\!\left\{ \boldsymbol{w} \cdot \Upsilon^{\mathrm{MHD}}_d \right\}\!\!\right\}^{\dagger} = \frac{1}{2} \Big(\boldsymbol{w}^+ \cdot \boldsymbol{\Upsilon}^{*,\mathrm{MHD}}_d(\boldsymbol{u}^+, \boldsymbol{u}^-) + \boldsymbol{w}^- \cdot \boldsymbol{\Upsilon}^{*,\mathrm{MHD}}_d(\boldsymbol{u}^-, \boldsymbol{u}^+) \Big) \tag{4.32}$$

with the two-point MHD source term

$$\boldsymbol{\Upsilon}^{*,\mathrm{MHD}}_d(\boldsymbol{u}^+, \boldsymbol{u}^-) = \boldsymbol{\Upsilon}^{*,\mathrm{Powell}}_d(\boldsymbol{u}^+, \boldsymbol{u}^-) + \boldsymbol{\Upsilon}^{*,\mathrm{GLM}}_d(\boldsymbol{u}^+, \boldsymbol{u}^-). \tag{4.33}$$

The two-point Powell (3.23) and GLM (3.25) terms read

$$\boldsymbol{\Upsilon}^{*,\mathrm{Powell}}_d(\boldsymbol{u}^+, \boldsymbol{u}^-) = \frac{1}{2} \left(B^+_d + B^-_d \right) \boldsymbol{\Phi}^{\mathrm{Powell}}\Big(\boldsymbol{u}^+\Big) \tag{4.34}$$

and

$$\boldsymbol{\Upsilon}^{*,\mathrm{GLM}}_d(\boldsymbol{u}^+, \boldsymbol{u}^-) = \frac{v^+_d}{2} \left(\Psi^+ + \Psi^- \right) \boldsymbol{\Phi}^{\mathrm{GLM}}\Big(\boldsymbol{u}^+\Big). \tag{4.35}$$

Next, we take the dot product of the cell's entropy vector $\overline{\boldsymbol{w}}_{q,\vec{i}}$ with the RHS (4.18) and subtract the scheme (4.30), and finally, get a measure for the entropy production rate

$\overline{\Delta \dot{\mathcal{S}}}_{q,\vec{i}}$ in cell $\mathcal{V}_{q,\vec{i}}$ as

$$\overline{\Delta \dot{\mathcal{S}}}_{q,\vec{i}} = \overline{\boldsymbol{w}}_{q,\vec{i}} \cdot \dot{\overline{\boldsymbol{u}}}_{q,\vec{i}} - \dot{\overline{\mathcal{S}}}_{q,\vec{i}}, \tag{4.36}$$

which is analog to expression (2.22). The entropy production rate must always be zero (or negative) in order to fulfill the entropy condition (2.15). Detailed proofs are presented in Derigs et al. (2018) and Rueda-Ramírez et al. (2022a).

For our EC FV scheme this is indeed the case as we show numerically with a snapshot of local cell entropy production rates in Figure 4.6 at $t = 2.5$ for the weakly magnetized KHI setup (4.28). The local entropy production is zero except for truncation errors due to double precision (64 bit) floating point arithmetic, i.e. of the order 10^{-13}, in each cell in the whole computational domain. In Figure 4.7 we further corroborate that the EC FV scheme conserves the entropy even with the timestepping method SSP-RK(4,3) by plotting the minimum and maximum cell entropy production rates measured in the whole domain at every timestep till the point of crash after 3180 timesteps respectively at simulation time $t = 3.727675$.

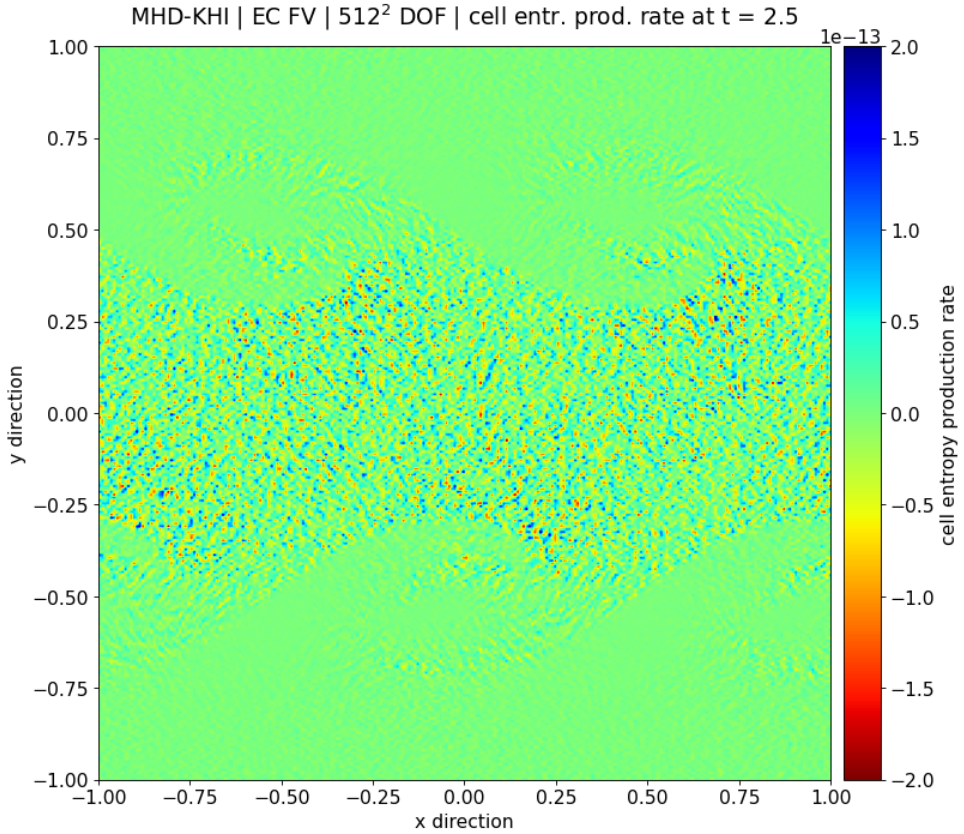

Figure 4.6: Cell entropy production rates (4.36) of the weakly magnetized KHI setup (4.28) at $t = 2.5$ computed with the EC FV scheme and timestepping method SSP-RK(4,3).

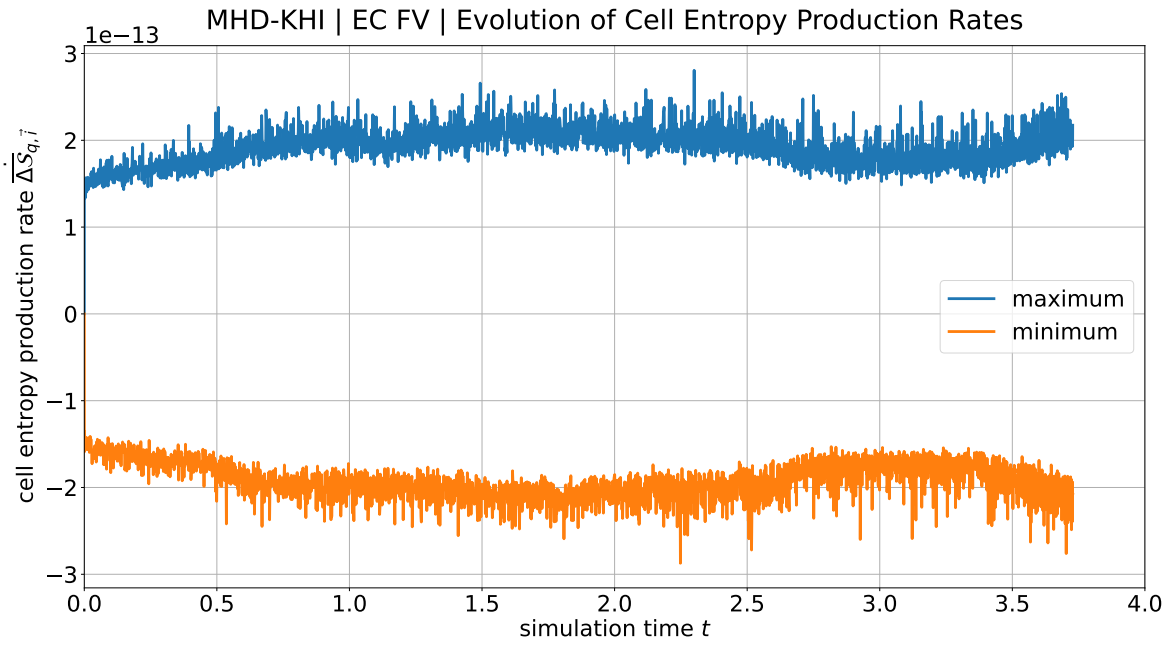

Figure 4.7: Evolution of the maximum and minimum cell entropy production rate (4.36) in the whole computational domain computed with the EC FV scheme and timestepping method SSP-RK(4,3). The simulation crashes (negative densities and pressures) after 3180 timesteps respectively at simulation time $t = 3.727675$.

4.5.6 Satisfying the Cell Entropy Inequality

An important and desirable property of robust numerical schemes for hyperbolic problems is that they should yield monotone solutions at discontinuities. Central schemes like our EC FV scheme give highly oscillatory solutions near shocks and other troublesome flow features, which are physically wrong and eventually lead to negative densities and pressures as we have seen in the previous two sections.

Generally, central schemes with appropriately added dissipation are able to compute non-oscillatory solutions. For scalar conservation laws, the TVD condition (4.6) allows the well-founded design of accurate, non-oscillatory schemes. A widely used and time proven TVD limiter is the minmod limiter, for which the gradient σ_i of the linear interface reconstruction

$$\boldsymbol{u}^{\pm}_{i\pm\frac{1}{2}} = \overline{\boldsymbol{u}}_i \pm \frac{\sigma_i}{2}, \tag{4.37}$$

is calculated as

$$\sigma_i = \operatorname{minmod}(\overline{\boldsymbol{u}}_i - \overline{\boldsymbol{u}}_{i-1}, \overline{\boldsymbol{u}}_{i+1} - \overline{\boldsymbol{u}}_i), \tag{4.38}$$

where

$$\text{minmod}(a,b) = \begin{cases} s\ \min(|a|,|b|) & \text{if } \text{sign}(a) = \text{sign}(b) \\ 0 & \text{otherwise.} \end{cases} \tag{4.39}$$

Both one-sided slopes are computed and compared, and the slope with smaller absolute value wins. However, if the two slopes point in different directions, the gradient is set to zero. In cells where the slope limiter is active, the scheme reduces to first order. Hence, the limiter should only be applied to problematic locations, such that the solution in smooth regions remains unaffected and converges with the maximum order. This can be achieved by, for example, relaxing the TVD condition and devising limiters who are total variation bounded (Shu 1987) at the price of surrendering some robustness, especially at strong shocks. Isolating discontinuities, and confining the limiter to these locations, proves to be difficult. The choice of the slope limiter is therefore in many cases a tradeoff between avoiding spurious oscillations and avoiding the clipping of smooth extrema. Since our final goal is to combine the FV scheme with DG anyway, the robustness of FV is our primary target. Hence, we deem the minmod limiter as our best option. For more details about the minmod limiter and other suitable reconstruction approaches, we refer to LeVeque (2002) or Godlewski and Raviart (2013).

Unfortunately, TVD does not simply carry over to nonlinear, hyperbolic systems like the compressible Euler equations or ideal MHD. There is still a lack of theoretical basis for the design of non-oscillatory schemes for systems of conservation laws, especially in two and three dimensions. The assumption is that the entropy condition (2.15) can be a useful criterion for the design of non-oscillatory schemes (Ismail and Roe 2009). However, this condition only specifies that there must be non-zero dissipation, which leads to some entropy production, but does not say how much entropy generation is necessary to yield monotone solutions. Nonetheless, we devise a Rusanov-type interface flux

$$\boldsymbol{f}^{*,\text{minmod}}_{i\pm\frac{1}{2}} = \boldsymbol{f}^{\#}(\overline{\boldsymbol{u}}_i, \overline{\boldsymbol{u}}_{i\pm1}) - \frac{\lambda^{\max}_{i\pm\frac{1}{2}}}{2}\left(\boldsymbol{u}^{-}_{i\pm1/2} - \boldsymbol{u}^{+}_{i\pm1/2}\right) \tag{4.40}$$

and assume that the linear interface reconstruction (4.37) injects enough numerical dissipation at the right places to become TVD for our nonlinear systems. $\boldsymbol{f}^{\#}$ is the symmetric, entropy conservative Riemann solver (3.41) and

$$\lambda^{\max}_{i\pm\frac{1}{2}} = \max\left\{\left|\lambda^{\max}(\overline{\boldsymbol{u}}_i)\right|, \left|\lambda^{\max}(\overline{\boldsymbol{u}}_{i\pm1})\right|\right\}$$

is the maximum characteristic wave speed estimate for, e.g., (3.35) of the ideal GLM-MHD equations at the interface.

Remark. Fjordholm et al. (2012) showed that the minmod limiter is sign preserving if the linear interface reconstruction (4.37) is applied on the entropy variables. The sign property is an important feature in order to perfectly ensure the correct "direction" of entropy dissipation. However, in our numerical tests we observed positivity issues (negative density and pressure) in very strong shock setups. Thus, we prefer to reconstruct on primitive variables instead, which has proven to be much more robust and is also much faster in terms of runtime performance.

We want to make sure that our devised Minmod FV scheme (4.40) with minmod interface dissipation does not overly compromise the targeted convergence order of two. We look at the total pressure (3.20) and calculate the ∞-norm (4.25)and 2-norm (4.26) for the 2D MHD vortex problem (4.27). For time integration we use the LS-RK(5,4) scheme even though a second order RK scheme would do as well. The results in Table 4.2 confirm the expected EOC of two.

Table 4.2: EOC of total pressure P of the MHD vortex problem (4.27) run by the second order Minmod FV scheme (4.40).

DOF	$\|\|P\|\|_\infty$	$\|\|P\|\|_2$	EOC_∞	EOC_2
16^2	1.487e-00	2.103e-00	n/a	n/a
32^2	9.390e-01	1.080e-00	0.66	0.96
64^2	2.078e-01	2.451e-01	2.18	2.14
128^2	4.559e-02	4.871e-02	2.19	2.33
256^2	1.366e-02	1.212e-02	1.74	2.01
512^2	3.618e-03	2.971e-03	1.92	2.03
1024^2	9.738e-04	7.287e-04	1.89	2.03

Figure 4.8 shows the results of a simulation of the weakly magnetized MHD KHI setup (4.28) computed with the Minmod FV till final simulation time $T = 10$. Clearly, the density (left in Figure 4.8) is smeared out compared to the results with the EC FV scheme (cf. Figure 4.2). But now, the simulation runs stably due to the regularizing numerical dissipation injected at steep solution gradients as is clearly visible in the cell entropy production rate snapshot (right plot in Figure 4.8). The entropy production rate (EPR) is zero or negative in the whole domain as is demanded by the entropy condition (2.15). In Figure 4.9 we also plot the minimum and maximum cell EPR at each timestep

and in the whole domain over the course of the simulation. The maximum EPR is kept at zero, while the minimum EPR intriguingly spikes at $t \approx 3.7$ matching nicely with the point of crash of the EC FV scheme. Apparently, a jump in the EPR evolution signifies a troubling time period when the scheme has to inject enough numerical dissipation at the right places in order to "survive" and not crash.

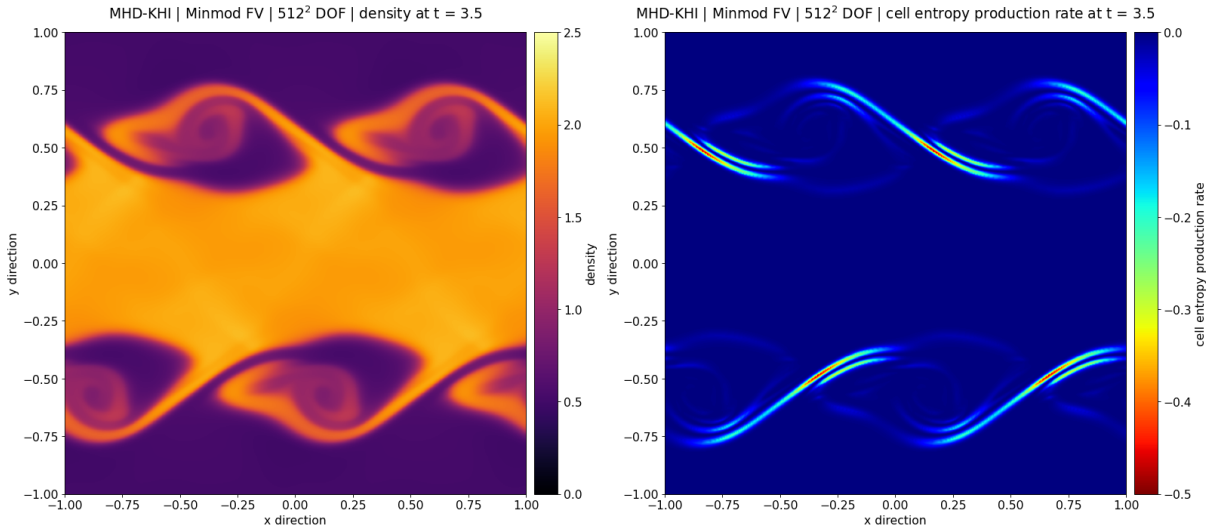

Figure 4.8: Contour plots of the density (left) and of cell entropy production rate (4.36) (right) of the weakly magnetized MHD KHI setup (4.28) run by the Minmod FV.

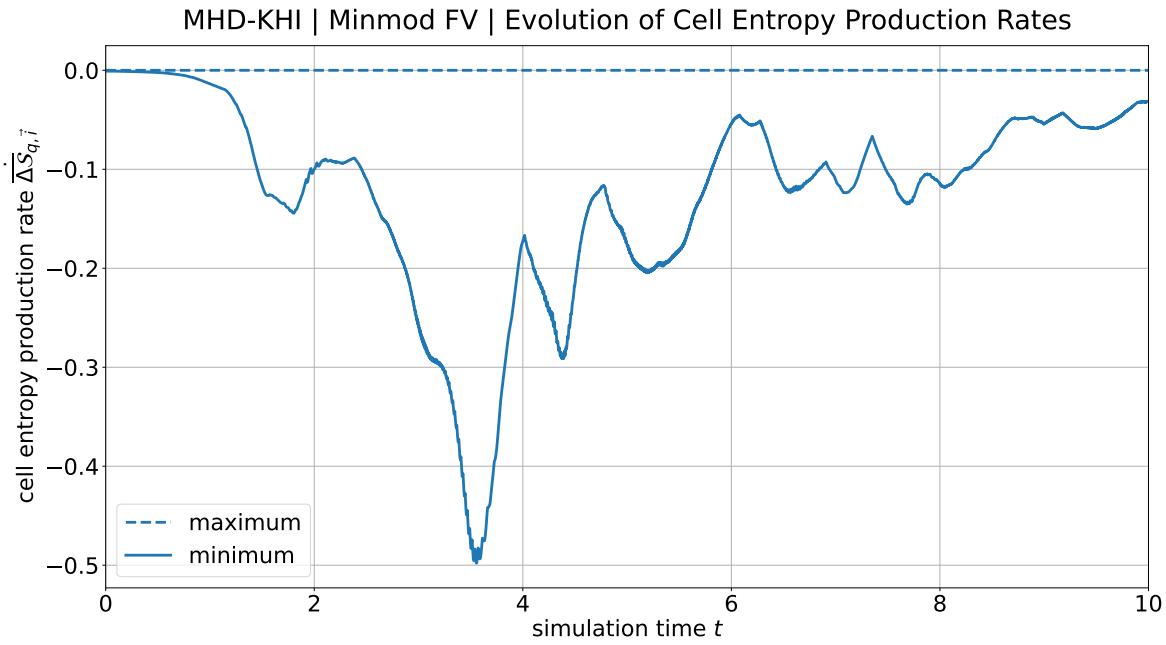

Figure 4.9: Evolution of the measured minimum and maximum entropy production rates at each timestep and in the whole domain of weakly magnetized MHD KHI setup (4.28) run by the Minmod FV till final simulation time $T = 10$.

We stress that we only consider the EPR as a qualitative tool to gain insights into the scheme's numerical dissipation mechanics, because a well-founded judgment on the amount (quantity) of the locally produced entropy is beyond the scope of this work and subject to future investigations.

4.6 Discontinuous Galerkin Spectral Element Method

Discontinuous Galerkin (DG) methods belong to the class of Finite Element (FE) methods (Reddy 2004), which are used for solving a vast range of partial differential equations. In these methods a mesh is generated by subdividing space into individual, non-overlapping elements, and basis functions defined on a reference element are used for representing the solution. FE methods are closely related to spectral methods (Canuto et al. 2012) and built on similar ideas. Spectral methods use basis functions that are nonzero and bounded over the whole domain, while FE methods use basis functions that are nonzero and bounded only within their respective elements. In other words, spectral methods take on a global approach while FE methods use a local approach. Spectral methods are appreciated for their excellent error properties, with the fastest possible ("exponen-

tial") convergence speed for sufficiently smooth solutions. Single domain spectral shock capturing methods in multi-dimensions, however, are still an open research question.

In the FE community, a method where the degree of the elements is very high or increases as the grid parameter $\frac{\Delta x}{n}$ decreases to zero for $n \to \infty$ is occasionally called a spectral element method. The idea of so-called Discontinuous Galerkin Spectral Element Methods (Black 1999) is to choose a finite dimensional space of candidate solutions, namely polynomials of degree $n-1$ pinned at n collocation points in the reference element, and to select the solution such that it satisfies the given equation at the collocation points. The hyperbolic fluid equations are solved in a weak formulation with respect to the basis functions and in contrast to continuous Galerkin methods, the solution in DG is discontinuous across element interfaces. Elements exchange information with their direct neighbors via Riemann solvers analogous to FV methods.

The discontinuous nature of DG gives rise to a couple of advantages. Unlike in FV methods, the high order stencil of DG is local and only information from touching surfaces of neighboring elements is required. This feature is of great value for unstructured meshes in complex geometries and for distributed high performance computing, since less time is spent on data communication and more time can be devoted to calculating the solution. Of course, for our envisaged astrophysics applications modeled in Cartesian geometry, the first advantage is of minor importance. Moreover, for hydrodynamic applications a discontinuous solution representation can potentially capture shocks very well. At locations near discontinuities in the flow troubled elements can be individually limited such that an oscillation-free representation can be achieved.

In the following, we outline the basic building blocks of DG, describe strategies to satisfy the entropy condition (2.15), and investigate the robustness and accuracy of several DG variants by the example of the weakly magnetized MHD-KHI setup (4.28).

4.6.1 Building Blocks

In this section, we briefly outline the construction of the Discontinuous Galerkin Spectral Element Method (DGSEM). A very detailed account can be found for example in Kopriva (2009b). From now on, we use DG and DGSEM synonymously. To derive the n^{th} order DG scheme in element Ω_q and within the Cartesian domain $\Omega \subset \mathbb{R}^3$, we start with the general conservation law (2.6) in variational form (2.9) and apply integration-by-parts in

space to the flux divergence. We get

$$\int_{\Omega_q} \Big(\partial_t \boldsymbol{u}(\vec{x},t)\Big)\,\boldsymbol{\phi}(\vec{x})\,\mathrm{d}\vec{x} = \sum_{d=1}^{3} \int_{\Omega_q} \boldsymbol{f}_d(\boldsymbol{u}(\vec{x},t))\,\partial_{x_d}\boldsymbol{\phi}(\vec{x})\,\mathrm{d}\vec{x} - \sum_{d=1}^{3} \oint_{\partial\Omega_q} \boldsymbol{f}_d(\boldsymbol{u}(\vec{x},t))\,\boldsymbol{\phi}(\vec{x})\,n_d(\vec{x})\,\mathrm{d}(\partial\Omega_q) \tag{4.41}$$

with the outward facing normal vector $n_d(\vec{\chi}) = \Big(n_1(\vec{\chi}), n_2(\vec{\chi}), n_3(\vec{\chi})\Big)^T$.

Reference Element

For practical reasons, we transform above expression into the unit reference space

$$\mathcal{I} = \left[-\frac{1}{2}, \frac{1}{2}\right] \tag{4.42}$$

and each element $\Omega_q \subset \Omega$ with midpoint $\vec{\mu}_q \in \Omega$ and size $\Delta\vec{x}_q \in \mathbb{R}^3_+$ is transformed to the reference space $\mathcal{I}$ by the mapping

$$\vec{x}_q(\vec{\chi}) = \vec{\mu}_q + \vec{\chi}\,\Delta\vec{x}_q\,, \quad \vec{\chi} \in \mathcal{I}^3. \tag{4.43}$$

The variational form (4.41) then reads

$$|\Omega_q| \int_{\mathcal{I}^3} \Big(\partial_t \boldsymbol{u}(\vec{\chi},t)\Big)\,\boldsymbol{\phi}(\vec{\chi})\,\mathrm{d}\vec{\chi} = \sum_{d=1}^{3} \int_{\mathcal{I}^3} \boldsymbol{f}_d(\boldsymbol{u}(\vec{\chi},t))\,\partial_{\chi_d}\boldsymbol{\phi}(\vec{\chi})\,\mathrm{d}\vec{\chi} - \sum_{d=1}^{3} \oint_{\partial\mathcal{I}^3} \boldsymbol{f}_d(\boldsymbol{u}(\vec{\chi},t))\,\boldsymbol{\phi}(\vec{\chi})\,n_d(\vec{\chi})\,\mathrm{d}(\partial\mathcal{I}^3) \tag{4.44}$$

with the volume of the element $|\Omega_q|$ given by

$$|\Omega_q| = \prod_{d=1}^{3} \Delta x_d = \Delta x\,\Delta y\,\Delta z. \tag{4.45}$$

Lagrange Polynomials

We choose the test function $\phi(\vec{\chi})$ to be the product of Lagrange polynomials ℓ

$$\phi(\chi) := \ell_i(\chi_1)\,\ell_j(\chi_2)\,\ell_k(\chi_3), \quad i,j,k = 1,\ldots,n, \tag{4.46}$$

where

$$\ell_i(\chi) = \prod_{k=1,k\neq i}^{n} \frac{\chi - \xi_k}{\xi_i - \xi_k}, \quad i = 1, \ldots, n, \tag{4.47}$$

are constructed with n distinct interpolation nodes $\xi_i \in \mathcal{I}$. Inside each element Ω_q we make a polynomial tensor product ansatz of degree $n-1$ and approximate the exact solution $\boldsymbol{u}_q(\vec{x}, t)$ by

$$\boldsymbol{u}_q(\vec{\chi}, t) \approx \boldsymbol{p}_q(\vec{\chi}, t) = \sum_{i,j,k=1}^{n} \widetilde{\boldsymbol{u}}_{q,ijk}(t)\, \ell_i(\chi_1)\, \ell_j(\chi_2)\, \ell_k(\chi_3), \tag{4.48}$$

where

$$\widetilde{\boldsymbol{u}}_{q,ijk}(t) = \boldsymbol{u}\Big(\vec{x}_q(\vec{\xi}_{ijk}), t\Big) \tag{4.49}$$

are the time-dependent polynomial coefficients of the tensor product ansatz evaluated on the set of interpolation nodes

$$\vec{\xi}_{ijk} = (\xi_i, \xi_j, \xi_k)^T \in \mathcal{I}^3, \quad i, j, k = 1, \ldots, n. \tag{4.50}$$

In this thesis, we call $\widetilde{\boldsymbol{u}}_{q,ijk}(t)$ nodal values and tag them with a tilde sign on top. Note, that (4.48) is equivalent to (4.2).

Remark. The Lagrange polynomials satisfy the Kronecker property, i.e.

$$\ell_j(\xi_i) = \delta_{i,j}. \tag{4.51}$$

Analog to (4.3), we can easily calculate the spatial derivative of (4.48) as

$$\partial_{\chi_d} \boldsymbol{u}(\vec{x}_q(\vec{\chi}), t) \approx \partial_{\chi_d} \boldsymbol{p}_q(\vec{\chi}, t) = \sum_{i,j,k=1}^{n} \widetilde{\boldsymbol{u}}_{q,ijk}(t)\, \partial_{\chi_d} \ell_i(\chi_1)\, \ell_j(\chi_2)\, \ell_k(\chi_3). \tag{4.52}$$

The derivatives of the Lagrange polynomials at nodes ξ_i can be precomputed and conveniently stored into a matrix

$$(\boldsymbol{D})_{ij} = \partial_\chi \ell_j(\chi)\Big|_{\xi_i}, \quad i, j = 1, \ldots, n, \tag{4.53}$$

which we coin the differentiation matrix $\boldsymbol{D} \in \mathbb{R}^{n\times n}$. Integration over the reference element

is done analogously to (4.4) by

$$\overline{\boldsymbol{u}}_q(t) \approx \int_{\mathcal{I}^3} \boldsymbol{p}_q(\vec{\chi}, t)\,\mathrm{d}\vec{\chi} = \sum_{i,j,k=1}^{n} \widetilde{\boldsymbol{u}}_{q,ijk}(t) \underbrace{\int_{\mathcal{I}^3} \ell_i(\chi_1)\,\ell_j(\chi_2)\,\ell_k(\chi_3)\,\mathrm{d}\vec{\chi}}_{=\omega_i\,\omega_j\,\omega_k\,=\,\text{const.}}. \tag{4.54}$$

The integral over the Lagrange polynomials in reference space gives the so-called quadrature weights and the sum of all weights is by construction one:

$$\omega_i = \int_{\mathcal{I}} \ell_i(\chi)\,\mathrm{d}\chi \quad \text{and} \quad \sum_{i}^{n} \omega_i = 1. \tag{4.55}$$

We collect the quadrature weights into a diagonal mass matrix $\boldsymbol{M} \in \mathbb{R}^{n\times n}$

$$(\boldsymbol{M})_{ij} = \delta_{ij}\,\omega_i, \quad i,j = 1,\ldots,n. \tag{4.56}$$

Although the numerical quadrature (4.54) is easy to implement and very efficient, it unfortunately introduces integration errors in case of highly nonlinear equations which may accumulate over time and in the worst case causes a breakdown of the numerics. This so-called aliasing is a well-understood problem in spectral-like schemes (Orszag 1971; Kirby and Sherwin 2006), but is a deliberately accepted tradeoff for the gains in speed and efficiency. In the FE community, this controversial topic is even condemned a variational crime (Strang 1972).

In spite of that, a proper choice of "good" quadrature nodes ξ_i has a great impact on the accuracy and stability of the DG scheme. In this work, we look at two choices of nodes: the Legendre-Gauss, in short Gauss, nodes and the Legendre-Gauss-Lobatto, in short Lobatto, nodes. These quadrature nodes are computed as roots of the Legendre polynomials (Abramowitz and Stegun 1965) and are internally not uniformly distributed. The Gauss quadrature rule is exact for polynomials of degree $2\,n-1$ and considered "optimal", while the Lobatto quadrature rule is only exact for polynomials of degree $2\,n-3$.

Legendre-Gauss Quadrature

The one-dimensional nodes of the Gauss rule are given as the roots of the Legendre polynomial $P_k(\xi)$ which are solutions to Legendre's differential equation. It reads

$$\frac{\mathrm{d}}{\mathrm{d}\xi}\Big((1-\xi^2)\frac{\mathrm{d}}{\mathrm{d}\xi}P_i(\xi)\Big) + i\,(i+1)\,P_i(\xi) = 0, \quad i = \mathbb{N}. \tag{4.57}$$

Given the first two polynomials $P_0(\xi) = 1$ and $P_1(\xi) = \xi$, the higher order polynomials can be computed with the recursion formula (Johansson and Mezzarobba 2018)

$$(i+1)\,P_{i+1}(\xi) - (2\,i+1)\,\xi\,P_i(\xi) + i\,P_{i-1}(\xi) = 0. \tag{4.58}$$

Above formula is solved numerically by means of the Newton-Raphson method (Kelley 2003). As starting values of the iterative root finding, approximate expressions for the roots (Lether and Wenston 1995) can be used,

$$\hat{\xi}_i \approx \cos\left(\pi\,\frac{2\,i-1}{2\,n}\right), \quad i = 1,\ldots,n. \tag{4.59}$$

The corresponding weights are calculated as (Abramowitz and Stegun 1965)

$$\hat{\omega}_i = \frac{2}{(1-\hat{\xi}_i^2)\partial_\xi P_n(\hat{\xi}_i)^2}, \quad i = 1,\ldots,n. \tag{4.60}$$

The quadrature nodes are only computed once and stored at the beginning of the simulation. In fact, for our targeted order $n = 4$, analytical expressions (Beyer 1991) for the quadrature nodes and weights are readily available:

$$\hat{\xi}_{1,4} = \pm\sqrt{\frac{3}{7} + \frac{2}{7}\sqrt{\frac{6}{5}}}, \qquad \hat{\omega}_{1,4} = \frac{18-\sqrt{30}}{36},$$
$$\hat{\xi}_{2,3} = \pm\sqrt{\frac{3}{7} - \frac{2}{7}\sqrt{\frac{6}{5}}}, \qquad \hat{\omega}_{2,3} = \frac{18+\sqrt{30}}{36}.$$

Note, that the above quadrature nodes and weights are usually defined for the interval $[-1,1]$. For our purposes, we transfer them to $\mathcal{I}$ beforehand, i.e.

$$\xi_i = \frac{1}{2}\,\hat{\xi}_i \quad \text{and} \quad \omega_i = \frac{1}{2}\,\hat{\omega}_i.$$

Legendre-Gauss-Lobatto Quadrature

The Lobatto quadrature is very similar to the Gauss quadrature rule, however, due to the additional condition of including the integration boundaries, they lose in accuracy. The nodes are the roots of the function $(1-\xi^2)\partial_\xi P_{n-1}(\xi)$ with corresponding weights given by Abramowitz and Stegun (1965)

$$\hat{\omega}_i = \frac{2}{n\,(n-1)\,P_{n-1}(\hat{\xi}_i)^2}, \quad i = 2,\dots,n-1. \tag{4.61}$$

As for the Gauss quadrature, for our targeted order of $n = 4$, there are analytical expressions available for the quadrature nodes and weights (Beyer 1991):

$$\begin{aligned} \hat{\xi}_{1,4} &= \pm 1, & \hat{\omega}_{1,4} &= \frac{5}{6}, \\ \hat{\xi}_{2,3} &= \pm\frac{1}{5}\sqrt{5}, & \hat{\omega}_{2,3} &= \frac{1}{6}. \end{aligned}$$

Note, that the above quadrature nodes and weights are defined for the interval $[-1,1]$ and we transfer them to $\mathcal{I}$ by simple multiplication with $1/2$.

Since Lobatto nodes include the end points in the approximation, the associated derivative operators have the so-called summation-by-parts property (SBP) which allows to discretely mimic the integration-by-parts step (4.41) and paves the way for efficient entropy stable DG schemes. Moreover, Lobatto-based DG schemes are generally easier to implement, are algorithmically less complex and allow larger timesteps compared to the Gauss quadrature rule (Parter 1999; Gassner and Kopriva 2011).

Collocation Scheme

We follow the standard procedure in Kopriva (2009b) and collocate the solution approximation (4.48) with the numerical differentiation (4.52) and the numerical quadrature (4.54). Inserting into the semi-discrete weak-form (4.44), our collocation scheme in semi-

discrete weak form reads

$$\begin{aligned}\dot{\widetilde{\boldsymbol{u}}}_{q,ijk} = &- \frac{1}{\omega_i \, \Delta x_q}\left(\boldsymbol{B}_i^+ \, (\widetilde{\boldsymbol{f}}_1^*)_{q+\frac{1}{2},jk} - \boldsymbol{B}_i^- \, (\widetilde{\boldsymbol{f}}_1^*)_{q-\frac{1}{2},jk} - \sum_{l=1}^{n} \omega_l \, \boldsymbol{D}_{li} \, (\widetilde{\boldsymbol{f}}_1)_{q,ljk}\right) \\ &- \frac{1}{\omega_j \, \Delta y_q}\left(\boldsymbol{B}_j^+ \, (\widetilde{\boldsymbol{f}}_2^*)_{q+\frac{1}{2},ik} - \boldsymbol{B}_j^- \, (\widetilde{\boldsymbol{f}}_2^*)_{q-\frac{1}{2},ik} - \sum_{l=1}^{n} \omega_l \, \boldsymbol{D}_{lj} \, (\widetilde{\boldsymbol{f}}_2)_{q,ilk}\right) \\ &- \frac{1}{\omega_k \, \Delta z_q}\left(\boldsymbol{B}_k^+ \, (\widetilde{\boldsymbol{f}}_3^*)_{q+\frac{1}{2},ij} - \boldsymbol{B}_k^- \, (\widetilde{\boldsymbol{f}}_3^*)_{q-\frac{1}{2},ij} - \sum_{l=1}^{n} \omega_l \, \boldsymbol{D}_{lk} \, (\widetilde{\boldsymbol{f}}_3)_{q,ijl}\right) + \widetilde{\boldsymbol{\Upsilon}}_{q,ijk}.\end{aligned} \tag{4.62}$$

Note, that we added the non-conservative source terms $\dot{\widetilde{\boldsymbol{\Upsilon}}}_{q,ijk}$ which we omitted in (4.44) for reasons of clarity and brevity. The non-conservative source terms are discretized by the DG machinery analogously to the fluxes and are discussed later in text.

The nodal volume flux

$$(\widetilde{\boldsymbol{f}}_d)_{q,ijk} = \boldsymbol{f}_d\Big(\widetilde{\boldsymbol{u}}_{q,ijk}\Big) \tag{4.63}$$

is computed from the polynomial coefficients (4.49) and the boundary interpolation operators read

$$\boldsymbol{B}_i^{\pm} = \ell_i\left(\pm\frac{1}{2}\right), \quad i = 1, \ldots, n. \tag{4.64}$$

The surface fluxes in x-direction

$$(\widetilde{\boldsymbol{f}}_1^*)_{q\pm\frac{1}{2},jk} = \boldsymbol{f}_1^*\Big(\widetilde{\boldsymbol{u}}^+_{q-\frac{1}{2}\pm\frac{1}{2},jk}, \widetilde{\boldsymbol{u}}^-_{q+\frac{1}{2}\pm\frac{1}{2},jk}\Big) \tag{4.65}$$

are calculated analogously to the FV scheme with the Rusanov flux (2.27). The interpolated element boundary values in x-direction read

$$\widetilde{\boldsymbol{u}}^{\pm}_{q\pm\frac{1}{2},jk} = \sum_{i=1}^{N} \boldsymbol{B}_i^{\pm} \, \widetilde{\boldsymbol{u}}_{q,ijk} \tag{4.66}$$

at the common interface $q\pm\frac{1}{2}$ between neighboring elements q and $q\pm1$. The computations in y- and z-direction are done analogously. Note that the notation for indexing the element interfaces, i.e. $q \pm \frac{1}{2}$, is to be understood in an abstract sense. Thus, $(\widetilde{\boldsymbol{f}_1^*})_{q\pm\frac{1}{2}}$, $(\widetilde{\boldsymbol{f}_2^*})_{q\pm\frac{1}{2}}$ and $(\widetilde{\boldsymbol{f}_3^*})_{q\pm\frac{1}{2}}$ point to the faces in x-, y- and z-direction, respectively.

Non-conservative Source Terms

We split the discretization of the Powell term (3.23) into a surface and volume part and write

$$\begin{aligned}\widetilde{\boldsymbol{\Upsilon}}^{\text{Powell}}_{q,ijk} = &- \frac{1}{\omega_i\,\Delta x_q}\Big(\boldsymbol{B}_i^+ \left\{\!\!\left\{\widetilde{B}_1\right\}\!\!\right\}_{q+\frac{1}{2},jk} - \boldsymbol{B}_i^- \left\{\!\!\left\{\widetilde{B}_1\right\}\!\!\right\}_{q-\frac{1}{2},jk} - \sum_{l=1}^{n}\omega_l\,\boldsymbol{D}_{li}\,(\widetilde{B}_1)_{q,ljk}\Big)\widetilde{\boldsymbol{\Phi}}^{\text{Powell}}_{q,ijk}\\ &- \frac{1}{\omega_j\,\Delta y_q}\Big(\boldsymbol{B}_j^+ \left\{\!\!\left\{\widetilde{B}_2\right\}\!\!\right\}_{q+\frac{1}{2},ik} - \boldsymbol{B}_j^- \left\{\!\!\left\{\widetilde{B}_2\right\}\!\!\right\}_{q-\frac{1}{2},ik} - \sum_{l=1}^{n}\omega_l\,\boldsymbol{D}_{lj}\,(\widetilde{B}_2)_{q,ilk}\Big)\widetilde{\boldsymbol{\Phi}}^{\text{Powell}}_{q,ijk}\\ &- \frac{1}{\omega_k\,\Delta z_q}\Big(\boldsymbol{B}_k^+ \left\{\!\!\left\{\widetilde{B}_3\right\}\!\!\right\}_{q+\frac{1}{2},ij} - \boldsymbol{B}_k^- \left\{\!\!\left\{\widetilde{B}_3\right\}\!\!\right\}_{q-\frac{1}{2},ij} - \sum_{l=1}^{n}\omega_l\,\boldsymbol{D}_{lk}\,(\widetilde{B}_3)_{q,ijl}\Big)\widetilde{\boldsymbol{\Phi}}^{\text{Powell}}_{q,ijk}.\end{aligned} \tag{4.67}$$

The source vector is calculated by inserting the nodal states into expression (3.24)

$$\widetilde{\boldsymbol{\Phi}}^{\text{Powell}}_{q,ijk} = \boldsymbol{\Phi}^{\text{Powell}}\Big(\widetilde{\boldsymbol{u}}_{q,ijk}\Big).$$

The element interface averages are calculated as

$$\left\{\!\!\left\{\cdot\right\}\!\!\right\}_{q\pm\frac{1}{2},jk} = \frac{1}{2}\Big((\cdot)^+_{q\pm\frac{1}{2},jk} + (\cdot)^-_{q\pm\frac{1}{2},jk}\Big) \tag{4.68}$$

from the interpolated interface values (4.66). Analog to above, the GLM term (3.25) is then

$$\begin{aligned}\widetilde{\boldsymbol{\Upsilon}}^{\text{GLM}}_{q,ijk} = &- \frac{(\widetilde{v}_1)_{q,ijk}}{\omega_i\,\Delta x_q}\Big(\boldsymbol{B}_i^+ \left\{\!\!\left\{\widetilde{\Psi}\right\}\!\!\right\}_{q+\frac{1}{2},jk} - \boldsymbol{B}_i^- \left\{\!\!\left\{\widetilde{\Psi}\right\}\!\!\right\}_{q-\frac{1}{2},jk} - \sum_{l=1}^{n}\omega_l\,\boldsymbol{D}_{li}\,\widetilde{\Psi}_{q,ljk}\Big)\widetilde{\boldsymbol{\Phi}}^{\text{GLM}}_{q,ijk}\\ &- \frac{(\widetilde{v}_2)_{q,ijk}}{\omega_j\,\Delta y_q}\Big(\boldsymbol{B}_j^+ \left\{\!\!\left\{\widetilde{\Psi}\right\}\!\!\right\}_{q+\frac{1}{2},ik} - \boldsymbol{B}_j^- \left\{\!\!\left\{\widetilde{\Psi}\right\}\!\!\right\}_{q-\frac{1}{2},ik} - \sum_{l=1}^{n}\omega_l\,\boldsymbol{D}_{lj}\,\widetilde{\Psi}_{q,ilk}\Big)\widetilde{\boldsymbol{\Phi}}^{\text{GLM}}_{q,ijk}\\ &- \frac{(\widetilde{v}_3)_{q,ijk}}{\omega_k\,\Delta z_q}\Big(\boldsymbol{B}_k^+ \left\{\!\!\left\{\widetilde{\Psi}\right\}\!\!\right\}_{q+\frac{1}{2},ij} - \boldsymbol{B}_k^- \left\{\!\!\left\{\widetilde{\Psi}\right\}\!\!\right\}_{q-\frac{1}{2},ij} - \sum_{l=1}^{n}\omega_l\,\boldsymbol{D}_{lk}\,\widetilde{\Psi}_{q,ijl}\Big)\widetilde{\boldsymbol{\Phi}}^{\text{GLM}}_{q,ijk}\\ &- \boldsymbol{\Phi}^{\text{damp}}\Big(\widetilde{\boldsymbol{u}}_{q,ijk}\Big).\end{aligned} \tag{4.69}$$

The source vector is calculated by inserting the nodal states into expression (3.26)

$$\widetilde{\boldsymbol{\Phi}}^{\text{GLM}}_{q,ijk} = \boldsymbol{\Phi}^{\text{GLM}}\Big(\widetilde{\boldsymbol{u}}_{q,ijk}\Big).$$

Remark. The scheme (4.62) is conservative except for the contribution of the non-

conservative source term

$$\sum_{q}^{Q} |\Omega_q| \sum_{ijk=1}^{n} \dot{\widetilde{\boldsymbol{u}}}_{q,ijk}\, \omega_i\, \omega_j\, \omega_k = \sum_{q}^{Q} |\Omega_q| \sum_{ijk=1}^{n} \widetilde{\boldsymbol{\Upsilon}}_{q,ijk}\, \omega_i\, \omega_j\, \omega_k \overset{\nabla\cdot\vec{B}\to 0}{\longrightarrow} \mathbf{0}. \tag{4.70}$$

which vanishes when the divergence error goes to zero.

4.6.2 CFL Condition

As for the FV method the timestep size of the scheme has to be restricted in order to maintain stability. The DG's spatial operator spectra are stiffer then FV methods, thus require smaller timesteps and also need more stable time integration methods, such as high order RK methods (Warburton and Hagstrom 2008; Toulorge and Desmet 2011; Chalmers et al. 2014).

To calculate a stable timestep for DG the maximum eigenvalue estimate (3.35) for ideal GLM-MHD is evaluated on all node values $\widetilde{\boldsymbol{u}}_{q,\vec{i}}$ in each element Ω_q. In three dimensions it reads as

$$\widetilde{\lambda}_q^{\max} = \max_{ijk=1}^{n} \lambda^{\max}\Big(\widetilde{\boldsymbol{u}}_{q,ijk}\Big). \tag{4.71}$$

Then the maximal timestep is estimated by the CFL condition (4.13) as

$$\Delta t := \frac{CFL}{d} \min_{q} \frac{\min(\Delta x_q, \Delta y_q, \Delta z_q)}{n\, \widetilde{\lambda}_q^{\max}}, \tag{4.72}$$

where $d = 3, CFL \in (0, 1)$ and n is the number of nodes in each direction of the element Ω_q. Furthermore, we calculate the global hyperbolic correction speed (3.28) as

$$\widetilde{c_{\mathrm{H}}} = \max_{q} \max_{d=1}^{3} \max_{ijk=1}^{N} c_d\Big(\widetilde{\boldsymbol{u}}_{q,ijk}\Big). \tag{4.73}$$

Compared to our discussion of the timestep estimate for FV methods in Section 4.5.2, the CFL condition is stricter (Warburton and Hagstrom 2008), hence we take half of before, i.e. $CFL = 0.4$.

4.6.3 Experimental Order of Convergence

To study the convergence properties of our DG implementation, we compute the numerical errors, that is the norm of the difference of the numerical solution u_{num} from the reference

solution u_{ref} with node values. As for FV schemes, we look at the ∞-norm, i.e.

$$\left|\left|\widetilde{u}_{\text{num}} - \widetilde{u}_{\text{ref}}\right|\right|_{\infty} = \max_{q=1}^{Q} \max_{ijk=1}^{n} \left|\widetilde{u}_{\text{num},q,ijk} - \widetilde{u}_{\text{ref},q,ijk}\right| \tag{4.74}$$

and the 2-norm, i.e.

$$\left|\left|\widetilde{u}_{\text{num}} - \widetilde{u}_{\text{ref}}\right|\right|_{2} = \left[|\Omega|^{-1} \sum_{q=1}^{Q} \sum_{ijk=1}^{n} V_q \, \omega_i \, \omega_j \, \omega_k \left|\widetilde{u}_{\text{num},q,ijk} - \widetilde{u}_{\text{ref},q,ijk}\right|^2 \right]^{\frac{1}{2}}, \tag{4.75}$$

where $V_q = \Delta x_q \, \Delta y_q \, \Delta z_q$ is the volume of element Ω_q and $|\Omega|$ is the volume of the computational domain.

For the exact solution, we again use the MHD vortex problem (4.27) and for time integration we use the LS-RK(5,4) scheme. We look at the total pressure (3.20) and calculate the ∞-norm and 2-norm according to (4.74) and (4.75). The results for the fourth order Standard Gauss DG and Standard Lobatto DG in Table 4.3 and Table 4.4 confirm the expected EOCs of around four. For this test, Standard Gauss DG is more accurate and more robust than Standard Lobatto DG, since the latter crashes at the lowest resolutions.

Table 4.3: EOC of total pressure P of the MHD vortex problem (4.27) run by the fourth order Standard DG with Gauss quadrature.

DOF	$\|\|P\|\|_\infty$	$\|\|P\|\|_2$	EOC_∞	EOC_2
16^2	1.398e-00	2.024e-00	n/a	n/a
32^2	6.114e-01	7.630e-01	1.19	1.41
64^2	6.324e-02	1.046e-01	3.27	2.87
128^2	7.886e-03	9.801e-03	3.00	3.42
256^2	2.090e-04	1.903e-04	5.24	5.69
512^2	1.048e-05	6.742e-06	4.32	4.82
1024^2	6.962e-07	3.815e-07	3.91	4.14

Table 4.4: EOC of total pressure P of the MHD vortex problem (4.27) run by the fourth order Standard DG with Lobatto quadrature. The simulations crash for resolutions 16^2 and 32^2 DOF.

DOF	$\|\|P\|\|_\infty$	$\|\|P\|\|_2$	EOC_∞	EOC_2
16^2	n/a	n/a	n/a	n/a
32^2	n/a	n/a	n/a	n/a
64^2	2.354e-01	3.055e-01	n/a	n/a
128^2	5.534e-02	7.948e-02	2.09	1.94
256^2	6.527e-03	4.000e-03	3.08	4.31
512^2	2.552e-04	1.436e-04	4.68	4.80
1024^2	1.660e-05	8.150e-06	3.94	4.14

4.6.4 Element Entropy Production Rate

We discretize the conservation law of the cell entropy (2.14) by the DG machinery and get

$$\begin{aligned}\dot{\tilde{\mathcal{S}}}_{q,ijk} = &- \frac{1}{\omega_i \, \Delta x_q}\left(\boldsymbol{B}_i^+ \, (\widetilde{\mathcal{F}}_1^*)_{q+\frac{1}{2},jk} - \boldsymbol{B}_i^- \, (\widetilde{\mathcal{F}}_1^*)_{q-\frac{1}{2},jk} - \sum_{l=1}^{n} \omega_l \, \boldsymbol{D}_{li} \, (\widetilde{\mathcal{F}}_1)_{q,ljk}\right) \\ &- \frac{1}{\omega_j \, \Delta y_q}\left(\boldsymbol{B}_j^+ \, (\widetilde{\mathcal{F}}_2^*)_{q+\frac{1}{2},ik} - \boldsymbol{B}_j^- \, (\widetilde{\mathcal{F}}_2^*)_{q-\frac{1}{2},ik} - \sum_{l=1}^{n} \omega_l \, \boldsymbol{D}_{lj} \, (\widetilde{\mathcal{F}}_2)_{q,ilk}\right) \\ &- \frac{1}{\omega_k \, \Delta z_q}\left(\boldsymbol{B}_k^+ \, (\widetilde{\mathcal{F}}_3^*)_{q+\frac{1}{2},ij} - \boldsymbol{B}_k^- \, (\widetilde{\mathcal{F}}_3^*)_{q-\frac{1}{2},ij} - \sum_{l=1}^{n} \omega_l \, \boldsymbol{D}_{lk} \, (\widetilde{\mathcal{F}}_3)_{q,ijl}\right)\end{aligned} \tag{4.76}$$

The surface entropy fluxes $\mathcal{F}_d^*$ are equivalent to the FV surface fluxes (4.31) with

$$(\widetilde{\mathcal{F}}_d^*)_{q\pm\frac{1}{2},ij} = \mathcal{F}^*\left(\tilde{\boldsymbol{u}}^+_{q\pm\frac{1}{2},ij}, \tilde{\boldsymbol{u}}^-_{q\pm\frac{1}{2},ij}\right) \tag{4.77}$$

The computations in y- and z-direction are done analogously. The exact form of the volume entropy fluxes $\widetilde{\mathcal{F}}_d$ is irrelevant, since they will cancel out in our next step. We take the dot product of the entropy vector $\boldsymbol{w}$ with the weak-form DG scheme (4.62) and subtract the scheme (4.76) to get a measure for the entropy production rate $\overline{\Delta \dot{\mathcal{S}}}_q$ of element Ω_q by

$$\overline{\Delta \dot{\mathcal{S}}}_q \approx \sum_{ijk=1}^{n} \left(\widetilde{\boldsymbol{w}}_{q,ijk} \cdot \dot{\tilde{\boldsymbol{u}}}_{q,ijk} \; - \; \dot{\tilde{\mathcal{S}}}_{q,ijk}\right) \omega_i \, \omega_j \, \omega_k. \tag{4.78}$$

However, above formula is only exact for quadrature rules, where the outermost collocation nodes lie on the element boundaries, e.g., the Lobatto quadrature. The reconstruction

of boundary states in the standard DG scheme (4.62) is done on conservative variables and is not consistent to the evaluation of the surface fluxes in entropy variable space according to (2.19). This issue is fixed by replacing the numerical fluxes in (4.62) with entropy projected fluxes $(\widetilde{\boldsymbol{f}_d^*})_{q\pm\frac{1}{2},ij} \rightarrow (\widetilde{\widetilde{\boldsymbol{f}_d^*}})_{q\pm\frac{1}{2},ij}$, where

$$(\widetilde{\widetilde{\boldsymbol{f}_d^*}})_{q\pm\frac{1}{2},ij} = \boldsymbol{f}^*\Big(\widetilde{\widetilde{\boldsymbol{u}}}^+_{q\pm\frac{1}{2},ij}, \widetilde{\widetilde{\boldsymbol{u}}}^-_{q\pm\frac{1}{2},ij}\Big) \tag{4.79}$$

and entropy projected states at element boundaries in x-direction

$$\widetilde{\widetilde{\boldsymbol{u}}}^{\pm}_{q\pm\frac{1}{2},jk} = \boldsymbol{u}\Big(\widetilde{\boldsymbol{w}}^{\pm}_{q\pm\frac{1}{2},jk}\Big) = \boldsymbol{u}\bigg(\sum_{i=1}^{N} \boldsymbol{B}_i^{\pm}\, \boldsymbol{w}(\widetilde{\boldsymbol{u}}_{q,ijk})\bigg). \tag{4.80}$$

Here, we introduce the double tilde sign above a symbol, which signifies an entropy projected quantity. The computations in y- and z-direction are done analogously. We finally write

$$\overline{\dot{\Delta\mathcal{S}}}_q = \sum_{ijk=1}^{n} \Big(\widetilde{\boldsymbol{w}}_{q,ijk} \cdot \dot{\widetilde{\widetilde{\boldsymbol{u}}}}_{q,ijk} \;-\; \dot{\widetilde{\widetilde{\mathcal{S}}}}_{q,ijk}\Big)\, \omega_i\, \omega_j\, \omega_k, \tag{4.81}$$

where the exact temporal change of entropy per element reads

$$\begin{aligned}
\dot{\widetilde{\widetilde{\mathcal{S}}}}_{q,ijk} = &- \frac{1}{\omega_i\, \Delta x_q}\Big(\boldsymbol{B}_i^+\, (\widetilde{\widetilde{\mathcal{F}_1^*}})_{q+\frac{1}{2},jk} - \boldsymbol{B}_i^-\, (\widetilde{\widetilde{\mathcal{F}^*}_1})_{q-\frac{1}{2},jk}\Big) \\
&- \frac{1}{\omega_j\, \Delta y_q}\Big(\boldsymbol{B}_j^+\, (\widetilde{\widetilde{\mathcal{F}_2^*}})_{q+\frac{1}{2},ik} - \boldsymbol{B}_j^-\, (\widetilde{\widetilde{\mathcal{F}^*}_2})_{q-\frac{1}{2},ik}\Big) \\
&- \frac{1}{\omega_k\, \Delta z_q}\Big(\boldsymbol{B}_k^+\, (\widetilde{\widetilde{\mathcal{F}_3^*}})_{q+\frac{1}{2},ij} - \boldsymbol{B}_k^-\, (\widetilde{\widetilde{\mathcal{F}^*}_3})_{q-\frac{1}{2},ij}\Big)
\end{aligned} \tag{4.82}$$

with entropy projected surface entropy fluxes $\mathcal{F}_d^*$ analogously to (4.79), i.e.

$$\widetilde{\widetilde{(\mathcal{F}_d^*)}}_{q\pm\frac{1}{2},ij} = \mathcal{F}^*\Big(\widetilde{\widetilde{\boldsymbol{u}}}^+_{q\pm\frac{1}{2},ij}, \widetilde{\widetilde{\boldsymbol{u}}}^-_{q\pm\frac{1}{2},ij}\Big). \tag{4.83}$$

The computations in y- and z-direction are done analogously. Consequently, the element entropy production rate (4.83) must always be zero or negative for the DG scheme in order to fulfill the entropy inequality (2.15), i.e.

$$\overline{\dot{\Delta\mathcal{S}}}_q \leq 0. \tag{4.84}$$

Clearly, expression (4.82) tells us, that the entropy balance of a DG element is only gov-

erned by the surface fluxes. However, for the Standard DG (4.62) this is not naturally the case. The volume fluxes are not perfectly entropy conservative and erroneously manipulate the balance by introducing entropy errors. This undesirable mechanism is also linked to aliasing and its associated instability issues. Thus, it is hoped that eliminating any entropy errors from the volume fluxes assists in stabilizing the DG scheme. Two approaches for entropy stable DG schemes are discussed in the next section. But before that, we want to assess the robustness of the Standard Gauss DG and Standard Lobatto DG for the weakly magnetized MHD-KHI setup (4.28). Additionally, we investigate if the entropy boundary projected (EBP) surface fluxes for Standard Gauss DG make any difference in terms of robustness and entropy production. The convergence test results for the EBP Standard Gauss DG are given in Table 4.5.

Table 4.5: EOC of total pressure P of the MHD vortex problem (4.27) run by the fourth order Standard DG with Gauss quadrature and EBP.

DOF	$\|\|P\|\|_\infty$	$\|\|P\|\|_2$	EOC_∞	EOC_2
16^2	1.445e-00	2.091e-00	n/a	n/a
32^2	7.689e-01	9.941e-01	0.91	1.07
64^2	1.784e-01	1.959e-01	2.11	2.34
128^2	6.831e-02	4.362e-02	1.38	2.17
256^2	1.328e-02	4.956e-03	2.36	3.14
512^2	5.366e-04	1.536e-04	4.63	5.01
1024^2	1.277e-05	3.768e-06	5.39	5.35

In Figure 4.10 we show the minimum and maximum element entropy production rates (4.79) over simulation time for all three schemes. Clearly, Standard Lobatto DG is less accurate compared to Standard Gauss DG and in terms of entropy error. It is the least robust scheme and crashes first at around $t \approx 3.2$. The entropy projection of Standard Gauss DG shifts the entropy production a bit to the negative side (as desired) but still crashes at $t \approx 2.25$, shortly followed by Standard Gauss DG.

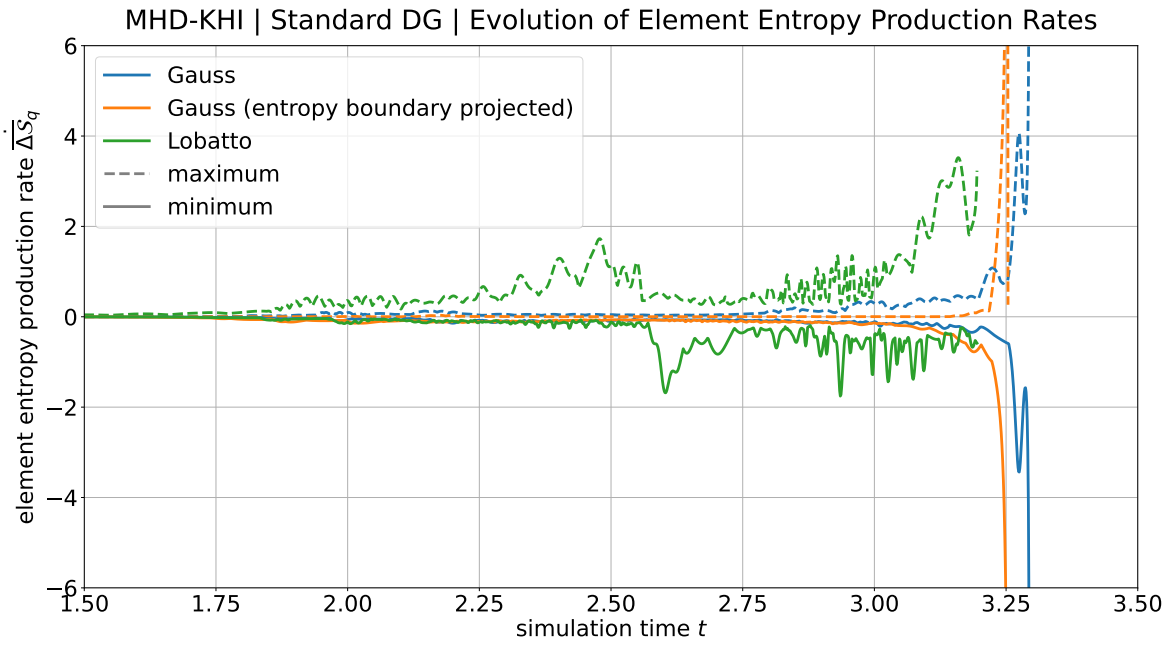

Figure 4.10: Evolution of the entropy production rates for the weakly magnetized MHD-KHI setup (4.28) and with the Standard DG schemes.

Compared to the EC FV scheme (without any explicit dissipation) Standard DG is less robust and crashes earlier. Similar to Figure 4.5, a spike in entropy production indicates an imminent break down of the numerics and a crash of the simulation.

4.6.5 Satisfying the Element Entropy Inequality

In the following, we introduce two approaches to achieve entropy stability for DG schemes. The first approach is to correct any spurious entropy increase produced by the volume term with an algebraic correction term by subtracting surplus entropy production in a conservative and entropy consistent manner. We coin this approach Entropy Corrected DG. The second approach is called Flux Differencing DG and builds on the (generalized) summation-by-parts property of the DG differentiation operator. To reach entropy conservation, respectively guaranteed entropy dissipation, the key ingredient is an entropy conserving numerical flux function.

Entropy Corrected DG

The entropy correction scheme was developed by Abgrall (2018). The idea is intuitive, applicable for a broad range of equations, and is straightforward to implement. Key

element is to find a correction term $\widetilde{\widetilde{\boldsymbol{r}}}_{q,ijk}$ if subtracted from the RHS

$$\dot{\widetilde{\widetilde{\boldsymbol{u}}}}'_{q,ijk} = \dot{\widetilde{\widetilde{\boldsymbol{u}}}}_{q,ijk} - \widetilde{\widetilde{\boldsymbol{r}}}_{q,ijk}, \tag{4.85}$$

such that $\dot{\widetilde{\widetilde{\boldsymbol{u}}}}'_{q,ijk}$ satisfies the element entropy (in-)equality (4.84), but does not break primary conservation. The two requirements define a linear system with always at least two unknowns. The solution, according to Abgrall (2018), reads

$$\widetilde{\widetilde{\boldsymbol{r}}}_{q,ijk} = \frac{\widetilde{\boldsymbol{w}}_{q,ijk} - \overline{\boldsymbol{w}}_q}{\sum_{ijk=1}^{n} (\widetilde{\boldsymbol{w}}_{q,ijk} - \overline{\boldsymbol{w}}_q)^2 \, \omega_i \, \omega_j \, \omega_k} \; \overline{\dot{\Delta\mathcal{S}}}_q \tag{4.86}$$

with $\overline{\boldsymbol{w}}_q = \sum_{ijk=1}^{n} \widetilde{\boldsymbol{w}}_{q,ijk} \, \omega_i \, \omega_j \, \omega_k$ and the element entropy production rate $\overline{\dot{\Delta\mathcal{S}}}_q$ given by (4.81). The correction term (4.86) is conservative, since

$$\sum_{ijk=1}^{n} (\widetilde{\boldsymbol{w}}_{q,ijk} - \overline{\boldsymbol{w}}_q) \, \omega_i \, \omega_j \, \omega_k = 0 \quad \Longrightarrow \quad \sum_{ijk=1}^{n} \widetilde{\widetilde{\boldsymbol{r}}}_{q,ijk} \, \omega_i \, \omega_j \, \omega_k = 0 \tag{4.87}$$

and indeed it does correct the scheme to satisfy equality (4.84), since

$$\begin{aligned}
\overline{\dot{\Delta\mathcal{S}}}'_q &= \sum_{ijk=1}^{n} \left(\widetilde{\boldsymbol{w}}_{q,ijk} \cdot \dot{\widetilde{\widetilde{\boldsymbol{u}}}}'_{q,ijk} \; - \; \dot{\widetilde{\widetilde{\mathcal{S}}}}_{q,ijk} \right) \omega_i \, \omega_j \, \omega_k \\
&= \sum_{ijk=1}^{n} \left(\widetilde{\boldsymbol{w}}_{q,ijk} \cdot \left(\dot{\widetilde{\widetilde{\boldsymbol{u}}}}_{q,ijk} - \widetilde{\widetilde{\boldsymbol{r}}}_{q,ijk} \right) \; - \; \dot{\widetilde{\widetilde{\mathcal{S}}}}_{q,ijk} \right) \omega_i \, \omega_j \, \omega_k \\
&= \overline{\dot{\Delta\mathcal{S}}}_q - \sum_{ijk=1}^{n} \widetilde{\boldsymbol{w}}_{q,ijk} \cdot \widetilde{\widetilde{\boldsymbol{r}}}_{q,ijk} \, \omega_i \, \omega_j \, \omega_k \\
&= \overline{\dot{\Delta\mathcal{S}}}_q - \frac{\sum_{ijk=1}^{n} \widetilde{\boldsymbol{w}}_{q,ijk} \cdot \left(\widetilde{\boldsymbol{w}}_{q,ijk} - \overline{\boldsymbol{w}}_q \right) \omega_i \, \omega_j \, \omega_k}{\sum_{ijk=1}^{n} (\widetilde{\boldsymbol{w}}_{q,ijk} - \overline{\boldsymbol{w}}_q)^2 \, \omega_i \, \omega_j \, \omega_k} \; \overline{\dot{\Delta\mathcal{S}}}_q \\
&= \cancel{\overline{\dot{\Delta\mathcal{S}}}_q} - \cancel{\overline{\dot{\Delta\mathcal{S}}}_q} = 0,
\end{aligned}$$

where we use the fact that

$$\begin{aligned}
&\sum_{ijk=1}^{n} \left(\widetilde{\boldsymbol{w}}_{q,ijk} - \overline{\boldsymbol{w}}_q\right)^2 \omega_i\,\omega_j\,\omega_k \\
&\quad= \sum_{ijk=1}^{n} \left(\widetilde{\boldsymbol{w}}_{q,ijk} \cdot \left(\widetilde{\boldsymbol{w}}_{q,ijk} - \overline{\boldsymbol{w}}_q\right) - \overline{\boldsymbol{w}}_q \cdot \cancel{\left(\widetilde{\boldsymbol{w}}_{q,ijk} - \overline{\boldsymbol{w}}_q\right)}\right) \omega_i\,\omega_j\,\omega_k \\
&\quad= \sum_{ijk=1}^{n} \left(\widetilde{\boldsymbol{w}}_{q,ijk} \cdot \left(\widetilde{\boldsymbol{w}}_{q,ijk} - \overline{\boldsymbol{w}}_q\right)\right) \omega_i\,\omega_j\,\omega_k.
\end{aligned}$$

We get an entropy stable scheme when we cut-off the entropy production rate (4.81) and only trigger the correction whenever $\overline{\dot{\Delta\mathcal{S}}}_q$ is positive. With

$$\widetilde{\widetilde{\boldsymbol{r}}}_{q,ijk} = \frac{\widetilde{\boldsymbol{w}}_{q,ijk} - \overline{\boldsymbol{w}}_q}{\epsilon + \sum_{ijk=1}^{n} \left(\widetilde{\boldsymbol{w}}_{q,ijk} - \overline{\boldsymbol{w}}_q\right)^2 \omega_i\,\omega_j\,\omega_k} \max\left(0, \overline{\dot{\Delta\mathcal{S}}}_q\right) \tag{4.88}$$

we ensure that

$$\overline{\dot{\Delta\mathcal{S}}}_q' \leq 0. \tag{4.89}$$

The parameter $\epsilon := 10^{-20}$ prevents division by zero in case of constant states. A disadvantage of this method is that expression (4.88) can get arbitrarily large introducing stiffness into the system. In some cases we observed positivity issues in density and pressure at the level of element averages causing the simulation to crash.

Table 4.6, Table 4.7, and Table 4.8 document the EOCs of the MHD vortex problem (4.27). In Figure 4.11 we plot the minimum and maximum entropy production rates (4.81) of the three investigated DG variants and we see that entropy correction for Lobatto DG and Gauss DG without EBP does not significantly increase robustness and they crash close before $t \approx 3.5$. Gauss DG with entropy correction and EBP, however, enjoys an increase in stability and is capable to "survive" much longer, but crashes shortly after $t \approx 5.25$.

Table 4.6: EOC of total pressure P of the MHD vortex problem (4.27) run by the fourth order Entropy Corrected Gauss DG without EBP.

DOF	$\|\|P\|\|_\infty$	$\|\|P\|\|_2$	EOC_∞	EOC_2
16^2	1.385e-00	1.988e-00	n/a	n/a
32^2	5.384e-01	7.158e-01	1.36	1.47
64^2	6.271e-02	1.046e-01	3.10	2.78
128^2	8.338e-03	9.926e-03	2.91	3.40
256^2	2.259e-04	1.914e-04	5.21	5.70
512^2	1.064e-05	6.456e-06	4.41	4.89
1024^2	7.489e-07	9.734e-07	3.83	2.73

Table 4.7: EOC of total pressure P of the MHD vortex problem (4.27) run by the fourth order Entropy Corrected Gauss DG with EBP.

DOF	$\|\|P\|\|_\infty$	$\|\|P\|\|_2$	EOC_∞	EOC_2
16^2	1.445e-00	2.091e-00	n/a	n/a
32^2	7.683e-01	9.911e-01	0.91	1.08
64^2	1.784e-01	1.958e-01	2.11	2.34
128^2	6.831e-02	4.362e-02	1.38	2.17
256^2	1.328e-02	4.956e-03	2.36	3.14
512^2	5.367e-04	1.536e-04	4.63	5.01
1024^2	1.277e-05	3.776e-06	5.39	5.35

Table 4.8: EOC of total pressure P of the MHD vortex problem (4.27) run by the fourth order Entropy Corrected Lobatto DG.

DOF	$\|\|P\|\|_\infty$	$\|\|P\|\|_2$	EOC_∞	EOC_2
16^2	1.068e-00	2.125e-00	n/a	n/a
32^2	6.666e-01	1.259e-00	0.68	0.76
64^2	2.081e-01	2.983e-01	1.68	2.08
128^2	5.395e-02	7.830e-02	1.95	1.93
256^2	6.495e-03	4.009e-03	3.05	4.29
512^2	2.472e-04	1.439e-04	4.72	4.80
1024^2	1.658e-05	8.038e-06	3.90	4.16

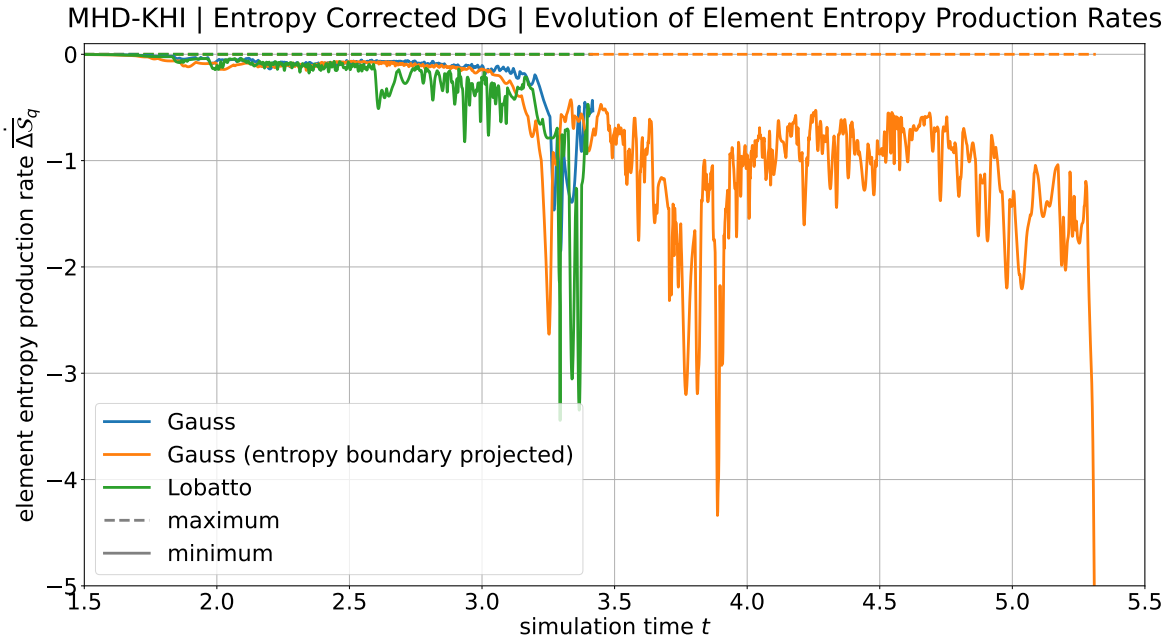

Figure 4.11: Evolution of the entropy production rates for the weakly magnetized MHD-KHI setup (4.28) and with the Entropy Corrected DG schemes.

Flux Differencing DG

Fisher and Carpenter (2013); Fisher et al. (2013); Carpenter et al. (2014) laid the groundwork for a popular class of entropy stable DG schemes by showing that the conditions to develop entropy stable approximations at low order, as we have done for the FV scheme in Section 4.5.5 and Section 4.5.6, immediately apply to high order methods provided the derivative approximation satisfies the summation-by-parts (SBP) property. In the context of entropy stable DG methods, this seminal insight was successfully applied for the shallow water equations (Gassner et al. 2016b; Wintermeyer et al. 2018), compressible Euler equations (Gassner et al. 2016d), compressible Navier-Stokes equations (Gassner et al. 2018) as well as the ideal and resistive MHD equations (Rossmanith 2013; Chandrashekar et al. 2016; Gallego Valencia 2017; Liu et al. 2018b; Bohm 2018). However, all entropy stable DG schemes constructed by the referenced authors rely on the Lobatto quadrature rule (where boundary points are included) in order to construct diagonal norm SBP operators, the key aspect to achieve entropy stability.

Chan (2018) introduced an efficient entropy stable DG scheme for the Gauss quadrature rule based on the construction of general SBP operators. A detailed discussion about general SBP operators in the context of hyperbolic conservation laws is provided, for

example, by Ranocha (2018). The generalized SBP operator is constructed by

$$(\boldsymbol{D}\,\boldsymbol{M}) + (\boldsymbol{D}\,\boldsymbol{M})^T = \left(\boldsymbol{B}^-|\boldsymbol{B}^+\right) \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix} \left(\boldsymbol{B}^-|\boldsymbol{B}^+\right)^T \tag{4.90}$$

where $(\boldsymbol{B}^-|\boldsymbol{B}^+) \in \mathbb{R}^{n\times 2}$ is the matrix of the two boundary interpolation operators (4.64). Again, the SBP property is a direct restatement of the integration-by-parts in the continuous case. Equipped with relation (4.90), we rearrange the semi-discrete, weak-form DG (4.62) according to Chan (2018) and get

$$\begin{aligned} \partial_t \widetilde{\widetilde{\boldsymbol{u}}}_{q,ijk} = &- \frac{1}{\omega_i\,\Delta x_q}\left(\boldsymbol{B}_i^+\,(\widetilde{\widetilde{\boldsymbol{f}_1^{\%}}})_{q+\frac{1}{2},jk} - \boldsymbol{B}_i^-\,(\widetilde{\widetilde{\boldsymbol{f}_1^{\%}}})_{q-\frac{1}{2},jk} - \sum_{l=1}^{n} \boldsymbol{S}_{li}\,(\widetilde{\boldsymbol{f}_1^{\#}})_{q,\{l,i\}jk}\right) \\ &- \frac{1}{\omega_j\,\Delta y_q}\left(\boldsymbol{B}_j^+\,(\widetilde{\widetilde{\boldsymbol{f}_2^{\%}}})_{q+\frac{1}{2},ik} - \boldsymbol{B}_j^-\,(\widetilde{\widetilde{\boldsymbol{f}_2^{\%}}})_{q-\frac{1}{2},ik} - \sum_{l=1}^{n} \boldsymbol{S}_{lj}\,(\widetilde{\boldsymbol{f}_2^{\#}})_{q,i\{l,j\}k}\right) \\ &- \frac{1}{\omega_k\,\Delta z_q}\left(\boldsymbol{B}_k^+\,(\widetilde{\widetilde{\boldsymbol{f}_3^{\%}}})_{q+\frac{1}{2},ij} - \boldsymbol{B}_k^-\,(\widetilde{\widetilde{\boldsymbol{f}_3^{\%}}})_{q-\frac{1}{2},ij} - \sum_{l=1}^{n} \boldsymbol{S}_{lk}\,(\widetilde{\boldsymbol{f}_3^{\#}})_{q,ij\{l,k\}}\right) \end{aligned} \tag{4.91}$$

where $\boldsymbol{S} = \boldsymbol{D}\,\boldsymbol{M} - (\boldsymbol{D}\,\boldsymbol{M})^T$ is the skew-symmetric flux differencing matrix and the volume flux operation is a telescopic sum defined by

$$\sum_{l=1}^{n} \boldsymbol{S}_{li}\,(\widetilde{\boldsymbol{f}_1^{\#}})_{q,\{l,i\}jk} = \sum_{l=1}^{n} \boldsymbol{S}_{li}\,\boldsymbol{f}_1^{\#}\left(\widetilde{\boldsymbol{u}}_{q,ljk}, \widetilde{\boldsymbol{u}}_{q,ijk}\right). \tag{4.92}$$

The special surface flux reads

$$(\widetilde{\widetilde{\boldsymbol{f}_1^{\%}}})_{q\pm\frac{1}{2},jk} = \boldsymbol{f}_1^*\left(\widetilde{\widetilde{\boldsymbol{u}}}^+_{q\pm\frac{1}{2},jk}, \widetilde{\widetilde{\boldsymbol{u}}}^-_{q\pm\frac{1}{2},jk}\right) + \boldsymbol{f}_1^{\#}\left(\widetilde{\widetilde{\boldsymbol{u}}}^{\pm}_{q,jk}, \widetilde{\boldsymbol{u}}_{q,ijk}\right) - \sum_{l=1}^{n} \boldsymbol{B}_l^{\pm}\,\boldsymbol{f}_1^{\#}\left(\widetilde{\widetilde{\boldsymbol{u}}}^{\pm}_{q+\frac{1}{2},jk}, \widetilde{\boldsymbol{u}}_{q,ljk}\right), \tag{4.93}$$

where as before the terms are evaluated on entropy projected states given by (4.80). The computations in y- and z-direction are done analogously.

Any combination of numerical surface, $\boldsymbol{f}^*$, and volume $\boldsymbol{f}^{\#}$ fluxes can be selected, in order to fulfill primary and secondary conservation constraints. As a matter of fact, a common practice to obtain a provably entropy stable scheme is to use the same entropy conservative flux in the volume and the surface, and to add a dissipation term in the surface flux. To get an entropy conservative scheme, the surface numerical flux $\boldsymbol{f}^*$ is simply set equal to the volume numerical flux $\boldsymbol{f}^{\#}$. For our case, we chose $\boldsymbol{f}^*$ to be the Rusanov-type surface flux (2.32) and $\boldsymbol{f}^{\#}$ to the familiar consistent, affordable and entropy

conservative Riemann solver (3.41) for the ideal GLM-MHD equations.

Rueda-Ramírez et al. (2022a) extended the work by Chan (2018), where he solved the compressible Euler equations, to the ideal GLM-MHD equations (3.22). The integration of the non-conservative source terms in Rueda-Ramírez et al. (2022a) is especially elegant and straightforward. Simply add to every numerical volume flux $\boldsymbol{f}^{\#}$ the following two-point source term function:

$$\boldsymbol{f}\left(\boldsymbol{u}^{+},\boldsymbol{u}\right) \rightarrow \boldsymbol{f}\left(\boldsymbol{u}^{+},\boldsymbol{u}\right) + \boldsymbol{\Phi}\left(\boldsymbol{u}^{+},\boldsymbol{u}\right). \tag{4.94}$$

Moreover, Rueda-Ramírez et al. (2022a) also shows detailed proofs that the scheme (4.91) is entropy stable for the ideal GLM-MHD equations if adequate numerical volume and surface fluxes are chosen.

The "generalized" flux differencing scheme (4.91) is directly compatible for both quadrature rules, Gauss and Lobatto. However, Rueda-Ramírez et al. (2022a) thoroughly discusses, that in case of Lobatto nodes, the scheme deflates to the version with a diagonal norm SBP operator presented by, e.g., Bohm (2018) in the case of zero viscosity and zero resistivity. Flux differencing schemes with diagonal norm SBP operators are less complex, have much less computational overhead and, thus, are much more efficient. We compare the runtime performance of both flux differencing variants later in Section 5.5. Here, we are primarily interested in the robustness of both schemes with regard to our numerical tests.

Table 4.9, Table 4.10 and Table 4.11 show the EOC for the MHD vortex (4.27).

Table 4.9: EOC of total pressure P of the MHD vortex problem (4.27) run by the fourth order Flux Diff. Gauss DG without EBP.

DOF	$\|\|P\|\|_\infty$	$\|\|P\|\|_2$	EOC_∞	EOC_2
16^2	n/a	n/a	n/a	n/a
32^2	n/a	n/a	n/a	n/a
64^2	7.653e-02	1.063e-01	n/a	n/a
128^2	1.134e-02	9.819e-03	2.75	3.44
256^2	2.269e-04	1.894e-04	5.64	5.70
512^2	1.074e-05	6.697e-06	4.40	4.82
1024^2	6.976e-07	3.810e-07	3.94	4.14

Table 4.10: EOC of total pressure P of the MHD vortex problem (4.27) run by the fourth order Flux Diff. Gauss DG with EBP.

DOF	$\|\|P\|\|_\infty$	$\|\|P\|\|_2$	EOC_∞	EOC_2
16^2	1.454e-00	2.103e-00	n/a	n/a
32^2	7.465e-01	9.574e-01	0.96	1.14
64^2	1.838e-01	1.960e-01	2.02	2.29
128^2	7.274e-02	4.393e-02	1.34	2.16
256^2	1.341e-02	4.975e-03	2.44	3.14
512^2	5.298e-04	1.517e-04	4.66	5.04
1024^2	1.274e-05	3.764e-06	5.38	5.33

Table 4.11: EOC of total pressure P of the MHD vortex problem (4.27) run by the fourth order Flux Diff. Lobatto DG.

DOF	$\|\|P\|\|_\infty$	$\|\|P\|\|_2$	EOC_∞	EOC_2
16^2	1.177e-00	2.090e-00	n/a	n/a
32^2	7.397e-01	1.165e-00	0.67	0.84
64^2	1.862e-01	2.006e-01	1.99	2.54
128^2	5.493e-02	3.310e-02	1.76	2.60
256^2	7.675e-03	2.945e-03	2.84	3.49
512^2	3.793e-04	1.256e-04	4.34	4.55
1024^2	2.606e-05	7.463e-06	3.86	4.07

In Figure 4.12 we plot the minimum and maximum entropy production rates (4.81) of the three investigate schemes. While Flux Diff. Gauss (without EBP) and Flux Diff. Lobatto crash early on, $t \approx 3.1$ and $t \approx 3.7$ respectively, the Flux Diff. Gauss with EBP is robust enough to "survive" the critical phase at $t \approx 3.7$, visible by a minimum peak in the entropy production rate, and is even able to run till final simulation time $T = 10$. So far, we can report that besides the Minmod FV scheme, only the genuinely entropy stable Flux Diff. Gauss DG with EBP is capable to complete our test setup without crashing.

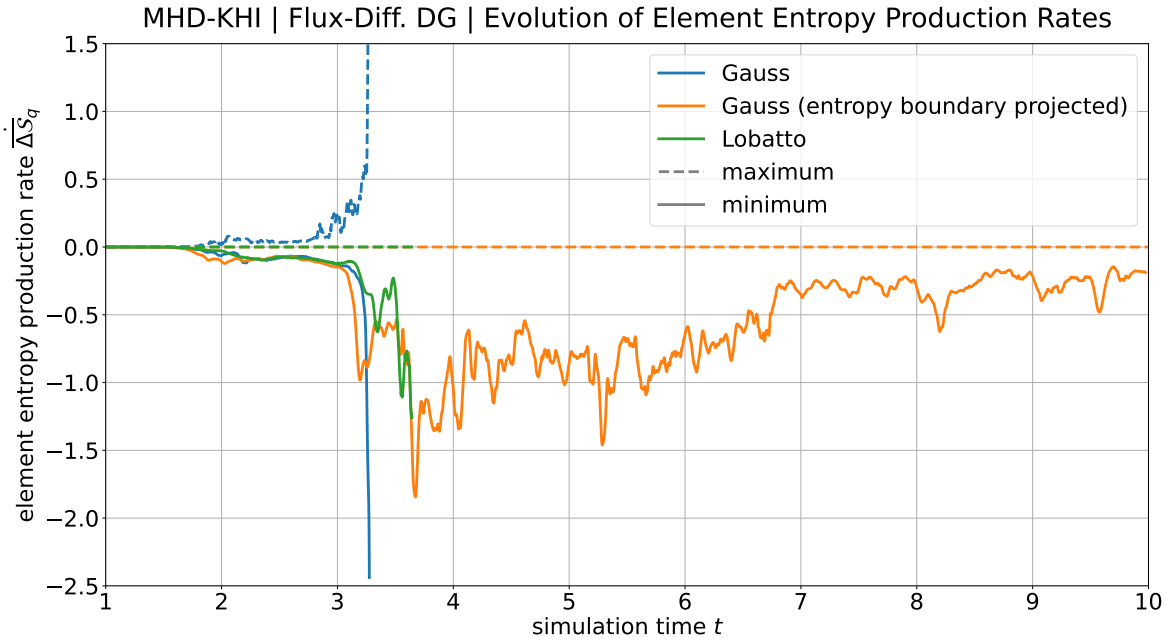

Figure 4.12: Evolution of the entropy production rates for the weakly magnetized MHD-KHI setup (4.28) and with the Flux Differencing DG schemes.

4.7 Convex Blending Scheme

One of the core contributions of this thesis is to augment a high order DG method with sub-element shock capturing capabilities, which allows the robust simulation of highly supersonic turbulent flows featuring strong shocks, as e.g. in astrophysics, without changing the data dependency footprint. Instead of flagging a troubled element and completely switching from the high order DG scheme to the subcell FV scheme, we aim to smoothly transition between both schemes independently for each element. Our aim is to keep the underlying data structures simple by maintaining the solution approximation space and to confine the shock capturing to an element local technique instead of changing the global mesh topology.

In the following, we briefly present the building blocks of our proposed convex blending scheme, which is also documented in a peer-reviewed paper in Markert et al. (2022). Again, the basic idea is to combine the robustness of FV and the accuracy of DG via convex blending in the vicinity of discontinuous flow features. In Markert et al. (2021) this scheme is also coined single-level blending scheme.

4.7.1 Building Blocks

The computational domain Ω is divided into Q non-overlapping blocks and each block holds N^3 mean values. From the perspective of the FV method, the block is divided into N^3 regular subcells of size $\frac{\Delta \vec{x}_q}{N}$ defining the uniform grid (4.15) with midpoints $\vec{\mu}_{\vec{i}}$ and corners $\vec{\mu}_{\vec{i}\pm\frac{1}{2}}$ in the reference space $\mathcal{I}^3 = [-\frac{1}{2}, \frac{1}{2}]^3$. Now, we overlay an N^{th} order DG element with N^3 nodal values over the block of N^3 mean values and get a new scheme that is a hybrid between FV and DG. An illustration of this concept for a one-dimensional fourth order DG element and four mean values ($N = 4$) is shown in Fig. 4.13.

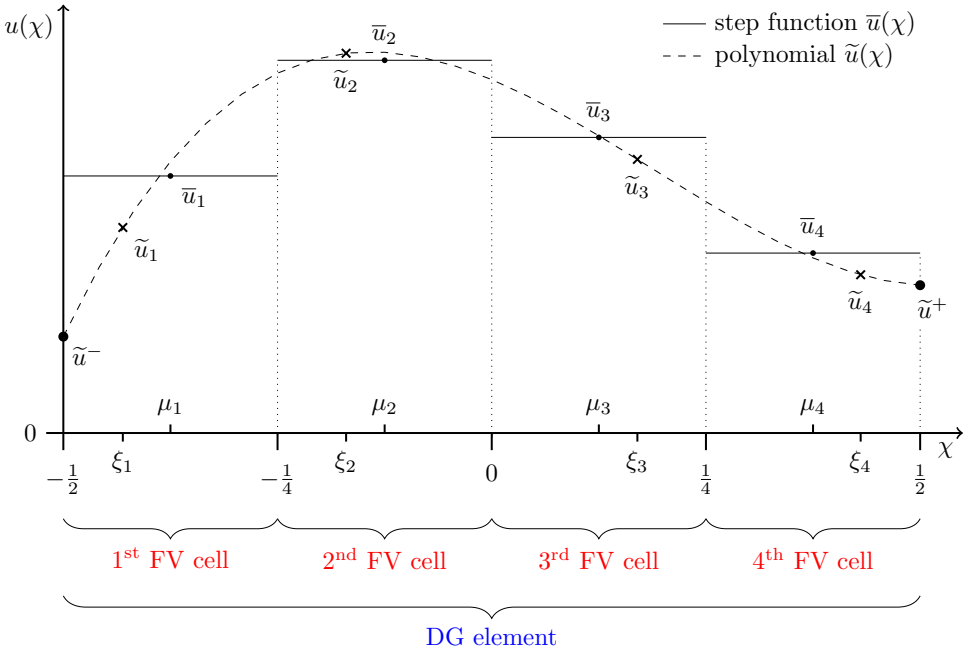

Figure 4.13: 1D schematic of four ($N = 4$) mean values $\overline{\boldsymbol{u}}_i$ and their reconstructed nodal values (polynomial coefficients) $\widetilde{\boldsymbol{u}}_i$ constituting a polynomial of degree 3 spanning over the whole DG element.

Projection & Reconstruction

In order to make FV and DG compatible we construct projection and reconstruction operators transforming between N_μ^3 mean values $\overline{\boldsymbol{u}}_{\vec{i}}$ and N^3 nodal values $\widetilde{\boldsymbol{u}}_{\vec{i}}$. We consider a polynomial $\widetilde{\boldsymbol{u}}(\chi), \chi \in \mathcal{I}$, of degree $N-1$ in one dimension and want to project the polynomial to N_μ mean values $\overline{\boldsymbol{u}}_i$ on N_μ regular subcells. We define the projection matrix operator $\boldsymbol{P}^{(N\to N_\mu)} \in \mathbb{R}^{N_\mu \times N}$ component-wise via the mean values of the polynomial in the

intervals of subcell midpoints μ_i. The ansatz thus is

$$\overline{\boldsymbol{u}}_i = N_\mu \int_{\mu_{i-\frac{1}{2}}}^{\mu_{i+\frac{1}{2}}} \widetilde{\boldsymbol{u}}(\chi)\, d\chi = \sum_{j=1}^{N} \widetilde{\boldsymbol{u}}_j \underbrace{N_\mu \int_{\mu_{i-\frac{1}{2}}}^{\mu_{i+\frac{1}{2}}} \ell_j(\chi)\, d\chi}_{:=\mathcal{P}_{ij}}, \tag{4.95}$$

The integration of the Lagrange polynomial in (4.95) is done exactly with an appropriate quadrature rule.

Remark. Summing the projection operator $\boldsymbol{P}^{(N\to N_\mu)}$ along each column gives

$$\sum_{i=1}^{N_\mu} P_{ij} = N_\mu\, \omega_j. \tag{4.96}$$

Proof.

$$\sum_{i=1}^{N_\mu} P_{ij} = N_\mu \sum_{i=1}^{N_\mu} \int_{\mu_{i-\frac{1}{2}}}^{\mu_{i+\frac{1}{2}}} \ell_j(\chi)\, d\chi = N_\mu \int_{\mu_{1-\frac{1}{2}}}^{\mu_{N_\mu+\frac{1}{2}}} \ell_j(\chi)\, d\xi = N_\mu \int_{-\frac{1}{2}}^{\frac{1}{2}} \ell_j(\chi)\, d\chi = N_\mu\, \omega_j. \tag{4.97}$$

□

In practice, we assign the DG polynomial and the FV grid equal amount of DOF, i.e. $N = N_\mu$. In this case, the projection matrix $\boldsymbol{P}^{(N\to N)}$ becomes quadratic. The operator is non-singular by construction and we simply write $\boldsymbol{P}$ from here on. Moreover, another side benefit is that the inverse $\boldsymbol{R} := \boldsymbol{P}^{-1}$ reconstructs N nodal values from given N mean values. For data blocks in 3D we compute

$$\begin{aligned} \overline{\boldsymbol{u}}_{ijk} &= \sum_{c=1}^{N} P_{kc} \sum_{b=1}^{N} P_{jb} \sum_{a=1}^{N} P_{ia}\ \overline{\boldsymbol{u}}_{abc} := P_{ijk}^{[abc]}\ \widetilde{\boldsymbol{u}}_{[abc]} \quad \text{and} \\ \widetilde{\boldsymbol{u}}_{ijk} &= \sum_{c=1}^{N} R_{kc} \sum_{b=1}^{N} R_{jb} \sum_{a=1}^{N} R_{ia}\ \widetilde{\boldsymbol{u}}_{abc} := R_{ijk}^{[abc]}\ \overline{\boldsymbol{u}}_{[abc]}, \end{aligned} \tag{4.98}$$

where we introduced a variant of the Einstein notation for brevity. That is, indices enclosed in brackets are summed over.

Remark. The weighted sum of the reconstruction operator $\boldsymbol{R}$ along each column gives

$$\sum_{i=1}^{N} \omega_i\, R_{ij} = \frac{1}{N}\,. \tag{4.99}$$

Proof. This property can be shown more elegantly in operator notation. Using property (4.96) and with $\vec{1}_N = (1, \dots, 1)^T \in \mathbb{R}^N$ being the vector of ones and $\vec{\omega} = (\omega_1, \dots, \omega_N)^T$ being the vector of quadrature weights we write

$$\begin{aligned} \vec{1}^T \boldsymbol{P} &= N \vec{\omega}^T \\ \vec{1}^T \boldsymbol{P} \boldsymbol{P}^{-1} &= N \vec{\omega}^T \boldsymbol{P}^{-1} = N \vec{\omega}^T \boldsymbol{R} \\ \frac{1}{N} \vec{1}^T &= \vec{\omega}^T \boldsymbol{R}. \end{aligned}$$

□

It is important to note that depending on the block data the reconstruction (4.98) can give a highly oscillatory polynomial with nonphysical node values such as negative densities or negative pressures. Thus, all reconstructed node values are checked for their validity and if a violation is detected, the DG solution is discarded and 100% of the FV solution is evolved to the next time level.

Convex Blending

We aim to blend the RHS solution of a LOW and HIGH order scheme with a continuous blending factor $\alpha \in [0, 1]$ in a convex manner:

$$\dot{\boldsymbol{u}}_q \;=\; (1 - \alpha_q)\, \dot{\boldsymbol{u}}_q^{\text{LOW}} \;+\; \alpha_q\, \dot{\boldsymbol{u}}_q^{\text{HIGH}}.$$

Note that we do not blend the solutions, but directly the discretizations themselves, i.e. the RHS, which we denote by $\dot{\boldsymbol{u}}_q$. If both schemes operate with different data representations, appropriate transformations ensure compatibility during the blending process. In our case we use the projection operator $\boldsymbol{P}^{(N \to N)}$, introduced in the previous paragraph, in order to transfer the nodal output of the DG operator to subcell mean values. The input for the DG scheme on the other hand, i.e. the nodal coefficients $\widetilde{\boldsymbol{u}}_q$, are reconstructed from the given mean values $\overline{\boldsymbol{u}}_q$. The specific convex blend of our discretizations reads

$$\dot{\overline{\boldsymbol{u}}}_{q,ijk} \;=\; (1 - \alpha_q)\, \dot{\overline{\boldsymbol{u}}}_{q,ijk}^{\text{FV}} \;+\; \alpha_q\, P_{ijk}^{[abc]}\, \dot{\widetilde{\boldsymbol{u}}}_{q,[abc]}^{\text{DG}}. \tag{4.100}$$

We call α_q the *volume blending factor*, which can be chosen to be unique for each element. However, we have to carefully ensure the conservation property of the blending scheme

by finding a common surface flux

$$(\overline{\boldsymbol{f}}_1^*)_{q\pm\frac{1}{2},ij} = (1-\alpha_{q\pm\frac{1}{2}})\,(\overline{\boldsymbol{f}}_1^{*\mathrm{FV}})_{q\pm\frac{1}{2},ij} + \alpha_{q\pm\frac{1}{2}}\,P_{ij}^{[ab]}\,(\widetilde{\boldsymbol{f}}_1^{*\mathrm{DG}})_{q\pm\frac{1}{2},[ab]}. \tag{4.101}$$

We call $\alpha_{q\pm\frac{1}{2}}$ the *surface blending factor*, which is shared between neighboring elements Ω_q and Ω_{q+1}. The outermost fluxes in (4.18) are replaced with expression (4.101):

$$(\overline{\boldsymbol{f}}_1^*)_{q,1jk} \to (\overline{\boldsymbol{f}}_1^*)_{q-\frac{1}{2},jk} \quad \text{and} \quad (\overline{\boldsymbol{f}}_1^*)_{q,Njk} \to (\overline{\boldsymbol{f}}_1^*)_{q+\frac{1}{2},jk}.$$

Likewise, we replace the surface flux in (4.62) with the transformed flux (4.101):

$$(\widetilde{\boldsymbol{f}}_1^*)_{q\pm\frac{1}{2},ij} \to R_{ij}^{[ab]}\,(\overline{\boldsymbol{f}}_1^*)_{q\pm\frac{1}{2},[ab]}.$$

The surface fluxes in y- and z-direction are treated analogously.

Remark. It is easy to see that with $\alpha_q = 0$ and $\alpha_{q\pm\frac{1}{2}} = 0$ for all elements, the pure subcell FV discretization is recovered and with $\alpha_q = 1$ and $\alpha_{q\pm\frac{1}{2}} = 1$ for all elements the blending scheme recovers the underlying pure high order DG method.

Proposition. Given arbitrary blending factors $\alpha_q \in \mathbb{R}$ for each element Ω_q the blending scheme (4.100) is conservative. That is, we discretely integrate the blended discretization over all elements Ω_q with the total number Q and get

$$\sum_q^Q \frac{|\Omega_q|}{N^3} \sum_{ijk}^N \dot{\overline{\boldsymbol{u}}}_{q,ijk} = \boldsymbol{0}. \tag{4.102}$$

The proof is given in Markert et al. (2021).

Calculation of the Blending Factor α

The surface blending factors $\alpha_{q\pm\frac{1}{2}}$ are estimated from the relative differences in the jumps of the element mean values $\overline{\boldsymbol{u}}_{q,ijk}$ and the reconstructed polynomial $\widetilde{\boldsymbol{u}}_{q,ijk}$ at element interfaces $\Omega_{q\pm\frac{1}{2}}$. We transform the interpolated nodal interface values to mean values beforehand:

$$\overline{\kappa}_{q\pm\frac{1}{2},jk}^{\mathrm{DG}} = \mathcal{P}_{jk}^{[bc]}\,\widetilde{\kappa}_{q\pm\frac{1}{2},[bc]},$$

where κ is a freely chosen *indicator variable*, such as density or pressure. Additionally, we introduce the element interface jumps

$$[\![\cdot]\!]_{q\pm\frac{1}{2},jk} = (\cdot)^{-}_{q\pm\frac{1}{2},jk} - (\cdot)^{+}_{q\pm\frac{1}{2},jk} \tag{4.103}$$

illustrated in Figure 4.14. The blending factor in x-direction then reads

$$\alpha_{q\pm\frac{1}{2},jk} = 1 - \mathcal{T}\left(\left[\!\left[\overline{\kappa}^{\mathrm{FV}} \right]\!\right]_{q\pm\frac{1}{2},jk}, \left[\!\left[\overline{\kappa}^{\mathrm{DG}} \right]\!\right]_{q\pm\frac{1}{2},jk} \right) \tag{4.104}$$

with the transfer function

$$\mathcal{T}\left((\cdot)^{\mathrm{FV}}, (\cdot)^{\mathrm{DG}} \right) = \left\lceil \tau_A \frac{\left| (\cdot)^{\mathrm{FV}} - (\cdot)^{\mathrm{DG}} \right| - \tau_S \left| (\cdot)^{\mathrm{FV}} \right|}{\max\left(\left| (\cdot)^{\mathrm{FV}} \right|, 1 \right)} \right\rfloor_1 \tag{4.105}$$

mapping the two input arguments $(\cdot)^{\mathrm{FV}}, (\cdot)^{\mathrm{DG}} \in \mathbb{R}$ to the unit interval $[0, 1]$ as is indicated by the notation $\lceil_0 (\cdot) \rceil_1$ trimming the input $(\cdot)$ for values below 0 and above 1. The *amplification parameter*, $\tau_A := 20$, and the *shifting parameter*, $\tau_S := 1.0$, are fixed for all numerical results shown in this work. Moreover, we choose the thermal pressure (3.31) as the indicator variable, which we justified in Section 3.5 about shocks in astrophysical settings.

For the common surface blending factor we pick the minimum result along the interface:

$$\alpha_{q\pm\frac{1}{2}} = \min_{ij=1}^{N} \alpha_{q\pm\frac{1}{2},jk}. \tag{4.106}$$

For the volume blending factor α_q we calculate the averages of the surface blending factors in each direction, for example in x-dimension

$$\alpha^x_q = \frac{1}{2}\left(\alpha^x_{q-\frac{1}{2}} + \alpha^x_{q+\frac{1}{2}} \right),$$

and determine the minimum among all directions

$$\alpha_q = \min\left\{ \alpha^x_q, \alpha^y_q, \alpha^z_q \right\}. \tag{4.107}$$

The idea of the proposed algorithm is to have a mechanism, which is designed such that

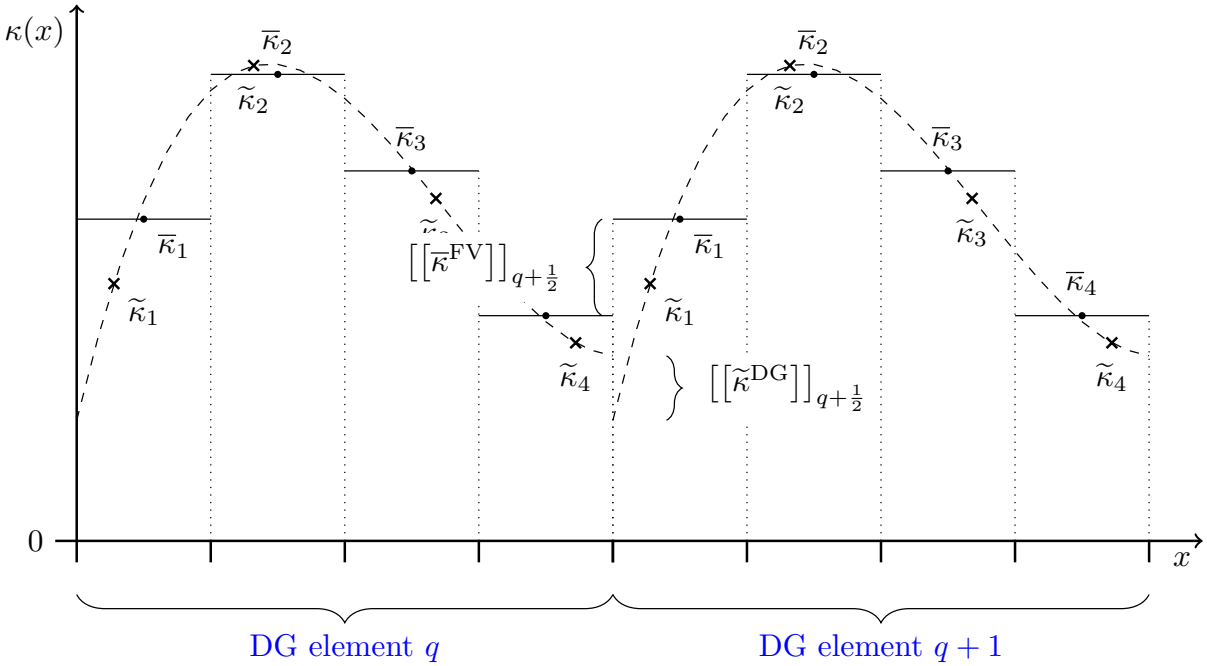

Figure 4.14: 1D schematic of two neighboring elements q and $q+1$ each with four ($N=4$) mean values of the indicator variable $\overline{\kappa}_i$ and their reconstructed polynomial coefficients $\widetilde{\kappa}_i$. The relative difference of the jumps at an element interface is considered to be a measure of the smoothness of the solution at hand.

for well resolved flows the solver yields the full DG solution and gradually shifts to the FV solution in case of discontinuities or strong under-resolution. Additionally, the blending factor is set to zero if the reconstructed polynomial coefficients yield unbound values, such as negative densities or negative pressures. The result is then the 100% second order slope limited FV solution in elements where the reconstruction of a DG polynomial failed. Hence, in a sense, the proposed method with convex blending of a low and high order operator guarantees that the discretization cannot be "worse" than the second order FV method.

In fact, the idea to look at DG interface jumps is not new, e.g. Krivodonova et al. (2004); Chandrashekar et al. (2016), and is based on the fact that for smooth regions high order DG solutions show super-convergent approximation at the outflow boundaries of the elements (Flaherty et al. 2002). Since the (super-)convergence at boundaries is very sensitive to any disturbances, one can deduce that the DG element contains a discontinuous solution whenever the polynomial is "troubled".

4.7.2 CFL Condition

The timestep restriction is computed analogously to the CLF condition for the FV scheme, which we already detailed in Section 4.5.2. The only difference is the CFL constant being the same as for the DG scheme, i.e. $CFL = 0.4$.

4.7.3 Experimental Order of Convergence

The experimental order of convergence is computed analogously to the FV scheme on mean values, which we already detailed in Section 4.5.3. It is important, however, that one has to compute the mean values of the reference solution with high accuracy as they should formally be the exact mean values of the reference solution. Hence, we evaluate the reference solution on quadrature nodes of the same order or higher as in (4.63) and then project to mean values.

Corollary. Given all $\alpha_{q\pm\frac{1}{2}} = 1$ and $\alpha_q = 1$, the blending scheme (4.100) is equivalent to the high order DG scheme.

Proof. Since time integration in explicit Runge Kutta methods is a linear superposition of intermediate states at different time levels, the linear projection and reconstruction operations cancel each other. This statement is easily shown with the Euler step (4.10)

$$\overline{\boldsymbol{u}}_{ijk}(t_1) = \overline{\boldsymbol{u}}_{ijk}(t_0) + \Delta t \; P_{ijk}^{[abc]} \; \dot{\tilde{\boldsymbol{u}}}^{\mathrm{DG}}_{q,[abc]}(t_0).$$

and directly transfers to all RK schemes like, for example, the explicit SSP-RK(4,3) method (4.12) or LS-RK(5,4). Application of the reconstruction operator (4.98) from left yields

$$\begin{aligned} R_{ijk}^{[abc]} \, \overline{\boldsymbol{u}}_{[abc]}(t_1) &= R_{ijk}^{[abc]} \, \overline{\boldsymbol{u}}_{[abc]}(t_0) + \Delta t \; R_{ijk}^{[def]} \; P_{[def]}^{[abc]} \; \dot{\tilde{\boldsymbol{u}}}^{\mathrm{DG}}_{q,[abc]}(t_0) \\ \tilde{\boldsymbol{u}}_{ijk}(t_1) &= \tilde{\boldsymbol{u}}_{ijk}(t_0) + \Delta t \; \dot{\tilde{\boldsymbol{u}}}^{\mathrm{DG}}_{q,[ijk]}(t_0). \end{aligned}$$

□

The results in EOC for our convex blending scheme are documented and discussed in Section 6.3.1.

4.7.4 Element Entropy Production Rate

A straightforward calculation of the element entropy production rate $\overline{\dot{\Delta\mathcal{S}}}_q^{\mathrm{CB}}$ for the convex blending scheme (4.100) is not available. While the entropy production of the FV scheme is individually measured for each subcell, the entropy production of the DG scheme is defined on the whole element. Moreover, the simultaneous calculation of the entropy production with a common entropy vector $\boldsymbol{w}$ is not compatible with regards to the solution space. A naïve application of the entropy vector $\overline{\boldsymbol{w}}$ in mean value space on the RHS of the convex blending scheme leads to dubious results due to the highly nonlinear dependency of the entropy variables on the solution.

Alternatively, one might be tempted to measure the entropy production rates of both schemes separately and convex blend (CB) the results in order to get an approximation of the "real" entropy production rate:

$$\overline{\dot{\Delta\mathcal{S}}}_q^{\mathrm{CB}} \approx (1-\alpha_q)\,\overline{\dot{\Delta\mathcal{S}}}_q^{\mathrm{FV}} + \alpha_q\,\overline{\dot{\Delta\mathcal{S}}}_q^{\mathrm{DG}}, \tag{4.108}$$

where $\overline{\dot{\Delta\mathcal{S}}}_q^{\mathrm{FV}}$ is the sum of the individual cell entropy production rates (4.30), i.e.

$$\overline{\dot{\Delta\mathcal{S}}}_q^{\mathrm{FV}} = \frac{1}{N^3}\sum_{ijk=1}^{N} \overline{\dot{\Delta\mathcal{S}}}_{q,ijk}^{\mathrm{FV}}. \tag{4.109}$$

With the approximation in (4.108) we experimented with an entropy correction scheme based on blending with the entropy dissipative Minmod FV scheme. A brief discussion and some results can be found in Appendix A.1.

4.7.5 Satisfying the Element Entropy Inequality

We conjecture, when both schemes are entropy stable, i.e. $\overline{\dot{\Delta\mathcal{S}}}_q^{\mathrm{FV}} \leq 0$ and $\overline{\dot{\Delta\mathcal{S}}}_q^{\mathrm{DG}} \leq 0$, the convex blend of both RHS is also entropy stable: $\overline{\dot{\Delta\mathcal{S}}}_q^{\mathrm{CB}} \leq 0$. Hennemann et al. (2021) showed entropy stability for a variant of convex blending schemes, where the solution space for FV is equal to the Lobatto DG scheme. The proof for entropy stability for their blending operation is straightforward. However, a thorough investigation of the exact influence of our proposed convex blending operation (4.100) on the entropy balance is not available yet.

Nevertheless, we test the robustness of our blending scheme on the weakly magnetized

MHD-KHI setup (4.28) and with different combinations of Gauss DG methods. The underlying FV scheme is always the robust, second order Minmod FV scheme. Especially, we test if the indicator, discussed in Section 4.7.1, is sensitive enough to stabilize the DG scheme in critical moments, but also does not excessively trigger the FV scheme smearing out the solution.

In Figure 4.15 we plot the minimum and maximum entropy production rates of the different variants of Gauss DG. Note, we plotted the element entropy production rate $\overline{\dot{\Delta\mathcal{S}}}_q^{\mathrm{DG}}$, not the approximation (4.108) of the blended rates $\overline{\dot{\Delta\mathcal{S}}}_q^{\mathrm{CB}}$. Considering the results in Figure 4.15, we first observe that all schemes ran till final simulation time $T = 10$. Furthermore, the entropy production rate seems to indicate when a scheme is under a lot of "stress". Clearly, the spikes in entropy production of the Standard Gauss DG scheme are much more pronounced and the scheme is not capable to limit the entropy production to negative values. The other schemes in this plot, on the other hand, run much "smoother" where the EBP has a stronger influence in reducing the noise of the entropy production than the entropy correction procedure. Conclusively, we can say that the blending mechanism is capable to stabilize all investigated Gauss DG variants.

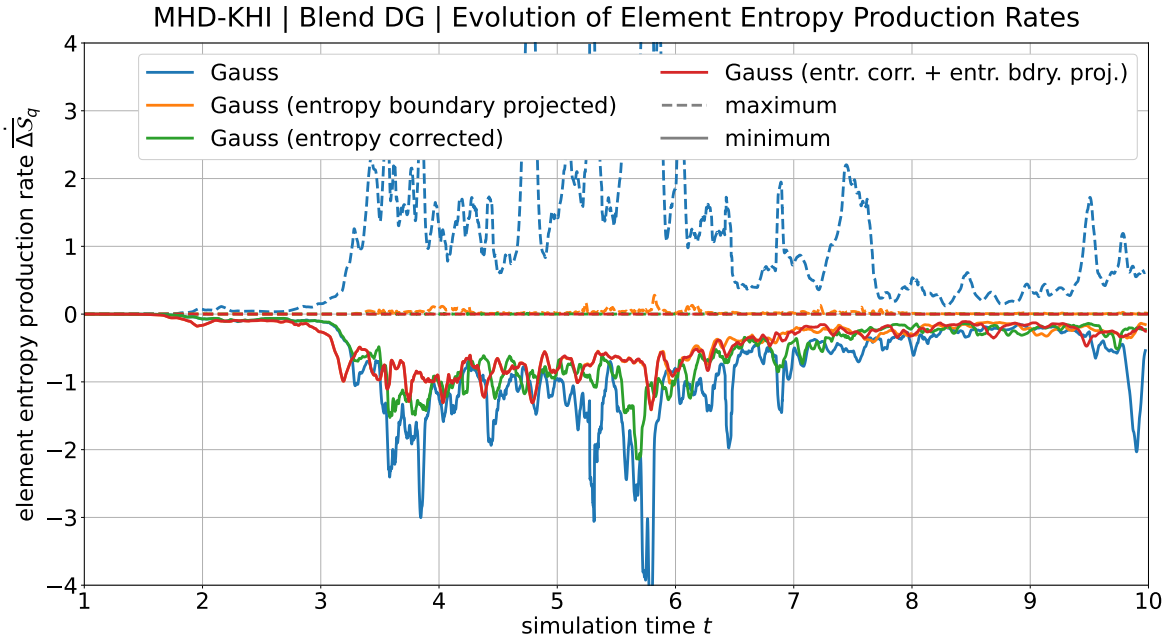

Figure 4.15: Evolution of the entropy production rates for the weakly magnetized MHD-KHI setup (4.28) and with the convex blended Gauss DG schemes.

We claimed that the entropy production rate can be interpreted as a kind of "stress"

indicator for a numerical scheme. To elucidate this insight, in Figure 4.16 we put together the results of Gauss DG and blended Gauss DG each with entropy correction and EBP, as well as the Flux Differencing Gauss DG with EBP. Clearly, the first scheme is under a lot strain, observable by the strong spikes, and eventually breaks at $t \approx 5.3$. The latter two show a similar pattern in their entropy production evolution and are robust enough to successfully finish the simulation.

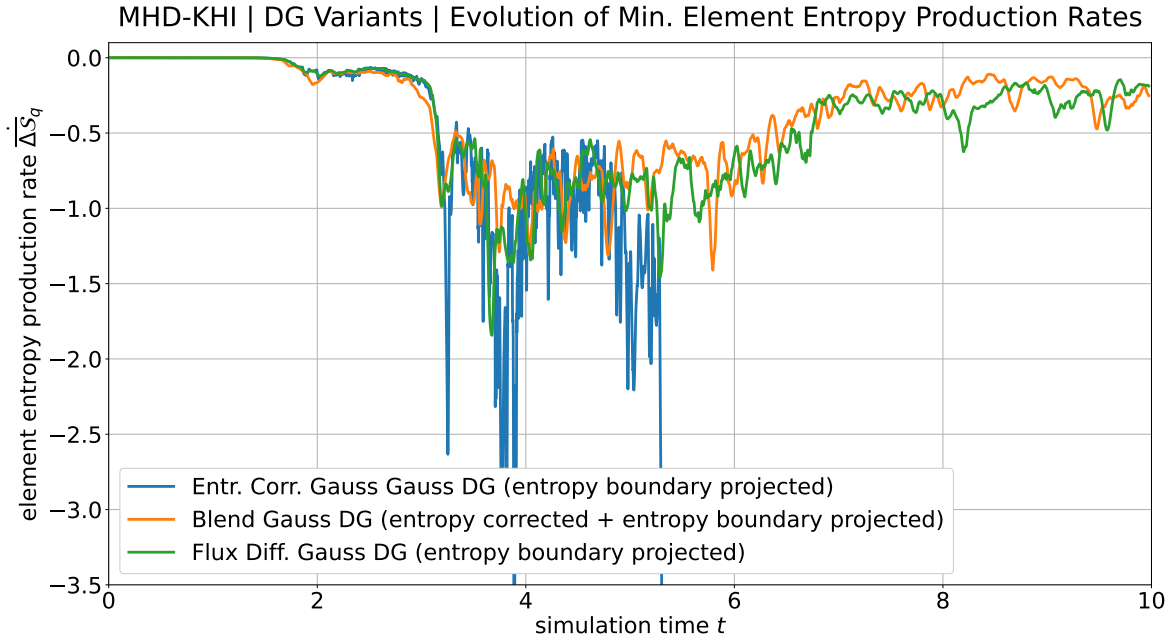

Figure 4.16: Evolution of the minimum entropy production rates for different DG variants. A noisy course with lots of high spikes indicates a troubled numerical scheme as is visible by the blue curve. The scheme eventually crashes at $t \approx 5.3$.

In Figure 4.17 we show a montage of density contour plots at $t = 7.5$ for four Gauss DG variants: Blend (Standard) Gauss DG and Blend Gauss DG with entropy correction (top row), Blend Gauss DG with entropy correction and EBP and the Flux Differencing Gauss DG with EBP (bottom row). We want to highlight two major insights coming from this plot. The first observation we make is that the solution for the Blend Gauss DG without any special entropy treatment is robust and yields acceptable results very similar to the other Gauss DG variants. The second observation concerns the solution with the Blend Gauss DG with EBP and entropy correction (bottom left). Clearly, the density solution is more dissipative than the rest.

To investigate the reason behind this observation, we additionally compare in Figure 4.18

the element-wise blending factors at $t = 7.5$ for the two entropy corrected Blend Gauss DG variants. The key difference between both schemes is the EBP, which is active only in the right plot. EBP clearly has an influence on the sensitivity of the shock indicator. Our explanation is as follows. Due to the highly nonlinear nature of the entropy variables, the resulting entropy projected polynomial tends to oscillates more, especially when the original solution is already under-resolved.

Figure 4.19 illustrates this phenomenon for a rather harmless density profile. While the reconstructed polynomial in conservative variable space (we call it standard projection in this context) is an acceptable approximation of the exact solution, the entropy projected polynomial deviates significantly producing pronounced jumps at the element boundaries. Not only do higher jumps cause an elevated blending, they can also enhance the stability of the DG scheme itself. We know that the numerical dissipation in DG is introduced by the jumps in the surface fluxes. This fact might also explain, why the Flux Differencing Gauss DG with EBP is so robust.

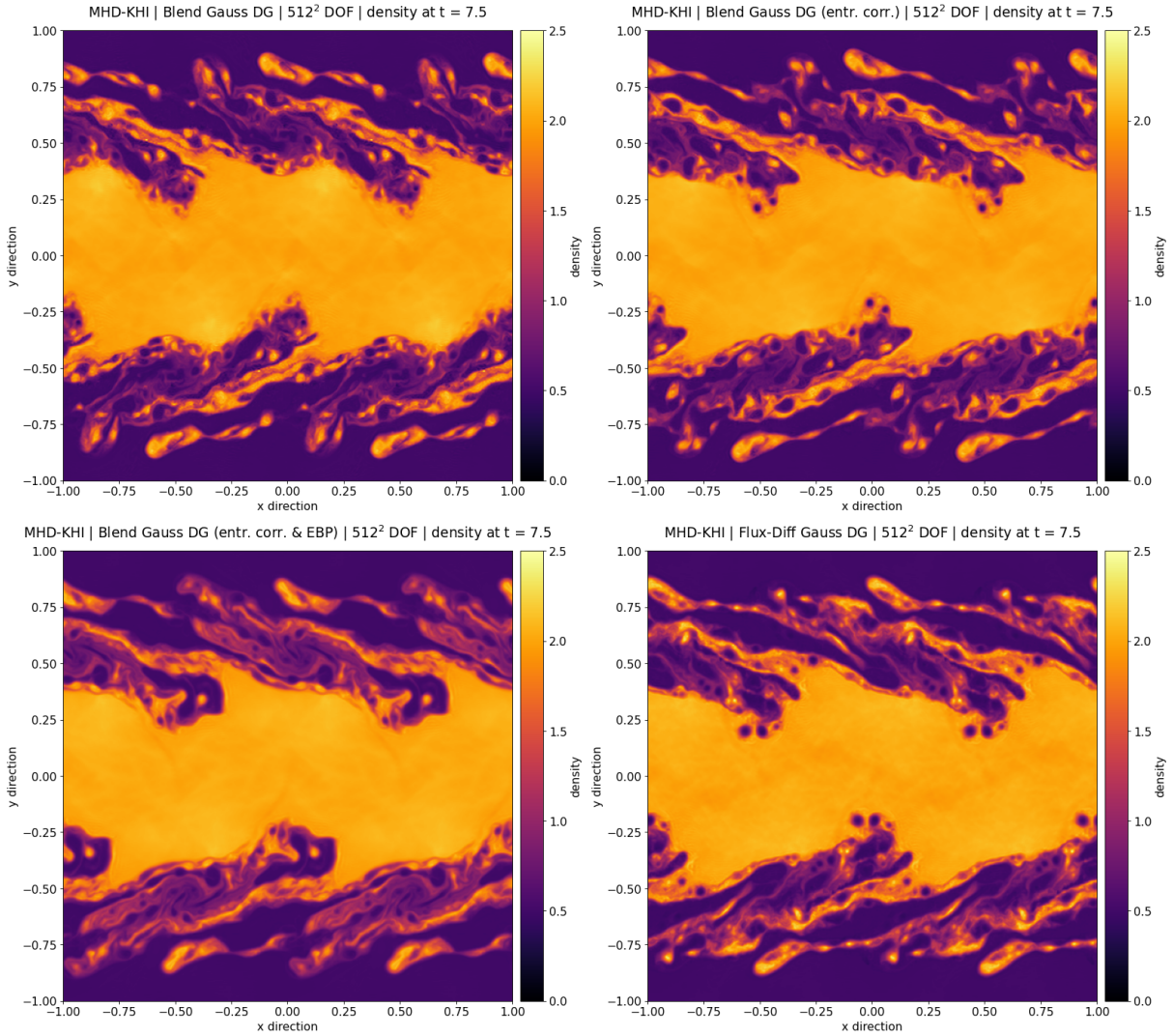

Figure 4.17: Density contours of four Gauss DG variants at time t = 7.5. Blend (Standard) Gauss DG and Blend Gauss DG with entropy correction (top row), Blend Gauss DG with entropy correction and EBP and the Flux Differencing Gauss DG with EBP (bottom row). The solution in the bottom left plot is a bit more dissipative than the rest. An explanation for this observation is given in the text.

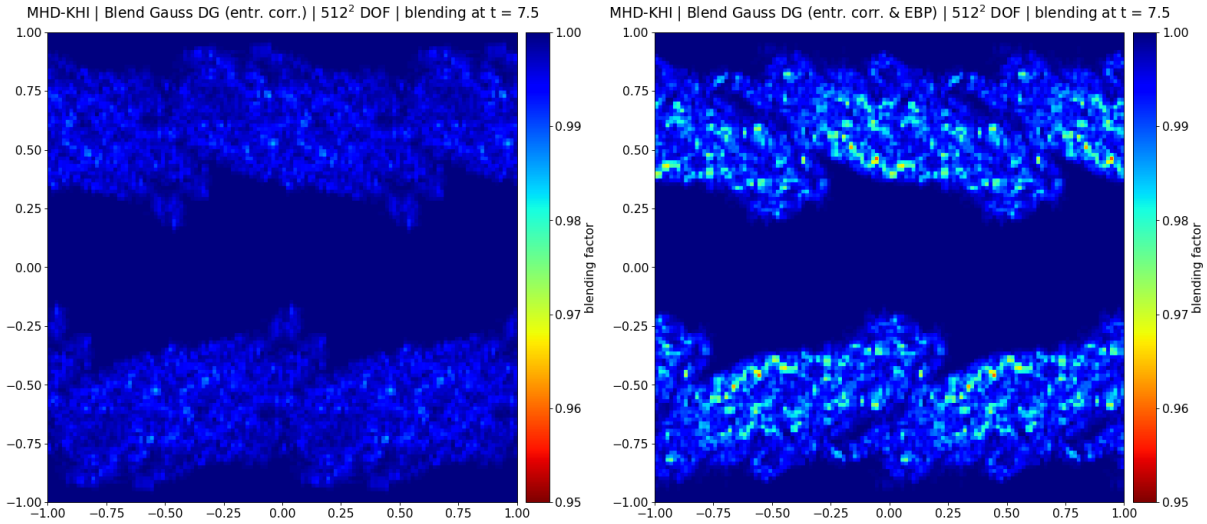

Figure 4.18: Blending factors of two DG schemes at later time t = 7.5. Blend Gauss DG with entropy correction (left) and Blend Gauss DG with entropy correction and EBP (right). Clearly, EBP increases the blending activity causing more smearing of the solution.

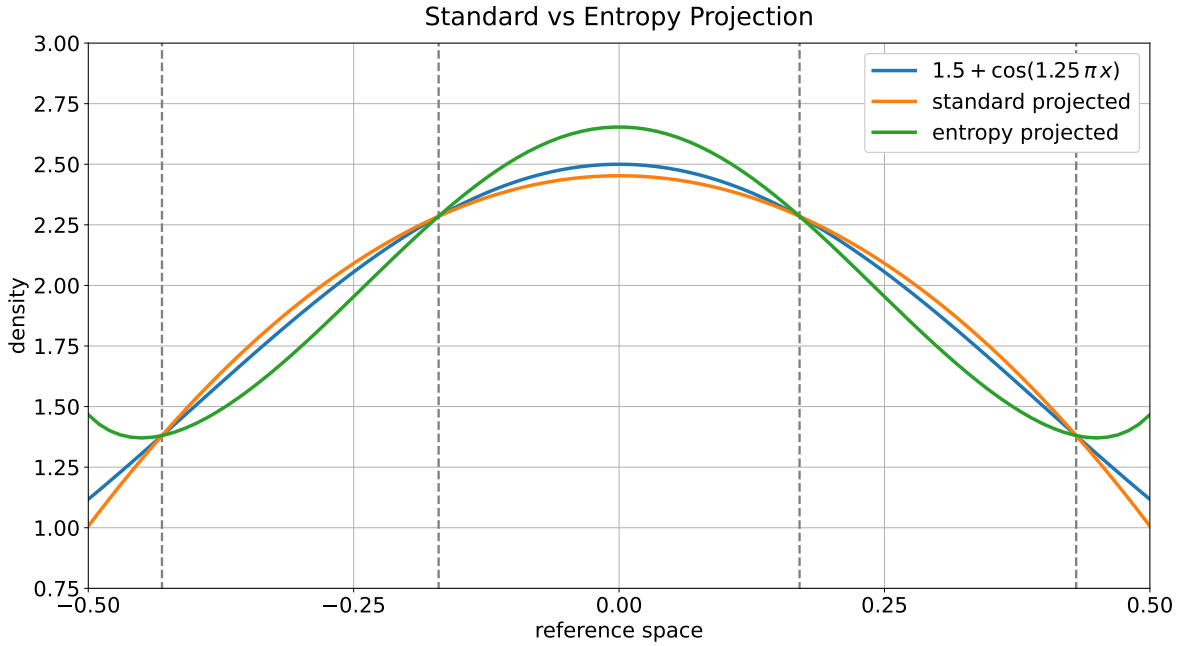

Figure 4.19: Example of the density profile by the standard projection and the entropy projection. The vertical, gray, dashed lines depict the four Gauss quadrature points ($N = 4$) within the reference element.

The results for the Lobatto DG variants are not shown, since all Lobatto DG schemes crashed at some point during the simulation due to negative densities or negative pres-

sures, even with provable entropy stability via Entropy Correction or Flux Differencing and even with the assistance of the blending mechanism. Of course, by tuning the indicator parameters, we can enforce a stable simulation for Lobatto DG, but at the expense of much diffuser results. We consider all Lobatto DG variants as not sufficiently robust and not practicable for our envisaged applications.

4.7.6 Enforcing Density and Pressure Positivity

In the conserved variables formulation (3.29), the thermal pressure (3.31) is derived from the total energy by subtracting the kinetic and magnetic energy terms. Hence, at strong shocks or under near-vacuum conditions the scheme can produce non-physical states. To alleviate this problem, we lift the troubled cells into positivity such that the permissibility condition

$$\Pi = \Big\{\text{permissible states}\Big\} = \Big\{\forall\, \boldsymbol{u} \;\Big|\; \rho > 0 \wedge p(\boldsymbol{u}) > 0\Big\}. \tag{4.110}$$

is fulfilled for all cells in a block Ω_q. First, we calculate the block average

$$\overline{\boldsymbol{u}}_q = \frac{1}{N^3} \sum_{ijk=1}^{N} \overline{\boldsymbol{u}}_{q,ijk} \tag{4.111}$$

and then determine a *'squeezing' parameter* $\beta \in [0, 1]$ which enforces physical states:

$$\overline{\boldsymbol{u}}_{q,ijk}^{\text{permissible}} = (1-\beta)\,\overline{\boldsymbol{u}}_q + \beta\,\overline{\boldsymbol{u}}_{q,ijk} \quad \in \Pi. \tag{4.112}$$

The algorithm to find a suitable β is straightforward. One starts with $\beta := 1$ and decrements in steps of $\Delta\beta := 0.1$ till the permissibility condition is fulfilled. The advantage of this algorithm lies in its simplicity and it is conservative by construction. However, it fails when the block average is not part of the permissible set after a time step respectively Runge-Kutta stage. In this case the code crashes and the simulation stops.

4.8 Final Remarks

In this chapter, we introduced the numerical scheme we are going to apply for our simulations presented in this thesis. Firstly, we introduced the basic building blocks of FV schemes, followed by describing the numerical ingredients for high order, nodal collocation DG schemes. Special focus was put on satisfying the entropy condition via two popular

approaches, namely Entropy Correction and Flux Differencing (via two-point entropy conservative flux functions in the volume terms), which we compared with respect to their robustness in simulating a weakly magnetized KHI setup. The setup starts with subsonic and smooth initial conditions, but gradually develops increasingly complex flow structures with few local transonic, highly compressive flow regions and stagnation points.

Our goal was to trigger a number of numerical instabilities, such as transonic shocklets and under-resolution (aliasing), purposely stressing our chosen DG variants. We investigated the Standard DG variants with Gauss and Lobatto quadrature rules, as well as their entropy stabilization with the two afforestation approaches. All DG variants were fixed to fourth order in spatial accuracy, our targeted order, and not "repaired" by any limiters such as lifting states a-posteriori into positivity. We observed that all DG variants with Lobatto quadrature rule are unacceptably fragile for this kind of simulations. DG with Gauss quadrature rules are noticeably more robust, but also need careful treatment to stabilize the simulation of compressive turbulent flows.

The entropy stable Flux Differencing DG with Gauss quadrature and proper EBP is the only unlimited DG variant capable to successfully finish our test simulation, which is remarkable! The Entropy Correction Gauss DG with EBP is also provably entropy stable and significantly increased the robustness in our tests. But it eventually crashed before reaching the final simulation time. One possible explanation might be the unboundedness of the algebraic correction terms adding stiff source terms to the RHS under certain flow condition. This leads to a very stiff system, whose eigenvalues lie beyond the stability region of our applied explicit Runge-Kutta schemes.

In a next step, we constructed a versatile blending scheme combining the robustness of low order FV schemes with the accuracy of higher order DG methods. This novel scheme is specifically designed for very challenging astrophysical simulations containing supersonic turbulence and very strong shocks and aims to integrate well within the multi-physics simulation framework FLASH. We also tested the robustness of our new limited DG variant with the KHI test setup and concluded that with minimal amount of focused blending any DG method with Gauss quadrature rule can be stabilized without intolerably smearing out small scale structures in the solution.

Of course, we further investigated all Gauss DG variants with other, more challenging tests such as the Sedov blast (see Section 6.3.3). We observed that the entropy consistent projection to the boundaries is ill-conditioned near shocks and under near-vacuum con-

ditions leading to severe numerical artifacts and loss of positivity in density and pressure averaged over the element. A visual explanation is given in Figure 4.19. Eventually, we conclude that the current state-of-the-art approaches in enforcing provable entropy consistency in DG and, consequently, in our blending scheme do more harm than good for our envisaged astrophysics simulation. Consequently, we rely on the Standard Gauss DG as our baseline DG for our simulations.

In Chapter 6 and Chapter 7 we show that our blending based limiter is capable to reliably stabilize the Standard DG with Gauss nodes both in strong shock conditions and for transsonic turbulent flows without smearing out the solution. Nevertheless, we also present in all honesty two simulations in Chapter 7, which push even our blending scheme to its limits, degrading the scheme basically to second order FV everywhere in the computational domain.

In the final remarks of the previous chapter (see Section 3.9) we gave a specification list a "perfect" scheme would satisfy. In Chapter 6 we show that we check the following bullet points from the list:

- High (order) accuracy in smooth, well-resolved flow regions.
- Robust and non-oscillatory handling of (very) strong shocks.
- Stable computation of under-resolved flow conditions, especially under near-vacuum conditions.
- Conservation of primary quantities (mass, linear momentum, energy).
- Handling of divergence errors in the magnetic field.
- Conservative and positivity-preserving advection of multi-species flows. Some abundances can be zero.
- In relation to a reference solver in FLASH: performant and scalable implementation allowing simulations with large dynamical ranges in space and time.

A short note on the multi-species advection is in order. We adopt the mass tracer approach given by (3.54) in Section 3.6, which we straightforwardly apply on the blended mass flux in (4.100). This method preserves the summed total density in each cell and never produces negative abundances by construction. Albeit very robust, it is, however, not high order accurate.

Chapter 5

Nemo - a modular fluid dynamics code for prototyping

5.1 Introduction

NEMO is a lightweight and modular open source simulation framework for computational fluid application. The code is written primarily in modern Fortran and aims to be accessible and encourages to tinker and add new features. NEMO implements robust, second order Finite Volume schemes and accurate, high order Discontinuous Galerkin Spectral Element methods for compressible Euler equations and ideal magneto-hydrodynamics on Cartesian meshes in 2D and 3D. Furthermore, the code offers a selection of different shock capturing methods for DG and the user can choose from a plethora of different Runge-Kutta schemes for explicit time integration.

For successful compilation a Fortran compiler, which implements the standard ISO/IEC 1539-1:2010 (informally known as Fortran 2008) or newer is necessary. A Python interpreter (version 2 or 3) and the venerable Make utility are useful for comfortably managing the build process.

The complete fluid simulation framework is open source and freely accessible under `https://github.com/jmark/nemo`. As a matter of fact, NEMO served as the workhorse for producing all numerical results in Markert et al. (2021).

Note, that the code has as similar name to NEMO Ocean (Madec et al. 2017), a modeling framework for research and forecasting in ocean and climate sciences, which is purely

coincidental.

Going beyond serial prototyping codes, NEMO utilizes two parallelization strategies, namely OpenMP and MPI, which can run individually or cooperatively depending on the scaling needs at hand. Furthermore, the framework has adaptive mesh refinement capabilities by linking to the open source package P4EST, a highly efficient and mature octree library for unstructured, dynamic meshes of quadrilaterals or hexahedras developed by Burstedde et al. (2011).

NEMO was written from scratch by the thesis' author as part of the research efforts in investigating and evaluating various stabilization strategies and shock capturing approaches for high order DG, especially with regards to the envisaged astrophysics applications. During the process of devising a robust shock capturing method for DG it was desired to have a flexible, easily modifiable, yet scalable and performant development framework for small to medium sized 2D and 3D simulations.

Checkpoints in NEMO are written as HDF5 files, which are 100% compatible to the checkpoint format used by the multi-physics framework FLASH, which enables direct transfer of simulation data across code boundaries. Moreover, the access to the huge collection of post-processing tools, established in innumerable man-years of research work, considerably eases the collaboration with the astrophysics community.

NEMO supports a number of strategies to organize the solution data, be it blocks of mean values for FV, high order elements of node values for DG or hybrids of the two. This flexibility allowed to gradually move towards a suitable scheme featuring the robustness necessary for the targeted astrophysics applications. Mirroring the block-based datastructures in FLASH streamlined the transfer of the "hardened" DG scheme from NEMO to the versatile but heavyweight multi-physics framework.

The code is structured in swappable modules each adhering to the identical internal interface guidelines. Adding new features or applying code modifications are straightforward to do by simply "dropping" new Fortran source files somewhere in the source tree. Compilation and linking is managed by lightweight Python scripts and is very fast. Even a full, parallel build on a typical multi-core workstation or a modern laptop is a matter of a few seconds. The build system is smart enough to only rebuild changed modules (and dependencies) allowing for a tight and smooth edit-compile-run development cycle.

For the rest of this chapter, we present our hardware and software stack we employed

for our simulations, discuss the most important aspects entailing a typical computational fluid dynamics code, namely adaptive mesh refinement and parallel performance on compute nodes respectively scalability in distributed computing environments and show some results.

5.2 Hardware & Compiler Stack

For development and our simulations, we had access to a performant multi-core workstation and two HPC clusters.

The workstation consists of one hexa-core CPU (Intel® Core™ model i7-8700 (3.20 GHz) with optional hyperthreading (12 threads). The workstation served as the primary development environment, data analytics and plotting platform. The runtime performance results on a single core shown in Table 5.1 have been retrieved on this machine. Gfortran (GCC version 11.1.0) with OpenMPI (version 4.1.1-3) was the standard Fortran compiler on the workstation.

The scaling tests and simulations were performed on the Cologne High Efficiency Operating Platform for Sciences (CHEOPS) cluster and on the group cluster ODIN, both hosted by the Regionales Rechenzentrum Köln (RRZK).

CHEOPS consists of over 800 compute nodes interconnected via Infiniband Quadruple Data Rate (QDR) network with a brutto bandwidth of 40 Gb/s according to the operator's specifications. We accessed a partition, which unites four compute nodes each equipped with two Intel® Xeon™ processors model E5-2680 v3 (2.5 GHz) á 12 cores.

The ODIN cluster has over 500 compute nodes connected via an Infiniband network. The partition, which we used for our runs, consists of 54 nodes each equipped with two Intel® Xeon™ processors model E5-2670 (2.6 GHz) á 8 cores.

Among many other software packages, ifort (Intel® compiler suite 18.0) with Intel® MPI (version 18.0) were installed on both clusters and it has proven to be the most reliable compiler stack with regards to avoiding crashes due to non self-inflicted segmentation faults or sudden interruptions of the MPI data exchange.

Remark. The same hardware and compiler details apply to the FLASH code results discussed in Chapter 6 and in Chapter 7.

5.3 Adaptive Mesh Refinement

In CFD, adaptive mesh refinement (AMR) is a popular and very successful method of dynamically adapting the resolution, and thus the accuracy, of flows within certain sensitive or turbulent regions. Uniform Cartesian grids are limited to a fixed precision and for 3D simulation a doubling in grid resolution would entail an eight-fold increase in memory consumption. From Section 3.7 and Section 3.8 we learned that there a problems in astrophysics which require a level of resolution simply not feasible with uniform grids and with today's hardware. Fortunately, such simulations usually do not require a uniform spatial precision and can profit from substantial speed gains and data storage savings if only specific areas of the domain are fully refined.

Refinement & Coarsening

The design of "good" refinement rules on where to refine the solution for nonlinear hyperbolic systems is rather a form of art then hard science and is closely related to the issue of shock capturing methods for high order methods. In this thesis, we rely on two methods. The first approach is a solution-independent, "static" selection rule in cases where flow regions of specific interest are already known beforehand. This method is very reliable and gives predictable runtime and memory consumption estimates. It is, of course, not generalizable to a broad range of problems. A plausible example for such a use case is the young supernova remnant simulation described in detail in Section 7.2. The second approach is to devise an estimator, which tries to assess if the solution is still properly resolved by the given level of refinement and adjusts the grid resolution accordingly. A robust and time-proven AMR indicator was invented by Löhner (1987). The estimator was originally developed for finite element applications and has the advantage that it is local. It is dimensionless, bounded between $[0, 1]$, and can be applied with complete generality to FV and DG discretizations in multiple dimensions and on any arbitrary shaped elements. The general, multi-dimensional form for element Ω_q and variable u is given by

$$L_q(u) = \left(\frac{\sum_{k,l=1}^{3} \left(\int_{\Omega_q} \left(\partial_{x_l} \partial_{x_k} u(\vec{x}) \right) \mathrm{d}|\Omega_q| \right)^2}{\sum_{k,l=1}^{3} \left(\int_{\Omega_q} \partial_{x_l} \left(|\partial_{x_k} u(\vec{x})| + \epsilon \, \partial_{x_k} |u(\vec{x})| \right) \mathrm{d}|\Omega_q| \right)^2} \right)^{\frac{1}{2}} . \tag{5.1}$$

The terms following ϵ function as a "noise" filter in order not to refine "wiggles" or "ripples" in the solution, which may appear due to loss of monotonicity. This is an

important feature, especially for high order methods. It is easy to see that with our tensor product ansatz (4.48) for DG and collocation of interpolation and quadrature, a construction of the estimator (5.1) for high order DG elements is straightforward. For FV schemes, a simple yet robust second order Finite Difference method is usually the means of choice. For example, given three discrete mean values $\overline{u}_{i-1}, \overline{u}_i, \overline{u}_{i+1}$, the specific 1D version of the estimator reads

$$\overline{L}_i = \frac{|\overline{u}_{i+1} - 2\,\overline{u}_i - \overline{u}_{i-1}|}{|\overline{u}_{i+1} - \overline{u}_i| + |\overline{u}_i - \overline{u}_{i-1}| + \epsilon\,(|\overline{u}_{i+1}| + 2\,|\overline{u}_i| + |\overline{u}_{i-1}|)}, \tag{5.2}$$

which basically is a modified second derivative normalized by the average of the gradient over one FV cell. Within a block Ω_q of mean values the maximum value of $\overline{L}_i$ over all cells defines L_q. Since our convex blending scheme (4.100) is also constructed on mean values, we prefer the second order Finite Difference ansatz over the construction of a high order DG version. Moreover, the estimator on mean values (5.2) is the default AMR refinement rule in FLASH and it is usually applied on the density and pressure variables. Löhner (1987) suggests to refine, when $L_q(u) > 0.3$ and to coarsen when $L_q(u) < 0.1$. The constant ϵ is usually chosen to be 0.2. In our tests, the proposed parameters work just fine.

A detailed account on refinement and coarsening algorithms as well as surface flux exchange based on blocks of mean values at different resolution levels, which also maintain mass and energy conservation and preserve high order of accuracy, is given in Markert et al. (2021).

Octree-based Adaptive Meshes with p4est

NEMO delegates the construction and management of the dynamic mesh to P4EST, which organizes the elements at different refinement levels into a logical, distributed octree. P4EST is designed to work in parallel and scales to hundreds of thousands, even millions, of processor cores (Burstedde et al. 2011; Isaac et al. 2015). The library is actively maintained, supports dynamic load balancing (partitioning), is used already in many research projects, e.g., Burstedde et al. (2014); Bangerth et al. (2011); Isaac and Knepley, and extensions to general polyhedras exists (Burstedde and Holke 2016, 2017; Holke et al. 2021). For memory efficiency and compact data structuring, only the leaf nodes (elements containing the solution data) are strung together by a space filling curve in Morton (1966) ordering, also known as Z-curve and linearly stored in memory. Morton ordering

compactly maps multidimensional data to one dimension while preserving locality of the data, i.e. neighboring elements are still positioned closely together. A 2D mesh with different refinement levels and its overlayed Z-curve is shown in Figure 5.1.

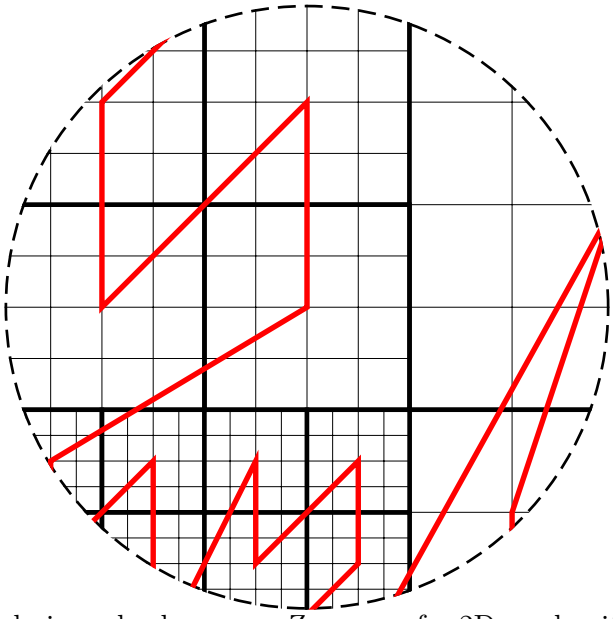

Figure 5.1: Morton ordering, also known as Z-curve, of a 2D mesh with elements on different refinement levels.

Since P4EST is written in C, NEMO must interface with the library via the ISO_C_BINDING feature introduced in ISO/IEC 1539-1:2004 (Fortran 2003). P4EST maintains its own private data structures and information about the elements such as location, refinement level and connectivity to neighboring elements must be queried through its public API. The API also readily provides routines for data exchange at ghost layers in case of distributed computing as well as balancing of computational load among MPI processes. NEMO maintains its own copy of the connectivity data for efficiency and only retrieves an update after refinements, coarsenings or repartitionings of the mesh. The solution data is not controlled by P4EST and selective access necessary for MPI data exchange is granted by passing C-compatible pointers to separate pre-allocated data buffers. The interface between NEMO and P4EST is developed and maintained by NEMO's authors as an independent package and can be freely accessed under https://github.com/jmark/p4wrap.

5.4 Hybrid Parallelization

Parallel computing is a type of computation in which many calculations or tasks are carried out simultaneously. Large problems, as for example is often the case in CFD, can often be divided into smaller ones, which can then be solved in parallel. Parallelization of computing tasks has long been employed in high-performance computing, but has gained broader interest due to the physical constraints preventing frequency scaling, i.e. making processors run "faster". Leveraging the performance benefits of multi-core processors and inter-connected clusters of computers, NEMO supports two parallel programming standards, OpenMP and MPI.

OpenMP stands for "Open Multi-Processing" and is an API that supports multi-platform, shared-memory, multiprocessing programming in C, C++, and Fortran. The open source specification is devised by the nonprofit OpenMP Architecture Review Board and developed and distributed as optional extension by all major compiler vendors. The specification uses a portable, scalable model that gives programmers a simple and flexible interface for developing parallel applications for computing platforms ranging from standard desktop computers to supercomputers. OpenMP-aware compilers offer convenient access to multithreading, a method of parallelization whereby a primary thread forks a specified number of sub-threads. In computing, a thread is defined to be a series of instructions or tasks executed consecutively. Work load is then evenly distributed among the threads running in parallel, with the runtime environment allocating threads to different cores in case of multi-processor hardware. Since all threads share the same hardware memory, all threads in the same thread pool have direct and fast read/write access to shared data.

MPI stands for "Message Passing Interface" and is a portable open source message-passing standard designed to function on distributed computing architectures, such as HPC clusters. The MPI standard defines the API exposing library routines that are useful for writing portable programs in C, C++, and Fortran sharing arbitrary data over computer networks. Both point-to-point and collective communication are supported. In contrast to OpenMP, MPI implementations are usually standalone shared libraries, which are dynamically linked into the application. MPI is not specified by any major standards body; nevertheless, it has become a de facto standard for message-based communication among processes that model a parallel program running on a distributed collection of network-connected compute nodes. In computing, a process is the instance of a computer program that is being executed. For maximum performance, each processor (or core in a multi-core

machine) will be assigned just a single process.

An application, such as NEMO, built with a hybrid model of parallel programming can run on a computer cluster using both OpenMP and MPI, such that OpenMP is used for parallelism within a (multi-core) node while MPI is used for parallelism between nodes. On each node runs one MPI process, which in turn manages a pool of OpenMP threads matching the number of available cores. Figure 5.2 sketches a distributed, intra-node shared memory configuration on a small model cluster with four connected compute nodes each having four cores.

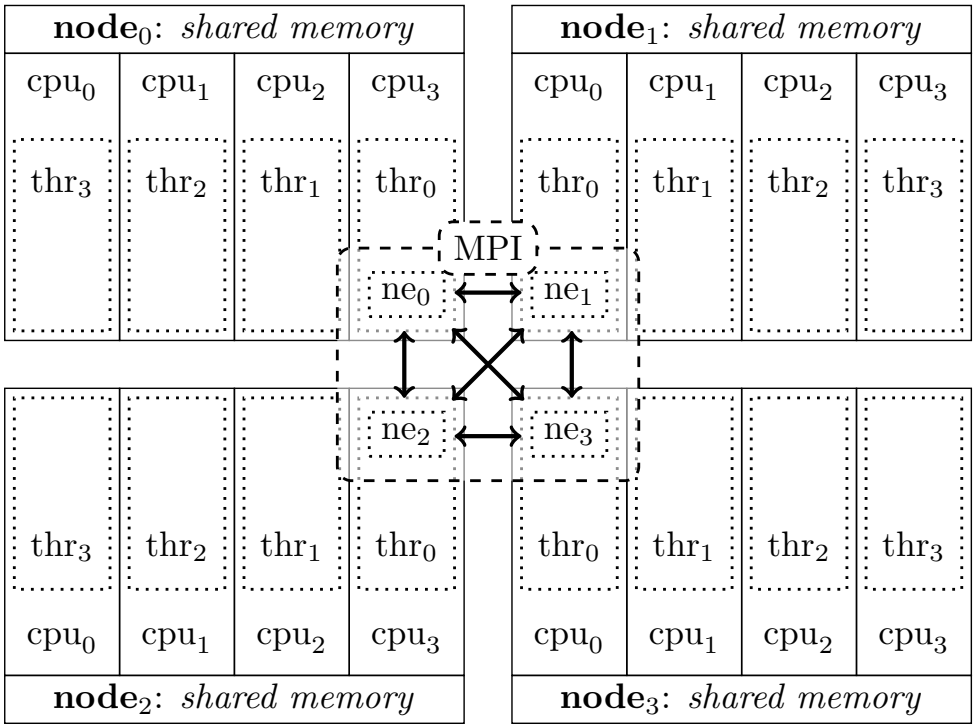

Figure 5.2: Example of a small cluster with four compute nodes each equipped with four cores. In computing, it is customary to count instances starting with 0. On each node one MPI process respective NEMO instance (ne_i) manages four OpenMP threads (th_i) each alloted to one core. Threads within a common thread pool can exchange data via shared memory, while data exchange between nodes is managed by the NEMO instances via the MPI message bus (dashed rectangle in the center).

In CFD, the typical MPI parallelization approach is to decompose the computational domain into non-overlapping MPI subdomains or zones matching the number of allotted processing cores or compute nodes (in case of hybrid parallelization). Based on the connectivity to neighboring zones each process allocates a ghost layer, which is then filled with data from other processes. The data exchange is carried out exclusively over the

MPI message bus. An example for the zoning of the computational domain is shown in Figure 5.3. The figure shows a 2D simulation with NEMO of two supersonic jets of gas (left plot) moving in opposite directions and getting diverted by a fictional black hole at the center. The setup is parallelized with six zones encoded as six different colors in the right plot. Obviously, AMR was already triggered at this stage in the simulation since the center and the infalling gas streams are finely resolved by the quadtree. Note, that some zones cover smaller portions of the domain, while the number of elements is almost equal among the zones. This is due to the load balancing feature provided by P4EST, which helps to keep the computational work balanced among all processors maximizing the utilization of the full processing capacity.

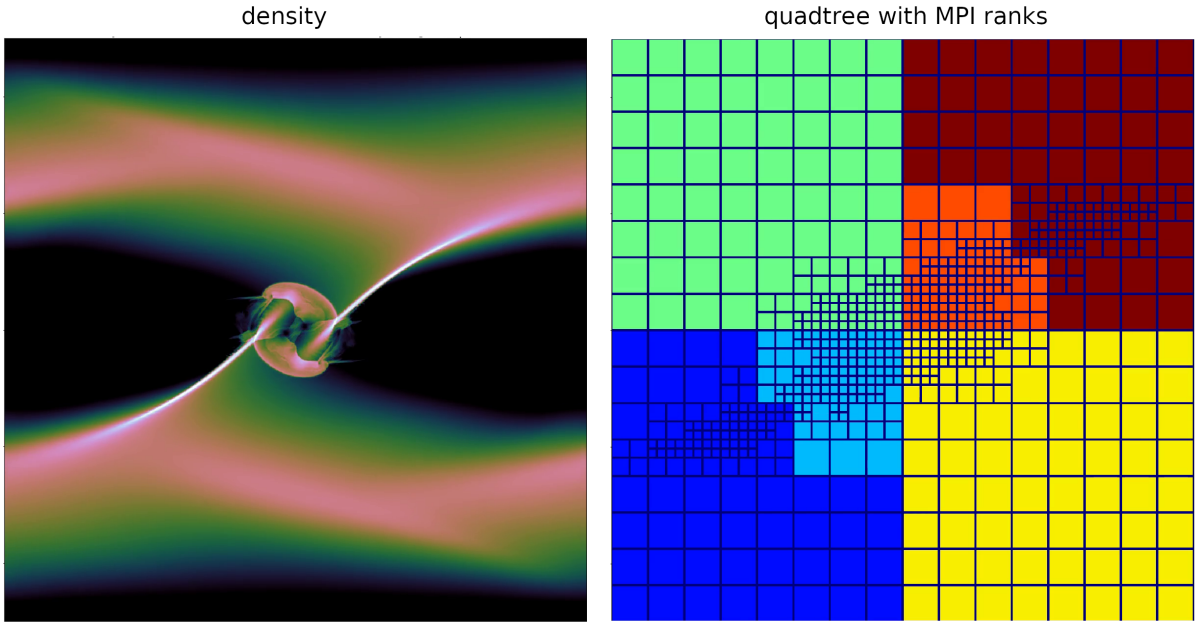

Figure 5.3: 2D simulation with NEMO of two supersonic jets of gas (left plot) moving in opposite directions and getting diverted by a fictional black hole at the center. The right plot visualizes the mesh with different resolution levels managed by the quadtree library P4EST with six color-encoded parallel zones. In MPI jargon, a rank is the number or identifier (id) of a MPI process. The dark blue grid lines highlight the element boundaries. Not explicitly shown are the ghost layers clinging along the fault lines of adjacent zones.

One might expect to get an S times speedup when running a program parallelized using OpenMP on S processors or MPI on S nodes. However, this ideal case is almost never achieved for computations with data dependency on other computing units. In general, a thread or process must wait until the data it depends on is computed and made accessible. This inevitable synchronization step prevents an application to become 100% parallelized,

which means that the theoretical upper threshold of maximum speedup is limited. We will revisit this aspect of parallelization again in the discussion about the scaling performance of NEMO.

5.5 Runtime Performance & Scaling Behavior

Performance of a program is not a clearly defined term and can be, among many others, refer to aspects of accuracy, robustness (or fault tolerance), throughput, latency, and efficiency. However, these aspects also have different meanings in different contexts. Accuracy in CFD is mostly affected by the chosen numerical method and the maximal achievable resolution limited by hardware constraints. Robustness could either refer to the numerical scheme, being capable of dealing with challenging flow states, e.g. shocks, or it could refer to a MPI-parallelized program dealing with, e.g., unresponsive compute nodes or read/write errors in storage systems. Scientific simulation codes that are robust (or fault tolerant) will get increasing attention in the advent of exascale computing with millions of individual computing units where the chances of a malfunctioning node during a large scale simulation is not unlikely due to the shear size of the cluster (Dongarra et al. 2015). Throughput, in general, is a measure of how many units of information a computing system can process in a given amount of time, while latency represents the time delay it takes for messages to get to its destination across, for example, a computer network. The overall performance of a computing cluster is determined by both, throughput and latency. Raw processing throughput can be directly doubled by simply doubling the number of computing units. Latency, on the other hand, is a consequence of signal speed limits in the transmission medium and of the inherently serial process of marshaling the fragmentented data packages. Applications in HPC are considered efficient when they exploit the maximum capacity of available processing throughput by avoiding idle periods due to time delays in the data transfer as much as possible. The process of offering more and more processing power to simulation software, which is capable to leverage the additional resources is called scaling. Note that the influence of latency is not limited to computer networks but, also is (on a much smaller timescale) significant for shared memory systems. Typical round trip delays of a (local) network are measured in $\sim 100\,\mu$s (10^{-4} s) while the access to main memory on modern hardware is of the order ~ 100 ns (10^{-7}s).

By using the example of our prototype code NEMO, we introduce two performance metrics

to gain basic insights into how our implementations of the numerical schemes, discussed in Section 4.6, perform in relation to each other.

Throughput on a Single Core

The first performance metric, we are interested in, is the aforementioned throughput (TP), which quantifies how "fast" a specific implementation of a numerical scheme is. Here, we define TP to be the amount of DOF a solver is capable to process within a given time on one computing unit, resp. core. The formula of TP reads

$$\mathrm{TP} = \frac{\#\mathrm{steps} \times \mathrm{DOF}}{\#\mathrm{cores} \times \mathrm{runtime}}. \tag{5.3}$$

We measure the TP by letting the code run till a fixed final simulation time for a fixed amount of DOF and a fixed number of computing units (#cores) alloted for the computation. Afterwards the total runtime is recorded, whereby contributions of code initialization and read/write operations have to be factored in. The number #steps amounts to the number of Runge-Kutta cycles times the number of Runge-Kutta stages.

Of course, TP does not only depend on the algorithmic efficiency of the numerical scheme, but also on the utilized hardware and the compiler stack. A brand-new processor of latest generation with high clock speeds and specialized instruction sets combined with highly optimizing modern compilers easily outperform older machines with slower hardware and antiquated software. The same applies to HPC clusters regarding the specifications of deployed compute nodes and the data transfer characteristics of the network. As long as we obtain TP results on the same machine, with the same compiler, and preferably with the same simulation framework, comparative studies regarding the runtime performance of various solver implementations can be considered accountable.

We want to stress that in this work we will not present full-blown benchmark analysis of our codes backed by solid profiling statistics and obtained under a broad range of work load scenarios. Instead, we focus on a couple of individual runtime performance results, which collectively reveal a trend with regards to the overall runtime costs our implementation of the proposed convex blending scheme, introduced in Section 4.7, entails.

Table 5.1 summarizes the TP results of the six fourth order DG variants, detailed in Section 4.6 and implemented in NEMO. The numerical setup was chosen to be the weakly magnetized MHD-KHI setup (4.28) run on a uniform grid of 256^2 DOF and on a single

core of the workstation till final simulation time $T = 2.5$. All simulations consumed a total number of 1263 RK cycles times 5 stages with the LS-RK(5,4) scheme.

Table 5.1: Runtimes of the six DG variants, detailed in Section 4.6, for the 2D MHD-KHI setup till $T = 2.5$ and on a uniform grid of 256^2 DOF run by NEMO on a single core (#core = 1) of the workstation. All simulations had a total number of iteration steps of #steps = 1263×5 (LS-RK(5,4)).

scheme	runtime [s]	TP [10^6 DOF/s]	slowdown
Std. DG (Lobatto)	72.652	5.696	1.00
Std. DG (Gauss)	74.240	5.574	1.02
Std. DG (Gauss + EBP)	100.143	4.132	1.38
Entr. Corr. DG (Lobatto)	108.929	3.799	1.50
Entr. Corr. DG (Gauss)	111.564	3.709	1.54
Entr. Corr. DG (Gauss + EBP)	131.164	3.155	1.80
Flux Diff. DG (Lobatto)	209.307	1.977	2.88
Flux Diff. DG (Gauss)	408.687	1.012	5.63
Flux Diff. DG (Gauss + EBP)	435.082	0.951	6.00

Our baseline in Table 5.1 is also the fastest scheme, namely the Standard DG with Lobatto quadrature, closely followed by the Standard DG with Gauss points. The costs of Entropy Correction amounts to a reasonable increase of up to 80% in runtime while Flux Differencing is a couple of times more expensive than the baseline scheme. The biggest contribution to the runtime costs for the Flux Differencing schemes are the repeated evaluations of the rather expensive logarithmic means in the entropy conservative Riemann flux, especially for the Gauss variants with its tight coupling of the surface fluxes with the inner states. The results in Table 5.1 are not far off from observations discussed in the literature. Chan (2018) gives a breakdown in algorithmic complexity of the Flux Differencing schemes for Lobatto and Gauss quadratures and arrives at the conclusion that for fourth order Flux Differencing DG, Gauss is roughly twice as expensive as Lobatto. Ranocha et al. (2021) presents a multitude of optimizations in order to bring down the costs of the expensive volume flux evaluations. Of course, low hanging fruits, such as reducing the total number of flux computations by exploiting the symmetries in the evaluation of the volume terms, have already been realized. Moreover, we note that volume flux evaluations impact the runtime costs differently depending on the specific implementation and targeted computational architecture. For example, while flux evaluations typically dominate runtimes for serial implementations targeting CPUs at all

approximation orders, they do not contribute significantly to runtimes at orders one to eight for implementations on Graphics Processing Units (GPUs) as was shown by Wintermeyer et al. (2018). This effect can be expected to become increasingly pronounced in the future due to the perpetually widening processor - memory performance gap (Mahapatra and Venkatrao 1999; Kadaifçiler 2017; Efnusheva et al. 2017). In Figure 5.4 we see that in the past decades CPU/GPU processing speeds grew by 50% to 60% each year while memory access rates only improved 9% annually. Although processing speeds reached a plateau from 2005 onward, upcoming innovations in tightly coupled, highly performant, multi-purpose systems-on-chip platforms, such as the ARM-based M1 chip from Apple Inc., might rekindle the speed race against shared memory data bus on multi-socket machines in the foreseeing future. Consequently, on such computing platforms costly but CPU-bound numerical schemes, such as the entropy stable Flux Diff. Gauss DG, might not be at disadvantage with regards to "raw" throughput anymore.

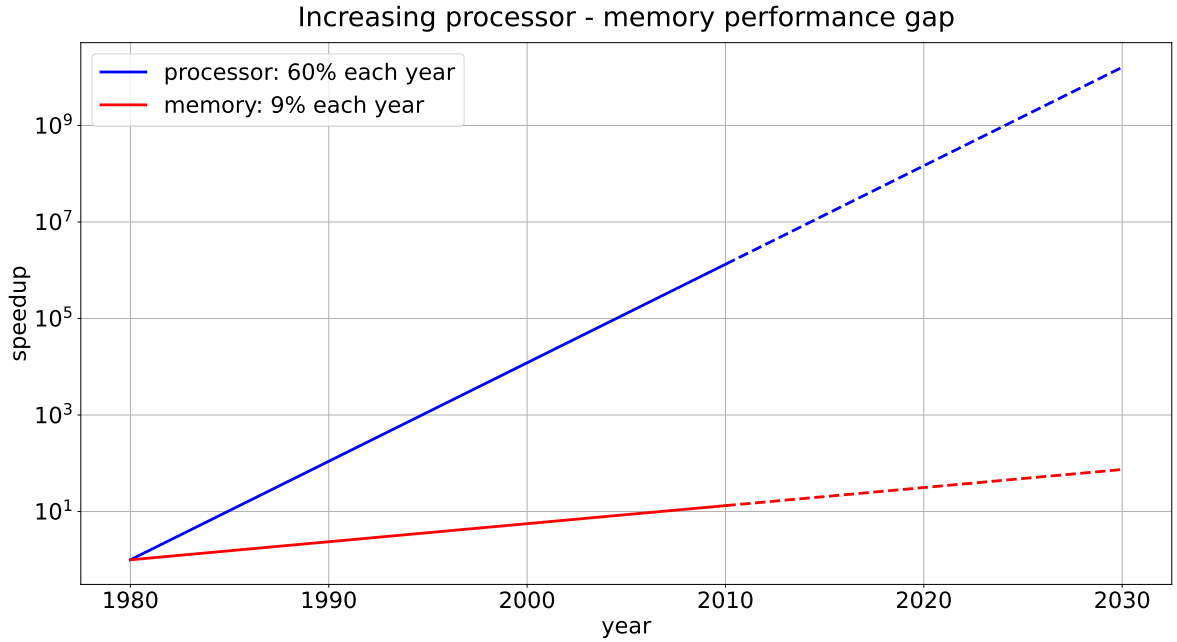

Figure 5.4: Sketch of the processor - memory performance gap over the years since 1980. Dashed lines are rough extrapolation of performance gains for upcoming next generation hardware.

Scaling on a Single Compute Node

Scalability of a parallelized program is a direct consequence of the program's efficiency. An efficient program can in general profit from the multiplication in speed via parallel

data processing on multi-core machines or clusters of compute nodes. In HPC, there are two metrics to characterize the scalability of an application, namely weak and strong scaling. The first metric quantifies how the total execution time varies with the number of processing units for a fixed problem size per unit. The latter is defined as how the total runtime changes with the number of processing units for a fixed total problem size. For our fluid simulations, we mostly want to run a simulation at a preset maximum resolution and want to know if we can speed up the simulation by "throwing" more hardware at the problem. Hence, we focus on the strong scaling properties of our codes.

In computing, there are so-called compute bound and memory bound problems. The completion time for compute bound tasks is determined principally by the speed of the processor. The complete data set for purely compute bound algorithms fits within the registers of the CPU and do not depend on the results from external sources. However, most real-world programs do not completely fit inside a CPU and also must share data with other computing resources. Data must be shoveled through the memory bus, which is slow compared to the processing speeds of modern CPUs. Instructions on CPUs with clock speeds of multiple GHz are processed in fractions of nanoseconds (10^{-10}s). If, however, the memory bus is saturated, the problem becomes memory bound. The CPU must wait for fresh data to arrive and cannot run at full capacity. Simple bound and bottleneck analysis is theoretically founded on the roofline model (Williams et al. 2009) and is well understood. Nowadays, hierarchies of caches (L1, L2, L3, etc.) with much faster access rates sit between CPU and main memory greatly improving the situation. Still, the memory bottleneck cannot be completely eradicated and will always degrade the scalability of programs on shared memory systems.

The theoretically expected speedup S of a parallelized program for fixed problem size can be adequately modeled with the following law introduced by Amdahl (1967). It reads

$$S(\#\text{cores}) = \frac{1}{(1-p) + p/\#\text{cores}} \tag{5.4}$$

with $p \in [0, 1]$ being the portion of execution time that runs in parallel. Obviously, the ideal case of a perfectly parallel application ($p = 1$) gives a linear relation between speedup and number of cores.

In Figure 5.5 we show the strong scaling of NEMO on a single compute node on CHEOPS tested with two DG variants, Standard DG and Flux Diff. DG in 2D and 3D (both

with Gauss nodes). A compute node on CHEOPS offers two processors each equipped with 12 cores. The parallelization method is pure OpenMP. The bad scaling of the 2D Standard DG scheme is not surprising, since the ratio of raw computation to data transfer is biased towards the latter. Curiously, the 2D Flux Diff. DG has such high demands on computation that it shows similar scaling to the 3D Standard DG scheme. The result for 3D Flux Diff. DG nearly follows the ideal scaling in this specific performance test. Standard DG and Flux Diff. DG transfer an equal amount of surface data, thus increased demand on CPU bound computations improves scaling. The results of all schemes nicely follow Ahmdal's law (5.4) on the first CPU, but clearly break away as soon as the second CPU is taking part in the computation. Apparently, the memory bus between CPU sockets has a different scaling dynamic than the memory lanes between cores on the same socket. For workloads with numerical data in 3D this effect is minor and still gives satisfying scaling results on multi-socket machines.

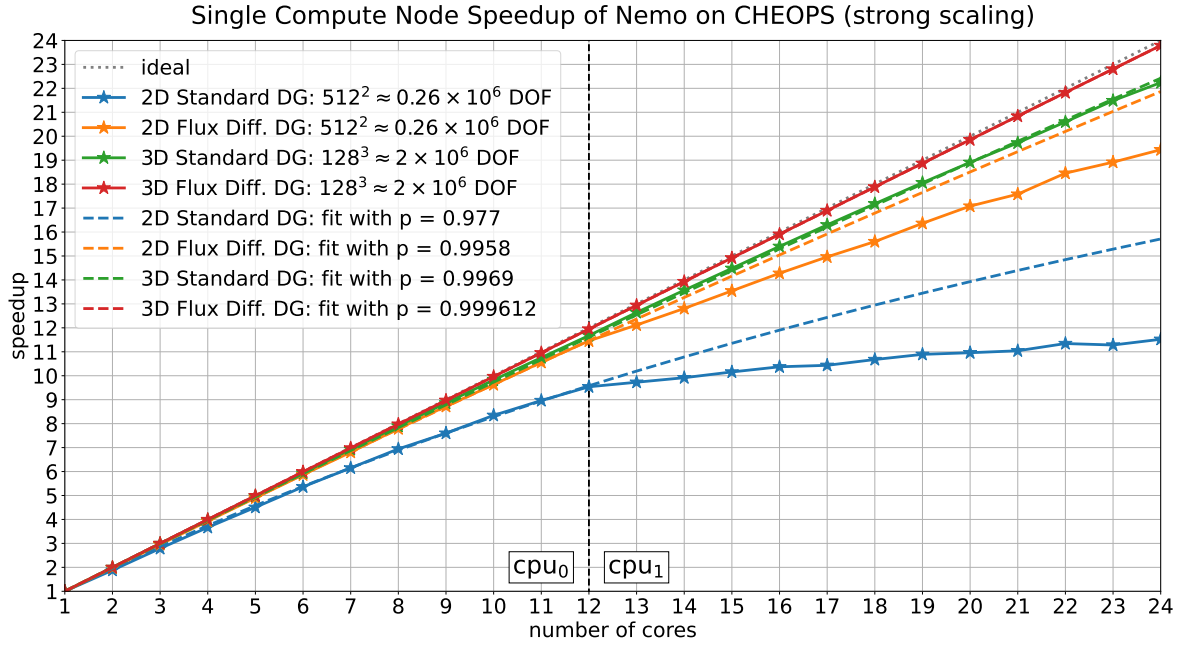

Figure 5.5: Speedup of NEMO on a single compute node on CHEOPS equipped with two CPUs á 12 cores. The parallelization method is pure OpenMP. Shown are the results (solid lines) for Standard DG and Flux Diff. DG (Gauss nodes) in 2D and 3D. The fits (dashed lines) with Ahmdal's law (5.4) were calculated only with data points from the first CPU. The course of the fits from 12 to 24 cores should be interpreted as extrapolations. The deviation of the measured results from the model in the second half of the plot are discussed in the text.

Scaling on Multiple Compute Nodes

Besides scaling on multi-core systems, NEMO is also capable to leverage distributed computing offered by HPC clusters. Information exchange between compute nodes is based on MPI, while intra-node parallelization is realized with threads via OpenMP. The scaling results with Standard DG (Gauss) in 3D for a fixed problem size of 256^3 DOF and two different clusters, namely CHEOPS and ODIN, are shown in Figure 5.6. NEMO scales better on CHEOPS than on ODIN. Clearly, running on CHEOPS has the advantage of leveraging bigger compute nodes with more cores. This works in favor for scaling. Curiously, on ODIN the scaling performance slides away from the theoretical expectation (fitted curve) for nodes 11, 13, 14, but comes back when the number of nodes is increased further. In fact, this behavior on ODIN can be observed with other codes and scaling tests as well and seems not related to NEMO. It should be mentioned, that ODIN was decommissioned by the first owners and got a "second" life as "playground" for code development and small to middle sized simulations. Thus, not much effort has been recently invested into tuning and maintaining the network infrastructure of the cluster.

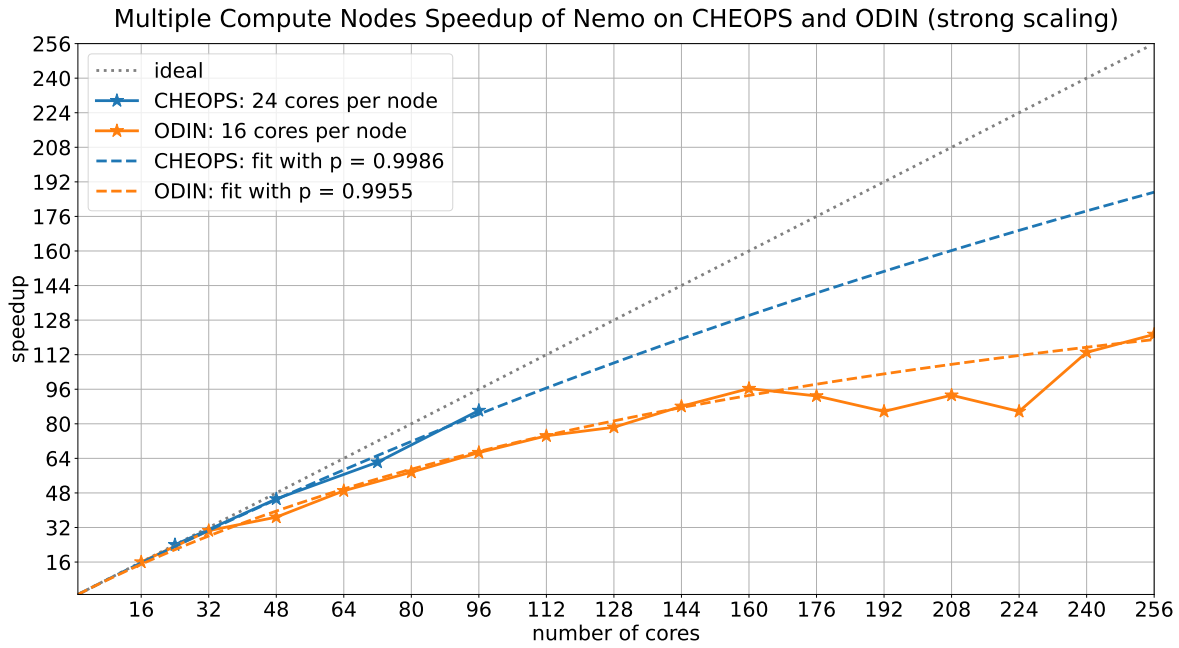

Figure 5.6: Speedup of NEMO on multiple compute nodes on the clusters CHEOPS and ODIN. Shown are the results (solid lines) with Standard DG (Gauss nodes) in 3D on a uniform grid of 256^3 DOF. The dashed lines are fits with Ahmdal's law (5.4). The fits were calculated from data points on CHEOPS for compute nodes 1 to 4 and on ODIN for nodes 1 to 10. The course of the fits beyond the fitting ranges should be interpreted as extrapolations. The dip in scaling on ODIN for nodes 11, 13 and 14 is discussed in the text.

Nevertheless, the scaling results show that NEMO is suitable for small to medium size simulations with reasonable speedup on multi-core machines and on small HPC clusters. Again, we want to emphasize, that the codes produced over the course of this work are still considered prototypes with lots of room for improvements regarding performance optimizations. Moreover, NEMO's ghost layer implementations is at disadvantage compared to optimized DG codes like FLEXI (Hindenlang et al. 2012; Krais et al. 2020), FLUXO (Gassner et al. 2016c; Rueda-Ramírez et al. 2021) or TRIXI.JL (Schlottke-Lakemper et al. 2021; Ranocha et al. 2022). NEMO offers access to the complete set of volume data for ghost elements, and thus has the needed flexibility for rapid prototyping of numerical schemes at the cost of transferring large chunks of volume data. Optimized DG codes, on the other hand, only transfer surface data, drastically reducing the pressure on the MPI message bus.

Chapter 6

Implementing DG in FLASH

6.1 Introduction

In this chapter, we present some context, technical details, and numerical test cases of our DG implementation in the astrophysics, multi-physics simulation framework FLASH. A large share of the content in this chapter has also been published in Markert et al. (2022). Here, we expand on the technical details and show more numerical results. The new module for FLASH is open source and can be publicly accessed under https://github.com/jmark/DG-for-FLASH.

The FLASH code is a modular, parallel multi-physics simulation code capable of handling general compressible flow problems found in many astrophysical environments. It is a set of independent code units put together with a versatile setup tool written in Python while the code itself is written in Fortran90 and to minor extent in C. It uses the Message Passing Interface (MPI) library for inter-processor communication and the HDF5 or Parallel-NetCDF library for parallel I/O to achieve portability and scalability on a variety of different parallel computing systems. The framework provides three interchangeable grid modules: a block-structured Uniform Grid, a block-structured octree based adaptive grid using the PARAMESH library (Olson et al. 1999) and a block-structured patch based adaptive grid using Chombo (Colella et al. 2009). The architecture of the code is designed to be flexible and easily extensible.

The standard solver in FLASH is based on an AMR enabled second order FV method, where the fluid variables are stored in the form of mean values. FLASH organizes by

default the FV mean values in blocks of specific sizes, e.g., of size $8 \times 8 \times 8$. It is important to note that all other physics modules of FLASH assume that the data is organized in form of such blocks with mean values and hence that the interaction between the additional physics modules and the FLASH fluid solvers is based on mean values. In contrast, typically, in DG the fluid variables are stored in form of local polynomials with either modal coefficients (Karniadakis and Sherwin 2005) or nodal values (Kopriva 2009b). Depending on the polynomial degree $N-1$ of the local ansatz, N^3 unknowns per fluid variable form a data package. Thus, the size of the blocks (or in DG jargon size of the "elements"), varies with the choice of the polynomial degree.

This difference in the representation of the solutions and the interpretation of the solution coefficients forms a tough challenge that needs to be overcome. Roughly, we have to major choices: (i) We can implement the DG scheme in its typical/natural form based on the polynomial representations. This means, however, that we than need to adjust the whole FLASH code to cater to the piece-wise polynomial data. Thus, changes in the way the grid is managed (not blocks of constant size, but smaller elements) and changes in the way all other physics modules interact with the DG solver have to be changed and implemented. (ii) We adapt the DG methodology and specifically design a variant that directly operates with the block based mean value data structure of FLASH. The upside is of course that in this case DG can with reasonable implementation effort directly access all the functionality that the FLASH code offers. The downside is that this is not the "natural" way one would implement the DG methodology and that there are thus some disadvantages, such as additional transforms between ansatz spaces, that one needs to accept.

In this work, we have decided in favor of option (ii), where we adjust the DG methodology such that we can directly use the rich multi-physics framework that FLASH offers. In previous work, we have presented a robust and accurate DG variant that uses blocks of mean values as input (Markert et al. 2021), which serves as a starting reference for the FLASH implementation. Besides option (ii) being presumably the only feasible approach regarding the necessary amount of implementation, it further has beneficial effects such as being able to directly use the post-processing tool chains for data organized in blocks of mean values, which have been established in many years of research work.

6.2 Implementation Details

The convex blending scheme described in Section 4.7 has been implemented within FLASH as an independent solver module named DGFV under the `hydro/HydroMain/split` namespace. Although our numerical scheme is technically an unsplit solver it mimics the same interface as the other directionally split solvers, since most physics modules necessary for our envisaged astrophysics simulations only interface to directionally split schemes. Contrary to the described 3D FV scheme in Section 4.5, we cannot split the 3D DGSEM scheme from Section 4.6 into separate arrays of 1D data; at least not without compromising the accuracy. These so-called sweeps in spatial direction d are treated independently by directionally split schemes and assembled to the new solution afterwards. Fortunately, the split scheme interface is flexible enough to accommodate a high order (unsplit) DG scheme.

We focus our solver implementation on the octree-based grid unit PARAMESH, which is the default in FLASH. PARAMESH is built on the basic component of blocks consisting of $N_x \times N_y \times N_z$ regular cells extended with N_g layers of guard cells. The default configuration is $N_x = N_y = N_z := 8$ and $N_g := 4$, which allows us to implement a fourth order DG method ($N = 4$) with 2^3 elements embedded within a block. In Figure 6.1 one DG element is highlighted in blue, where the blue dots represent the Gauss quadrature nodes (4.50) at element boundaries. The red square represents a FV cell, or mean value, with face-centered red dots at cell interfaces.

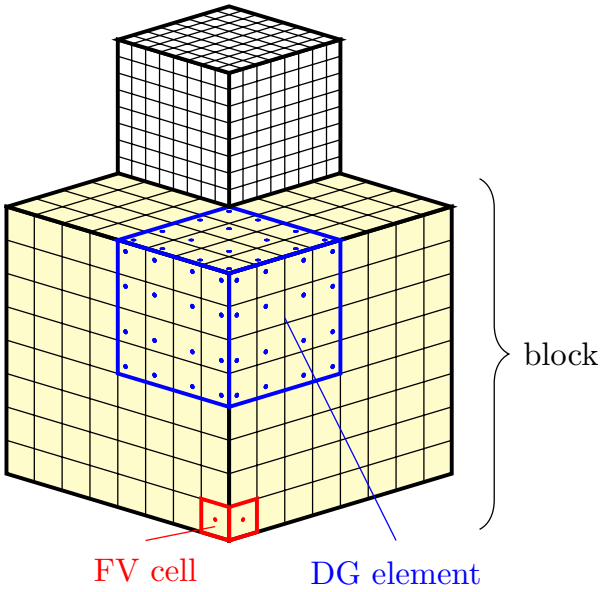

Figure 6.1: Cutout of a PARAMESH grid with one block of 8^3 cells (shaded in yellow) and a neighboring block (shaded in white) on top at a finer resolution level. One 4^{th} order DG element is highlighted in blue and the blue dots indicate the position of the Gauss quadrature nodes (4.50) at element interfaces. The red square with its face-centered red boundary points exemplifies one FV cell respectively mean value within the block.

For the sake of clarity, we change our perspective to a 2D crossection of our 3D PARAMESH grid as shown in Figure 6.2. The block of interest is colored in yellow while the guard cells are shaded in gray. The complete computational grid consists of a collection of such blocks with different physical cell sizes. They are related to each other in a hierarchical fashion using a tree data structure (octree), which evenly spreads over all parallel processors (MPI ranks) in case of distributed computing. Blocks do not overlap and there are only 1:4 relations (one coarse block interfaces to 4 finer blocks) allowed between neighbors sharing a common face. PARAMESH handles the filling of the guard cells with information from neighboring blocks or at the boundaries of the physical domain. If the block's neighbor has the same level of refinement, PARAMESH fills the corresponding guard cells using a direct copy from the neighbor's interior cells. If the neighbor is at a higher refinement level, the data is restricted via second order averaging. If the neighbor is at a lower refinement level, the data is prolongated via monotonic interpolation guaranteeing positive densities and pressures.

What follows is a brief outline of the algorithm implemented within the DGFV solver module. The module enters the Runge-Kutta cycle of m stages starting with the first stage. A full timestep is completed after exiting the loop with the last stage where we compute the

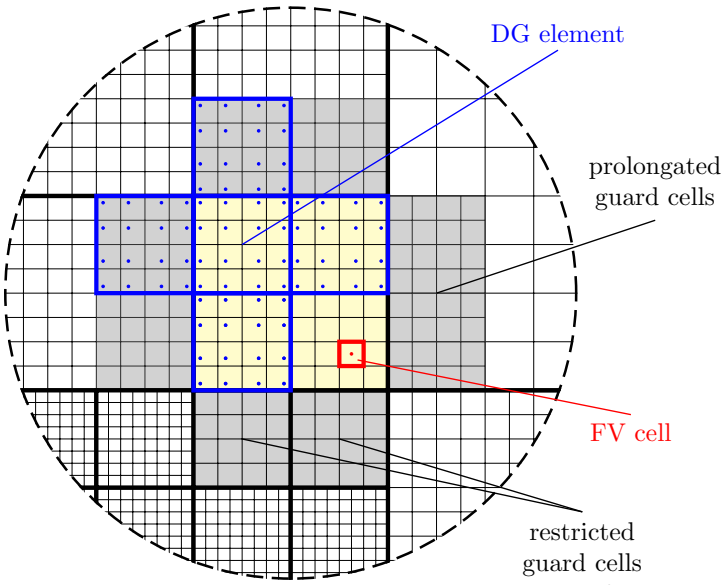

Figure 6.2: Cross section of a PARAMESH grid with blocks of 8^3 cells at different refinement levels (i.e. mesh resolution). The block of interest is shaded in yellow and its guard cell layer is shaded in gray. One 4^{th} order DG element with its adjacent elements is highlighted in blue and the blue dots indicate the position of the Gauss quadrature nodes (4.50). The red square with its red midpoint exemplifies one FV cell in the block.

global timestep of the next cycle according to the CFL condition specified in Section 4.7.2. The updated states are returned back to FLASH, which in turn calls the other modules handling different aspects of the simulation such as grid coarsening/refinement, load balancing, gravity, chemistry, etc. At the beginning of each Runge-Kutta stage the guard cells of each block are filled by PARAMESH with the latest data from direct neighbors. As already mentioned, the data transfer among MPI ranks, refinement and coarsening as well as prolongation and restriction along non-conforming interfaces is managed by PARAMESH and opaque to the solver. We reconstruct the nodal data of the DG elements via (4.98) and compute the blending factors according to Section 4.7.1. If any of the calculated blending factors is less then one, indicating under-resolved flow features within a DG element, the blending procedures are activated and we compute the standard FV solution described in Section 4.5. Otherwise it suffices to just compute the DG solutions from the reconstructed nodal values according to Section 4.6. If blending is active one has to determine the common surface fluxes (4.101) at DG element interfaces in order to maintain the conservation property. The final solution is then the convex blend (4.100) of both solutions. Moreover, the (common) surface fluxes at block boundaries are handed over to PARAMESH, which does a flux correction by replacing the coarse fluxes with the restricted fine fluxes at non-conforming block interfaces. We retrieve the corrected

fluxes and calculate the total surface flux error among all block boundaries. The error is then evenly deducted from all cells restoring mass conservation. Finally, we calculate the gravity source terms (3.55,3.56), which we add to the solution. The solver can now proceed to the next Runge-Kutta stage. This completes the outline of the algorithm.

Three remarks are in order. Firstly, FLASH stores its fluid data in primitive state variables. For DG we need to reconstruct on conservative state variables in order to maintain high order accuracy and stability. In other words, the conservative state variables are first calculated from the primitive state variables in mean value space and then transformed to nodal values. Secondly, the treatment at non-conforming block interfaces by PARAMESH is at most second order accurate. For DG the standard approach is the so-called mortar method (Kopriva et al. 2002), which matches the spatial order of the scheme if done correctly. However, we decided to stick with the default procedure provided by PARAMESH, since it would otherwise lead to substantial restructuring of the internal workings of the FLASH code. Furthermore, we observed that in practice the gain in accuracy by a higher order mortar method is negligible in our simulations, neither do we have any troubles with numerical artifacts or stability issues. Thirdly, physics and chemistry units in FLASH are designed and implemented around mean values, i.e. they expect mean values as input, do their calculations on mean values and produce mean values as output. This formally reduces the convergence order to at most second order, however, provides the benefit of directly using the rich collection of physics modules available in the FLASH framework.

6.3 Numerical Results

In this section, we present simulation results using our new fourth order (DGFV4) fluid solver module in FLASH. All test problems shown are computed on 1D, 2D or 3D Cartesian grids, for which the refinement level l in accordance to the convention in FLASH corresponds to $2^{l-1} \times 8$ grid points per dimension. We gradually increase the complexity of the test cases, where the later test cases are multi-physics applications with multiple physics modules working together.

6.3.1 Experimental Order of Convergence

First, we verify the experimental order of convergence (EOC) of our fourth order DGFV4 implementation. Measuring convergence of higher order MHD codes can be challenging

since smooth MHD test problems present numerical subtleties that are potentially revealed by the very low numerical dissipation of higher order methods. In this work we rely on widely-used smooth MHD test problems for convergence assessment: the nonlinear circularly polarized Alfvén waves in 2D and a manufactured solution setup for 3D. For our convergence tests, we enabled any shock indicators and limiters on purpose in order to confirm that the limiters do not interfere for smooth well resolved solutions. We verified the EOC on conforming Cartesian grids, since the treatment at non-conforming block interfaces is not high order as already discussed in the previous section.

Circularly polarized smooth Alfvén waves are exact analytic solutions of the MHD equations for arbitrary wave amplitude perturbations. Hence, they are suitable to study the experimental convergence order of the scheme, as well as its amount of numerical dispersion and dissipation errors. The solution consists of plane waves in which the magnetic field and velocity oscillate in phase in a circular polarization perpendicular to the propagation direction. We initialize the Alfvén waves within a periodic 2D domain $\Omega = [0, 0] \times [\cos^{-1}(\alpha), \sin^{-1}(\alpha)]$ at an angle of $\alpha = 45°$. The longitudinal Alfvén wave speed is fixed at $|v_A| = B_{||}/\sqrt{\rho} := 1$ such that the wave returns to its initial state at integer times $T \in \mathbb{N}$. For our test we let the wave turn around five times, i.e. $T = 5$. The heat capacity ratio is $\gamma = 5/3$. With the rotated coordinate $\hat{x} = x\,\cos(\alpha) + y\,\sin(\alpha)$, we define $B_\perp = 0.1\,\sin(2\pi\,\hat{x})$ and $B_3 = 0.1\,\cos(2\pi\,\hat{x})$. The initial state in primitive state variables then reads $(\rho, \vec{v}, p, \vec{B}, \Psi)^T = (1.0, \vec{v}_0, 0.1, \vec{B}_0, 0)^T$ with $\vec{v}_0 = (-B_\perp \sin(\alpha), B_\perp \cos(\alpha), B_3)^T$ and $\vec{B}_0 = (\cos(\alpha), \sin(\alpha), 0)^T + \vec{v}_0$. We look at the total pressure (3.20) and calculate the ∞-norm and 2-norm according to (4.25) and (4.26). Since the total pressure involves all state variables it is a good quantity to measure the overall convergence of the code. Moreover, the reference solution has to be evaluated with high order accuracy as already discussed in Section 4.7.3. The results are shown in Table 6.1. The formally fourth order DGFV4 scheme in 2D yields the expected EOCs.

In order to verify the convergence order in 3D we construct a manufactured solution, which reads in primitive state variables as

$$\mathbf{\Phi}^{\text{man}} = \Big(h, 0.1, 0.2, 0.3, h, h, -h, 1, 0\Big)^T \tag{6.1}$$

with $h = h(t, x, y, z) = 0.5\sin(2\pi(x + y + z - t))$. The domain is a cubic box $\Omega = [0, 1]^3$ and the heat capacity ration is set to $\gamma = 2.0$. At each Runge-Kutta stage we subtract

Table 6.1: EOC of total pressure P of the smooth Alfvén wave problem in 2D run by the fourth order DGFV4 scheme.

DOF	$\left\|\left\|P\right\|\right\|_\infty$	$\left\|\left\|P\right\|\right\|_2$	EOC_∞	EOC_2
16^2	9.852e-05	4.951e-05	n/a	n/a
32^2	6.086e-06	2.110e-06	4.016	4.552
64^2	3.634e-07	1.285e-07	4.065	4.036
128^2	2.334e-08	8.002e-09	3.960	4.006
256^2	1.461e-09	4.998e-10	3.997	4.000
512^2	9.221e-11	3.423e-11	3.985	3.868

Table 6.2: EOC of total pressure P of the manufactured solution problem in 3D run by the fourth order DGFV4 scheme.

DOF	$\left\|\left\|P\right\|\right\|_\infty$	$\left\|\left\|P\right\|\right\|_2$	EOC_∞	EOC_2
16^3	1.259e-03	4.122e-04	n/a	n/a
32^3	4.998e-05	1.773e-05	4.655	4.538
64^3	3.248e-06	1.015e-06	3.943	4.126
128^3	1.962e-07	6.594e-08	4.048	3.944
256^3	1.290e-08	4.141e-09	3.927	3.992
512^3	7.963e-10	2.540e-10	4.018	4.027

the residual

$$\boldsymbol{\Upsilon}^{\text{man}} = \partial_x F\Big|_{\boldsymbol{\Phi}^{\text{man}}} + \partial_y G\Big|_{\boldsymbol{\Phi}^{\text{man}}} + \partial_z H\Big|_{\boldsymbol{\Phi}^{\text{man}}} \tag{6.2}$$

from the right-hand-side $\dot{\boldsymbol{u}}$ and advance in time. As before, the source term is evaluated with an appropriate quadrature rule and projected to mean values. At final time $T = 0.1$ the convergence of the total pressure P is computed as previously described in the Alfvén wave test. The results in Table 6.2 confirm the correct EOCs for the formally fourth order DGFV4 scheme in 3D.

6.3.2 Divergence Control

We now turn to test problems more specifically aimed at evaluating the efficacy of the hyperbolic divergence cleaning approach discussed in Section 3.4.

Magnetic Loop Advection

This test investigates the proper advection of a magnetic field loop (Gardiner and Stone 2005). On a periodic domain $\Omega = [-0.5, 0.5]^2$, the background medium has $\rho = 1$, $p = 1$,

and a global advection velocity $(v_1 v_2) = (2, 1)$ so that the ambient flow is not aligned with grid directions. Letting $r = \sqrt{x^2 + y^2}$ be the radial distance to the center of the domain, the magnetic field is initialized from a vector potential $\vec{A} = (0, 0, A_3(r))$ with $\vec{B} = \nabla \times \vec{A}$. To define a magnetic field loop of radius $r_0 = 0.3$, we set $A_3(r) = \max(0, 10^{-3}(r_0 - r))$ and obtain a very weakly magnetized configuration with a plasma β of order 10^6, in which the magnetic field is essentially a passive scalar. For this field configuration, the MHD current vanishes everywhere, except at $r = 0$, and $r = r_0$ where the corresponding current line and current tube become singular. Note, that we added a smoothing parameter $\delta r = 0.05$ to the radius r in order to avoid spurious ringing caused by the unresolved singularity of the initial magnetic field at the domain center. The aim of the test is to verify that the current loop is advected without deformation or noise.

Figure 6.3 shows the z-component of the current density $\vec{j} = \nabla \times \vec{B}$ at final time $T = 1$ corresponding to two horizontal domain crossings and at a uniform resolution of 128^2 DOF. The current density is a stringent diagnostic since, being a derivative of the magnetic field, it is very sensitive to noise and local fluctuations. The scheme preserves the exact circular shape of the current loop very well, with very little noise and oscillations in the current.

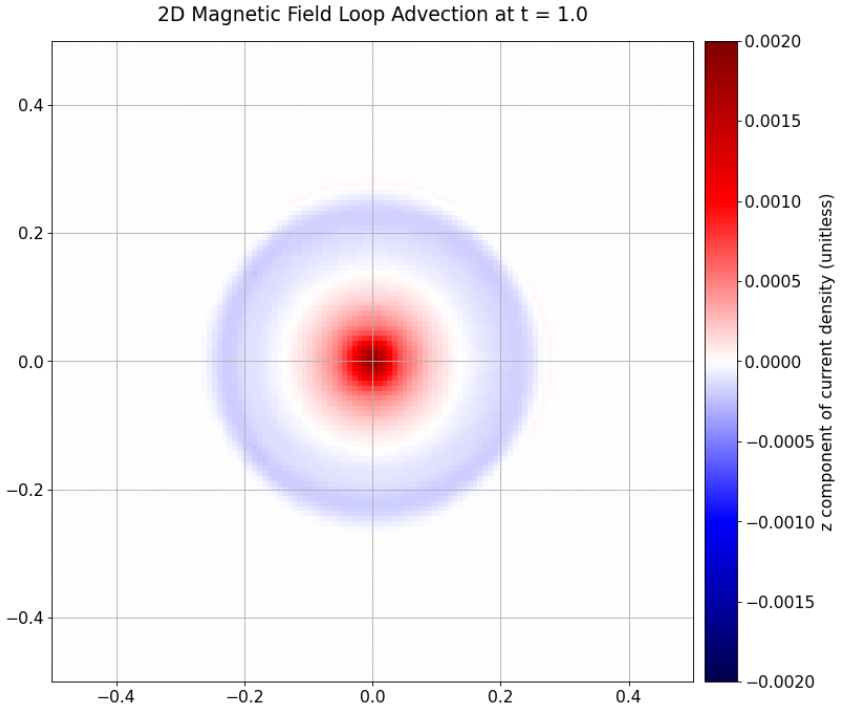

Figure 6.3: Shown is the z-component of the current density $\vec{j} = \nabla \times \vec{B}$ at final time $T = 1.0$ as calculated with the DGFV4 scheme.

MHD Current Sheet

The two-dimensional current sheet problem in ideal MHD regimes has been extensively studied before (see e.g. Gardiner and Stone 2005; Guillet et al. 2019; Rastätter et al. 1994; Rueda-Ramírez et al. 2021). Two magnetic currents are initialized in opposite direction sharing a common interface and disturbed with a small velocity fluctuation, which provokes magnetic reconnections. In the regions where the magnetic reconnection takes place, the magnetic flux approaches very small values and the lost magnetic energy is converted into internal energy. This phenomenon changes the overall topology of the magnetic fields and consequently affects the global magnetic configuration.

The square computational domain is given as $\Omega = [-0.5, 0.5]^2$ with periodic boundary conditions. We initialize the setup in primitive variables

$$(\rho, \vec{v}, p, \vec{B}, \Psi)^T = (1.0, v_1, 0, 0, 0.05\, B_0^2, 0, B_2.0, 0)^T \tag{6.3}$$

with $B_0 = 1/\sqrt{4\pi}$, $v_1 = 0.1 \sin(2\pi y)$ and $B_2 = -B_0$ when $-0.25 < x < 0.25$ and $B_2 = B_0$ in the rest of the domain. The heat capacity ratio is set to $\gamma = 5/3$ and the final simulation time is $T = 10$.

The changes in the magnetic field seed the magnetic reconnection and develop formations of magnetic islands along the two current sheets. The small islands are then merged into bigger islands by continuously shifting up and down along the current sheets until there is one big island left in each current sheet. Technically, changes in the field topology are not allowed in the ideal MHD regime due to the absence of any resistivity. Since high order DG needs to be stabilized for this setup, our blending scheme injects enough numerical resistivity through the FV scheme to allow for field reconnections. In Figure 6.4 we show the magnetic pressure with overlayed field lines (left) and the blending factor (right) at an earlier stage in the simulation. Evidently, at the shear layer between countering magnetic field lines, where the reconnections take place, the blending activity is strongest.

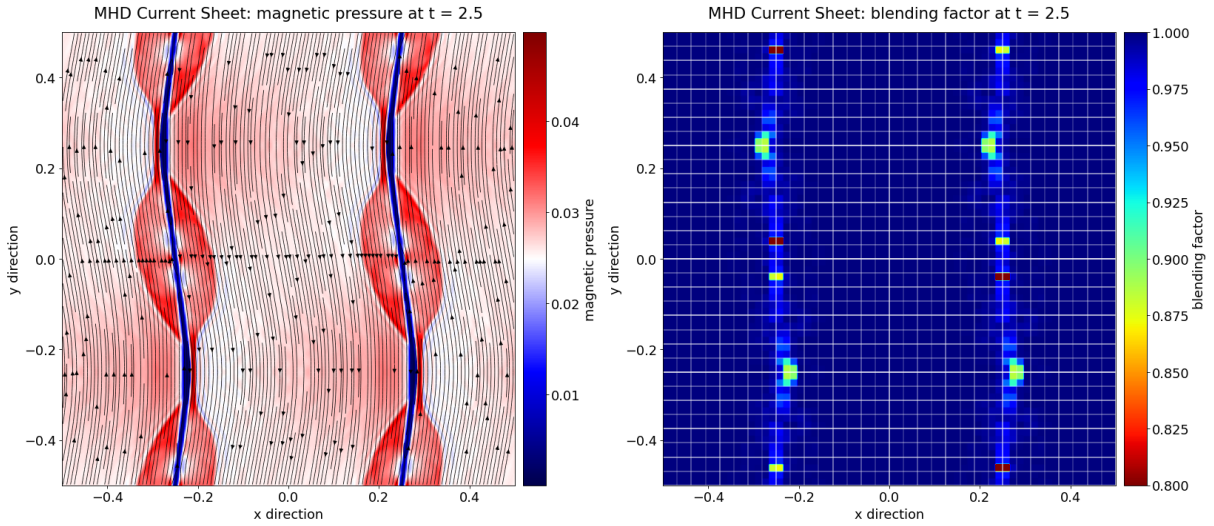

Figure 6.4: Magnetic pressure (left) with overlayed magnetic field lines (black) and blending factors (right) with overlayed grid lines (white) for the MHD current sheet test at simulation time $t = 2.5$.

The final result is shown in Figure 6.5, which depicts the z-component of the current density with overlayed magnetic field lines.

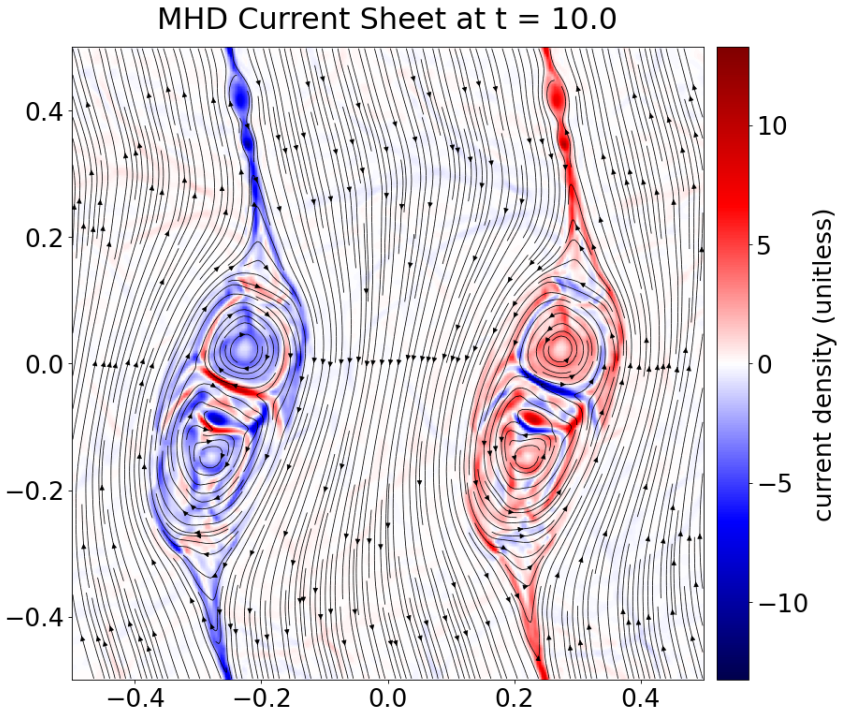

Figure 6.5: Shown is the z-component of the current density $\vec{j} = \nabla \times \vec{B}$ with overlayed magnetic field lines for the MHD current sheet test at final time $T = 10$.

Figure 6.6 shows the domain-integrated divergence error over time for three different runs: without any divergence cleaning (none; blue line), with just Powell source terms (Powell; orange line) and with Powell source terms plus hyperbolic divergence cleaning (Powell + hyb. div. cleaning; green line) which is the default in our code. The first simulation crashes early on due to the rapid surge of divergence error leading to the well known instability issues of uncontrolled magnetic field divergence growth in MHD simulations (Brackbill and Barnes 1980; Tóth 2000; Kemm 2013). The second run with Powell terms does not crash and is at least able to keep the overall errors at bay over the course of the simulation (see also Figure 6.7). As expected, the run with hyperbolic divergence cleaning has the lowest divergence error at all times. This particular setup is insofar challenging in that it perpetually prompts a significant production of divergence errors, thus highlighting the importance of a proper and robust divergence cleaning technique.

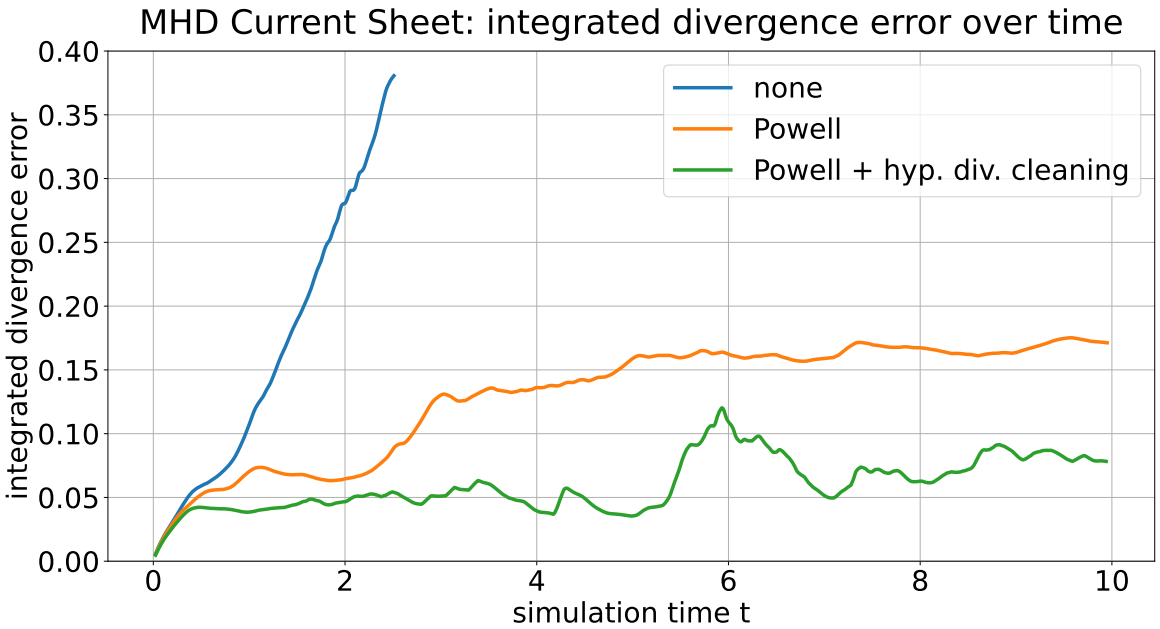

Figure 6.6: Evolution of the domain-integrated absolute magnetic field divergence $\int_\Omega |\nabla \cdot \vec{B}(t)| d\Omega$ for the MHD current sheet test. Three runs with different flavors of divergence cleaning are shown. The simulation without any divergence correction method ("none"; blue line) crashed at around $t = 2.5$. The run where only the Powell source terms are used (orange line) is stable but has a substantially larger divergence error than the full scheme with Powell source terms and hyperbolic divergence cleaning (green line) which is the default setting in our code.

Figure 6.7 shows the time evolution of the kinetic, internal, magnetic, and total energies

of the current sheet problem with (solid lines) and without (dashed lines) hyperbolic divergence cleaning activated. The decline of the magnetic energy is compensated by the increase in internal energy due to the heating driven by the magnetic reconnection. Furthermore, without divergence cleaning a significant total energy drift is introduced by the non-conservative Powell terms and the accompanied high divergence errors, which are not properly corrected for in this case. This energy drift also causes artificial cooling at the reconnection points giving rise to a non-physical behavior. However, with hyperbolic divergence cleaning the total energy is conserved.

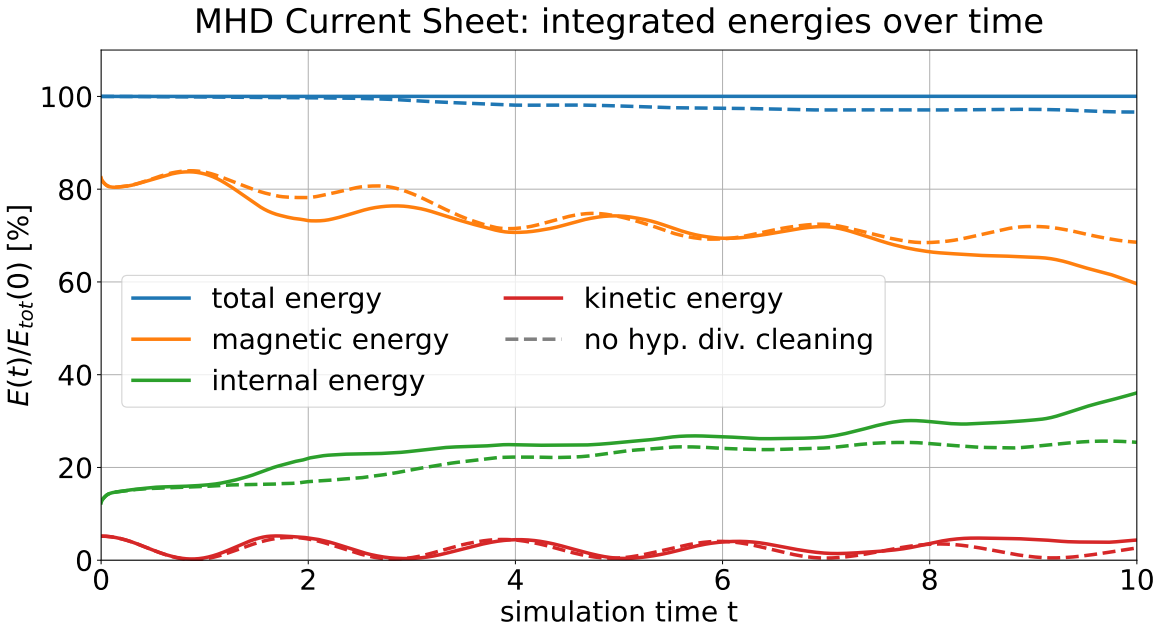

Figure 6.7: The percentage of total and individual energies as a function of time for the MHD current sheet test. We compare the results with (solid lines) and without (dashed lines) our hyperbolic divergence cleaning method. The results for the run without any divergence treatment are not shown, since it crashed early on.

6.3.3 Shock Problems

In this section, we test the correct resolution of shock waves inherent to ideal MHD regimes by our DGFV4 scheme in 1D, 2D and 3D. Furthermore, one shock tube problem investigates the proper handling of waves in a multi-species setting and if applicable we utilize adaptive mesh refinement for a further challenge. Note that we actually run our 1D test problems in 2D, with perfectly y-independent initial conditions and periodic boundaries. A correct implementation of our multidimensional scheme does not develop

any y-dependence of the solution.

Briu-Wu Shock Tube

The established MHD shock tube problem introduced by Brio and Wu (1988) has now become a classic shock test for MHD codes. For this test, we take the computational domain to be $[0, 1]$ with outflow boundaries left and right. In the whole domain, the flow is initially at rest ($v = 0$) and $(B_1, B_2) = (0.75, 0)$. The initial primitive variables are discontinuous at $x = 0.5$, with the left and right states given by $(\rho, p, B_2)_{\mathrm{L}} = (1, 1, 1)$ and $(\rho, p, B_2)_{\mathrm{R}} = (0.125, 0.1, -1)$, respectively. We set $\gamma = 2$, and run the simulation until the final time $T = 0.1$.

Figure 6.8 presents the density, pressure, y-component of the magnetic field, and blending factor of the Brio-Wu shock tube test problem at final time $T = 0.1$, using the DGFV4 scheme on an AMR grid with a minimum resolution of 32 cells and a maximum resolution of 512 cells. The reference solution is obtained using the latest version of ATHENA (v4.2) (Stone et al. 2008) with default settings on 2048 cells. Our DGFV4 scheme captures all the MHD waves correctly, and the limiter sharply resolves the shocks and the contact discontinuity within very few cells. The limiter sufficiently suppresses overshoots and oscillations around shocks. Furthermore, our scheme cooperates well with the standard AMR method provided by FLASH (not directly visible in Figure 6.8) and properly traces the shock fronts at the highest resolution possible.

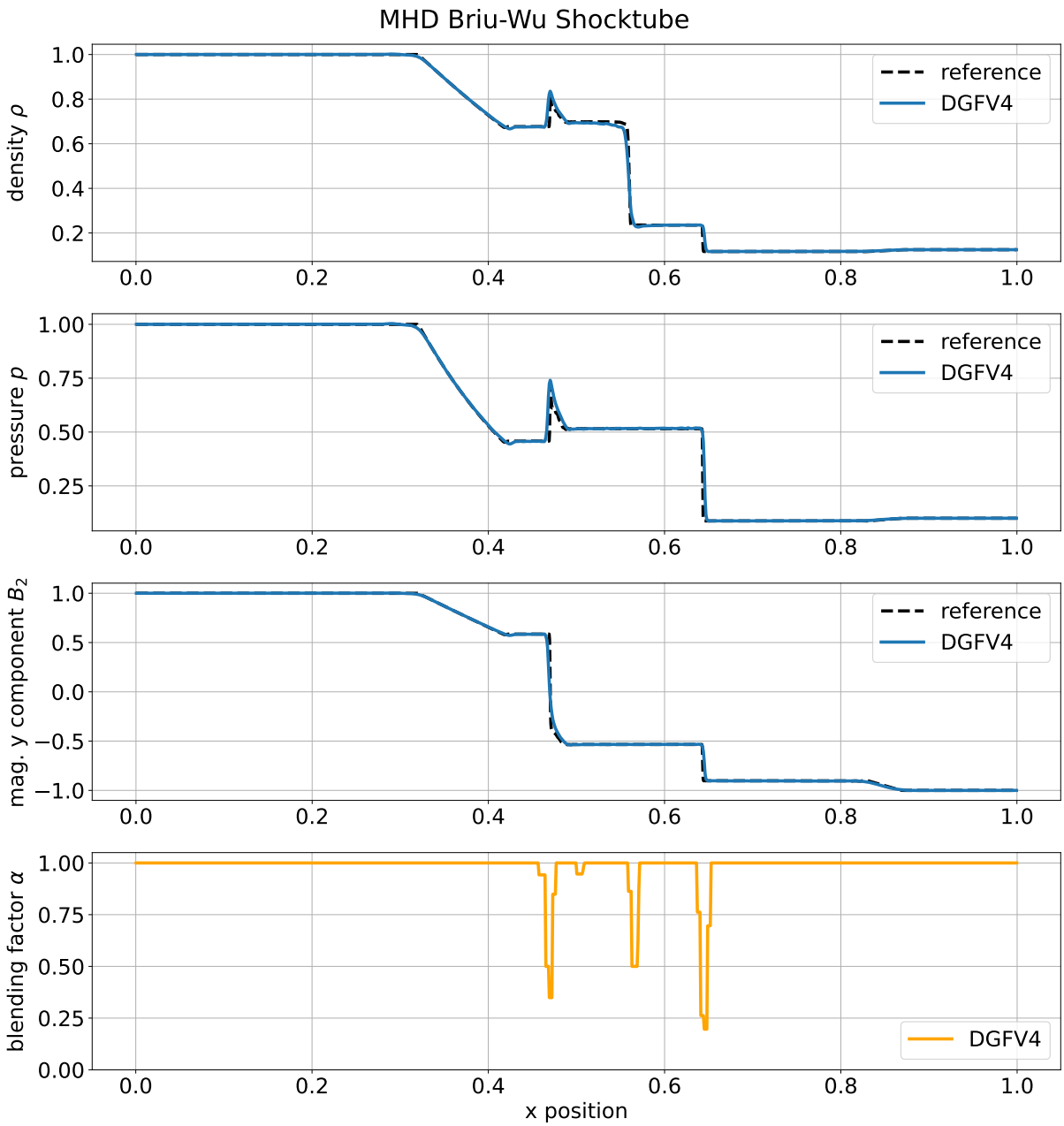

Figure 6.8: Density, pressure, magnetic field, and blending profiles of the Briu-Wu shock tube at final time $T = 0.1$ run by the DGFV4 scheme on an AMR grid with a minimum resolution of 32 cells and a maximum resolution of 512 cells. We compute the reference solution with the ATHENA code using a resolution of 2048 cells.

MHD Shu-Osher Shock Tube

The MHD version of the 1D Shu-Osher shock tube test (Shu and Osher 1988) proposed by Susanto (2014) becomes increasingly popular (Derigs et al. 2016; Guillet et al. 2019) to test the scheme's ability to resolve small-scale flow features in the presence of strong shocks in ideal MHD regimes.

The setup follows the interaction of a supersonic shock wave with smooth density perturbations. The computational domain is $\Omega = [-5, 5]$ with outflow boundary conditions. At $t = 0$, the shock interface is located at $x_0 = -4$. In the region $x \leq x_0$, a smooth supersonic flow is initialized with primitive states given by

$$(\rho, \vec{v}, p, \vec{B}, \Psi)_{\mathrm{L}} = (3.5, 5.8846, 1.1198, 0, 42.0267, 1, 3.6359, 0, 0). \tag{6.4}$$

In the rest of the domain $x > x_0$, smooth stationary density perturbations are setup in primitive state as

$$(\rho, \vec{v}, p, \vec{B}, \Psi)_{\mathrm{R}} = (1 + 0.2\sin(5\,x), 0, 0, 0, 1, 1, 1, 0, 0). \tag{6.5}$$

The flow is evolved until the final time $T = 0.7$.

The resulting profiles in density, pressure, y-component of the magnetic field, and blending factor at the final time T are shown in Figure 6.9, for two maximum resolution levels: 256 cells and 512 cells. As in the Briu-Wu shock tube, we employ AMR in order to confirm the correct tracking of the small-scale flow features and the smooth interplay with our MHD solver. The reference solution has been computed using ATHENA on 2048 grid cells. Our scheme properly resolves all expected flow features for both levels of resolutions shown.

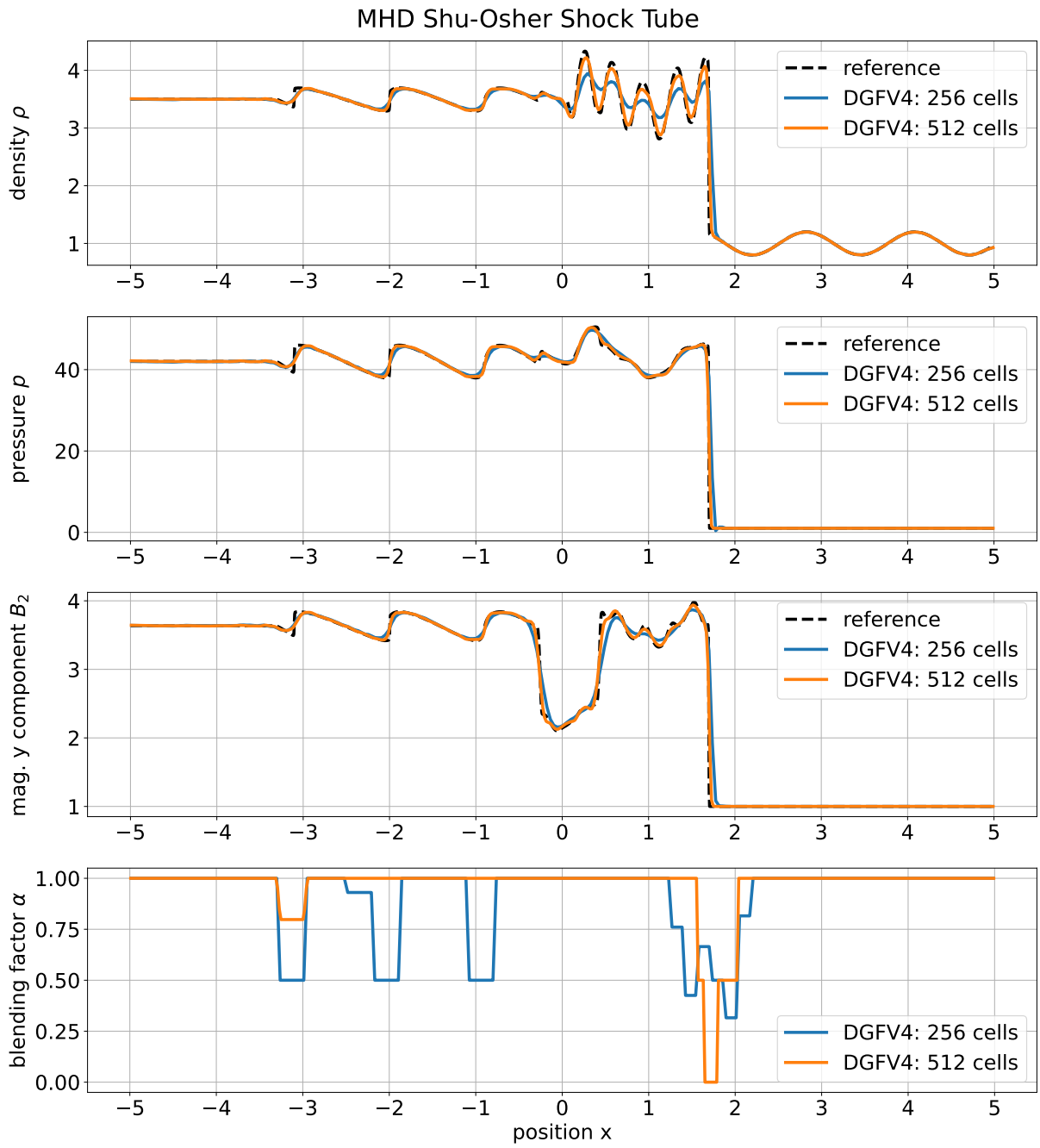

Figure 6.9: MHD Shu-Osher shock tube test problem: The density, pressure, magnetic field, and blending profiles are shown at final time $T = 0.7$. The reference solution has been computed using ATHENA on 2048 grid cells.

Two-component Sod Shock Tube

We simulate a Sod shock tube problem extended with two species as done for example in Gouasmi et al. (2020). The computational domain is $\Omega = [0, 1]$ with outflow boundary conditions. At $t = 0$, the shock interface is located at $x_0 = 0.5$. In the region $x \leq x_0$, the fluid is initialized with primitive states given by $(\rho, v, p, \sigma_1, \sigma_2) = (1, 0, 1, 1, 0)$. In the rest of the domain $x > x_0$, we setup the primitive states $(\rho, v, p, \sigma_1, \sigma_2) = (0.125, 0, 0.1, 0, 1)$. The individual heat capacity ratios of the two fluid components are $\gamma_1 = 1.4$ and $\gamma_2 = 1.6$ with equal heat capacities for constant volume $c_1^{\text{vol.}} = c_2^{\text{vol.}} = 1$. The shock tube is evolved until the final time $T = 0.2$ run by the DGFV4 scheme on an AMR grid with a minimum resolution of 32 cells and a maximum resolution of 256 cells. Figure 6.10 shows the total density, pressure, specific heat ratio profiles, and blending factor at the final time. There is an excellent agreement with the exact solution provided by Karni (1994).

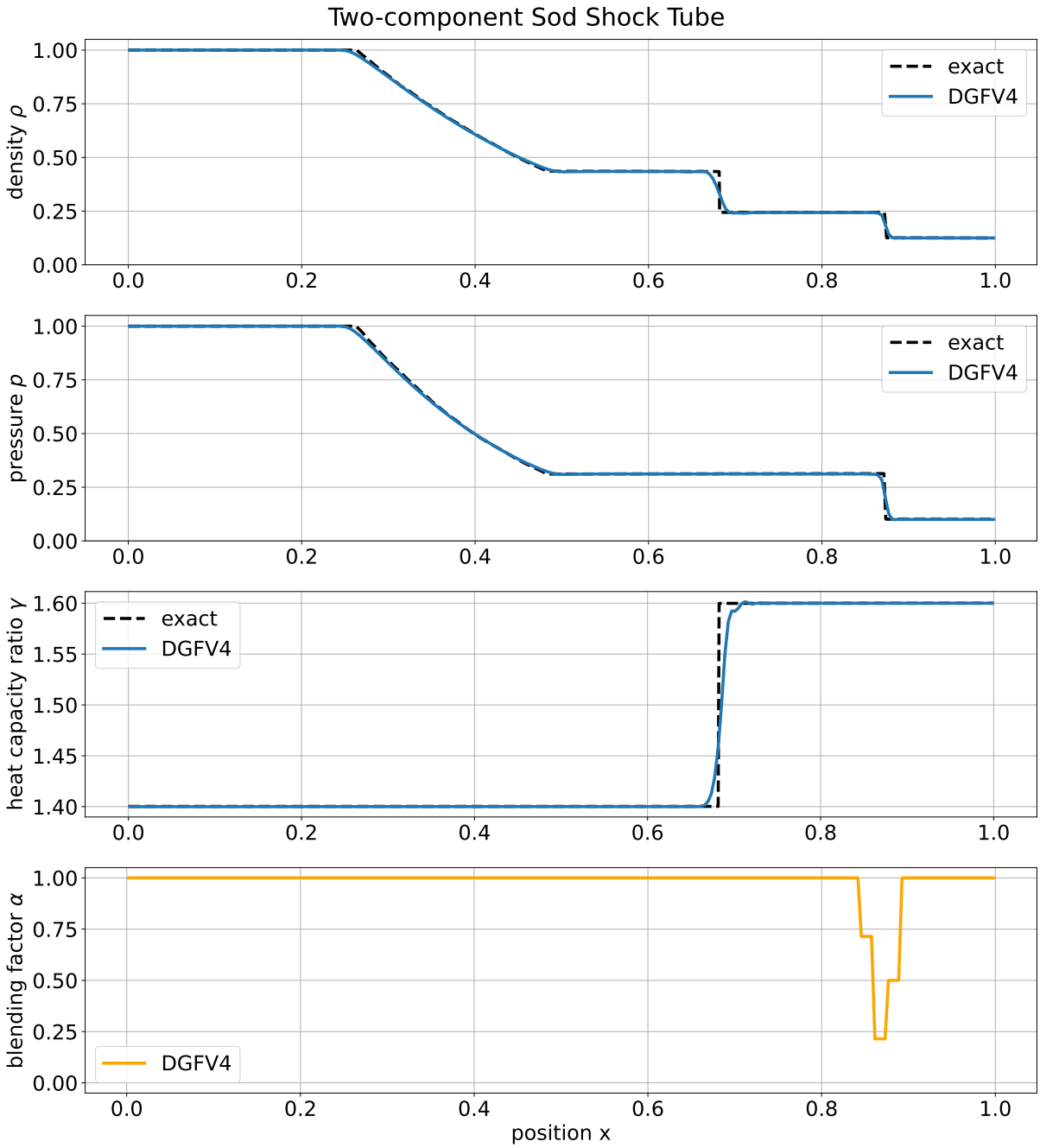

Figure 6.10: Shown are the profiles of the numerical solution of the two-component Sod shock tube problem computed with the DGFV4 scheme (on an AMR grid with a minimum resolution of 32 cells and a maximum resolution of 256 cells) together with the exact solution at final time $T = 0.2$. Apparently, the blending is triggered at the pressure jump as designed. The contact discontinuity (density jump between $x = 0.6$ and $x = 0.8$) is sufficiently resolved by DG without any need in stabilization via blending.

Sedov Blast

The purely hydrodynamical Sedov blast problem (Zhang and Shu 2010, 2012; Dumbser et al. 2018) describes the self-similar evolution of a radially symmetrical blast wave from an initial pressure point (delta distribution) at the center into the surrounding, homogeneous medium. The analytical solution is given by Sedov (1959); Korobeinikov (1991). In our setup, we approximate the initial pressure point with a smooth Gaussian distribution

$$\mathcal{E}_0(\vec{x}) = \frac{p_0}{\gamma - 1} + \frac{E}{(2\,\pi\,\sigma^2)^{d/2}} \exp\Big(-\frac{1}{2}\frac{\vec{x}^2}{\sigma^2}\Big), \tag{6.6}$$

with the spatial dimension $d = 2$, the blast energy $E = 1$ and the width σ such that the initial Gaussian is reasonably resolved. The surrounding medium is initialized with $\rho_0 = 1$ and $p_0 = 10^{-14}$. From Section 3.5 we know that the analytical solution of the density right at the shock front is determined by

$$\rho_{\text{shock}} = \frac{\gamma + 1}{\gamma - 1}\,\rho_0.$$

With the adiabatic coefficient $\gamma = 1.4$, we investigate how close the numerical results match $\rho_{\text{shock}} = 6$. The Cartesian AMR mesh goes from minimum resolution of 32^2 to maximum resolution of 512^2 cells. The spatial domain is $\Omega = [-0.25, 0.25]^2$ with the initial blast width $\sigma = 5 \times 10^{-3}$. Figure 6.11 shows the contour plots of density (left) and pressure (right) at final time $T = 0.05$. The solution looks clean and sharp, without visible noise or other numerical artifacts.

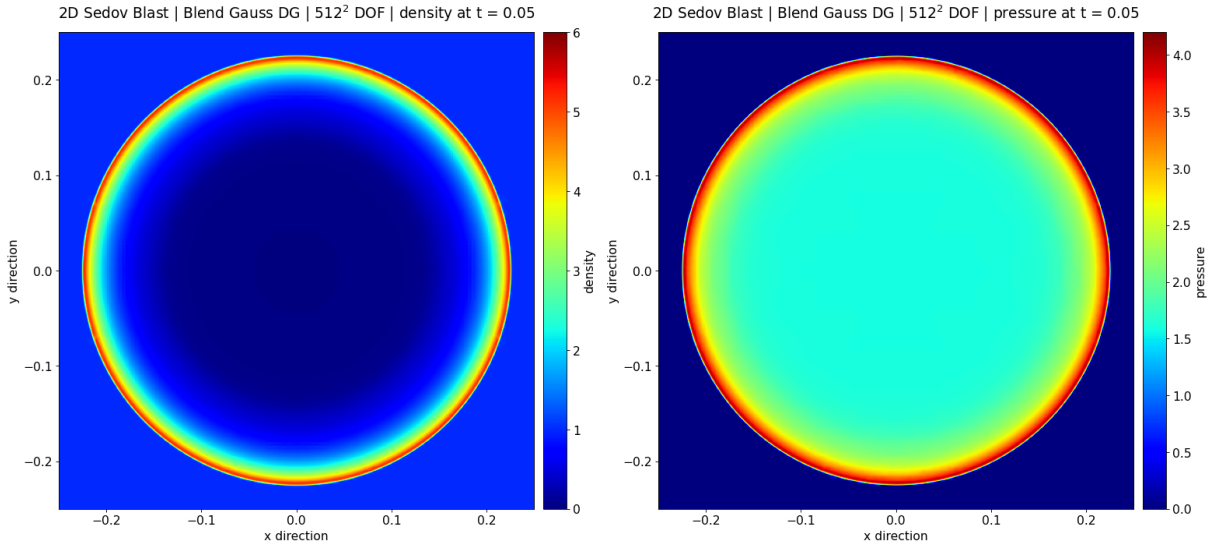

Figure 6.11: Density (left) and pressure (right) contours of the 2D Sedov Blast setup computed with the DGFV4 scheme at final simulation time $T = 0.05$.

In Figure 6.12 we take a sub-section in the first quadrant of Figure 6.11 and show zoom-ins of density (top left) and pressure (bottom left) contours along with the blended element entropy production rate (top right) given by (4.108) and blending factors (bottom left). The grid lines are overlayed in black (left) and white (right), showing the proper tracing of the shock front by the AMR procedure. The blending factor aligns perfectly with the shock front as intended. In Section 3.5 we discussed that shocks produce immense amounts of entropy through dissipative processes, despite the lack of viscosity in the model of compressible Euler equations. The strong negative entropy production rates shown in the top left plot in Figure 6.12 uncovers the indispensable injection of numerical dissipation into the shocked solution in order to stabilize the computation.

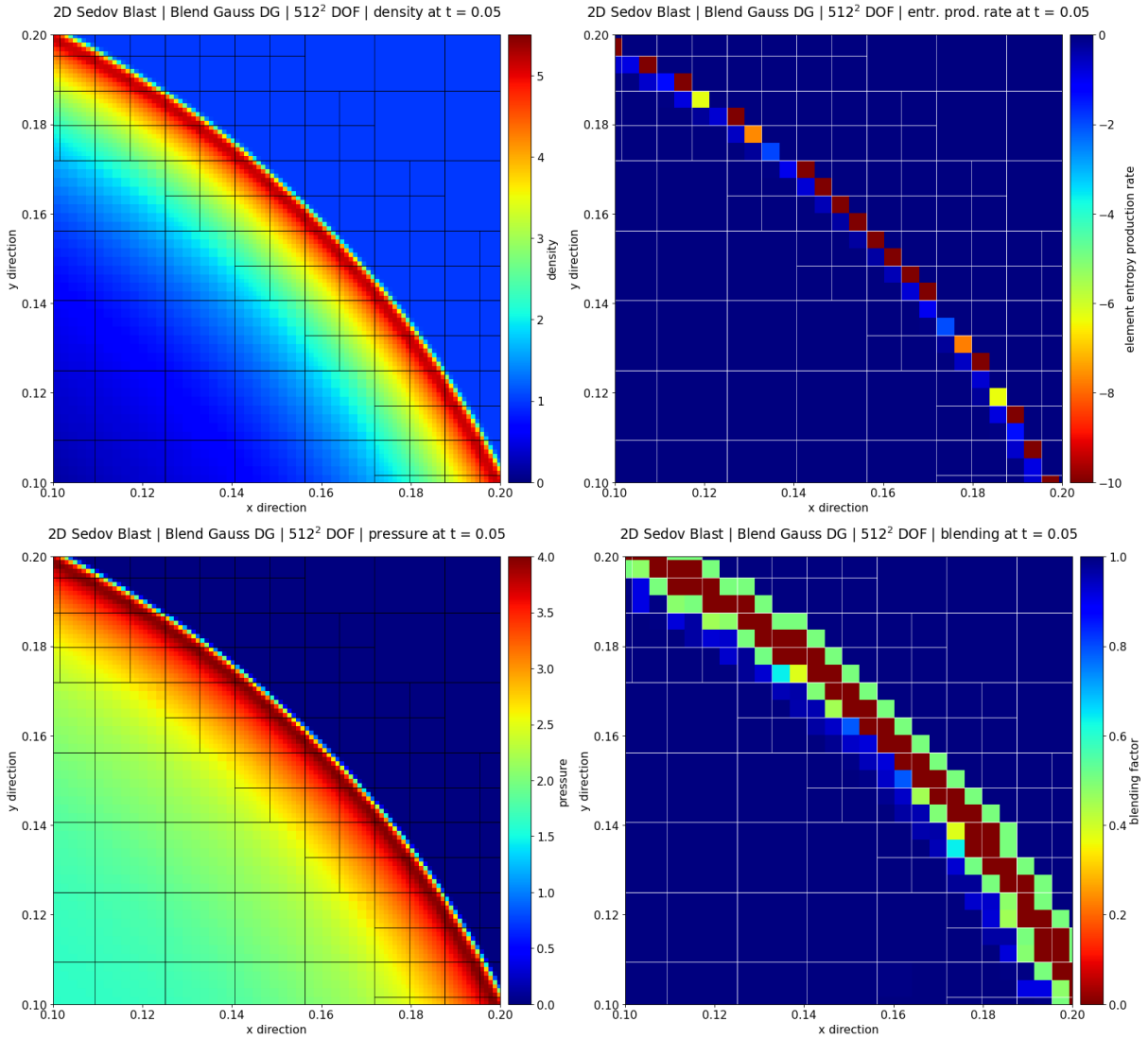

Figure 6.12: Zoom-in of Figure 6.12 together with blended element entropy production rates (top right) given by (4.108) and blending factor (bottom right). The grid lines are overlayed in black (left) and white (right).

In Figure 6.13 we show the shell-average profiles of normalized density (ρ/ρ_0), normalized pressure (p/p_0), scaled entropy production rate ($\Delta\dot{\mathcal{S}}/100$), and blending factor ($\alpha$) combined into one plot. The numerical solution of density and pressure follow the exact solution (dashed) nicely, confirming the correct modeling of the shock's propagation speed. The peak in entropy production is exactly at the height of the shock and the blending factor wraps around the density and pressure jumps like a protective layer keeping DG from polluting the solution.

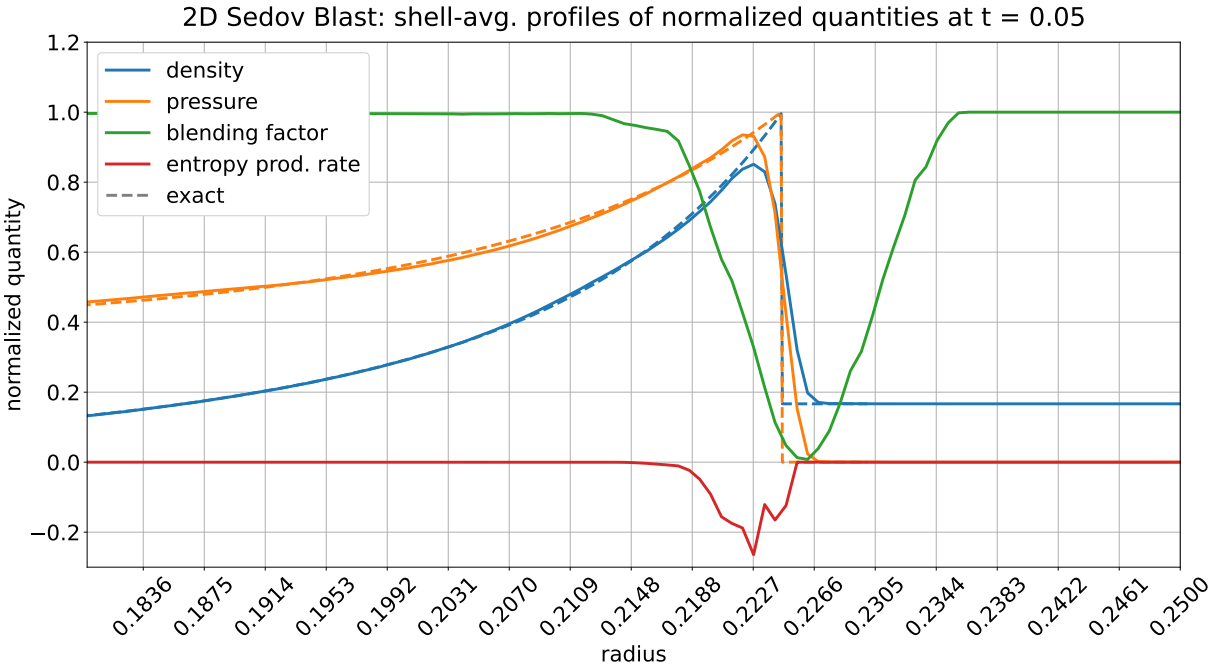

Figure 6.13: The profiles of shell-average profiles of normalized density (ρ/ρ_0), normalized pressure (p/p_0), scaled entropy production rate ($\dot{\Delta\mathcal{S}}/100$), and blending factor ($\alpha$) of the 2D Sedov blast setup computed with the DGFV4. The exact solution for density and pressure is shown as dashed lines.

Orszag-Tang Vortex

Now we look at the 2D Orszag–Tang vortex problem (Orszag and Tang 1979), a widely-used test problem for ideal MHD. The vortex starts from a smooth initial field configuration, and quickly forms shocks before transitioning into turbulent flow. For this problem, our computational domain is $\Omega = [0, 1]^2$ and we use $\gamma = 5/3$. The initial density and pressure are uniform, $\rho = 1$ and $p = 1/\gamma$. The initial fluid velocity is $\vec{v} = (-\sin(2\pi\, y), \sin(2\pi\, x), 0)^T$, and the initial magnetic field is $\vec{B} = (-\sin(2\pi\, y), \sin(4\pi\, x), 0)^T/\gamma$.

The final solution at time $T = 0.5$ is shown in Figure 6.14. We employ an AMR grid with resolution levels going from 64^2 cells to 1024^2 cells, which is highlighted as black lines on the left half of the density plot. We recognize the well-known density distribution of the Orszag–Tang vortex, which is commonly presented in MHD code papers. Note that we obtain both sharp shocks and smooth, noise-free flow with resolved features between the shocks.

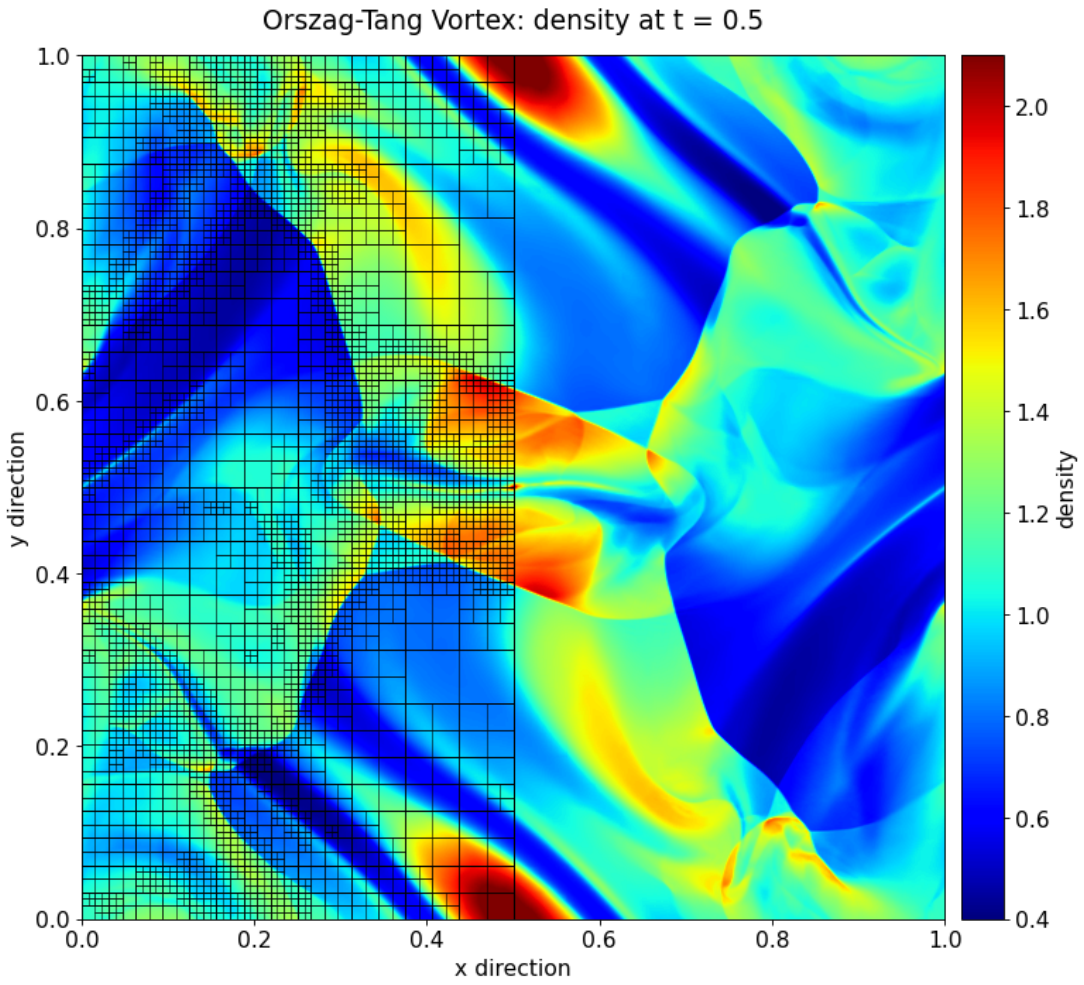

Figure 6.14: Density profile of the Orszag-Tang vortex at the final time $T = 0.5$ as calculated with our DGFV4 scheme on an AMR mesh (shown as black lines in the left half of the plot) with a maximum refinement level of $l = 8$, which is equivalent to a maximum resolution of 1024^2 cells.

In order to confirm the correct positioning of the shock waves, we computed a reference solution on a finer grid of 2048^2 cells with the ATHENA code and overlay 1D cuts of both pressure solutions in Figure 6.15. The resulting profile of our scheme matches the reference very well and is very sharp and without spurious oscillations or overshoots.

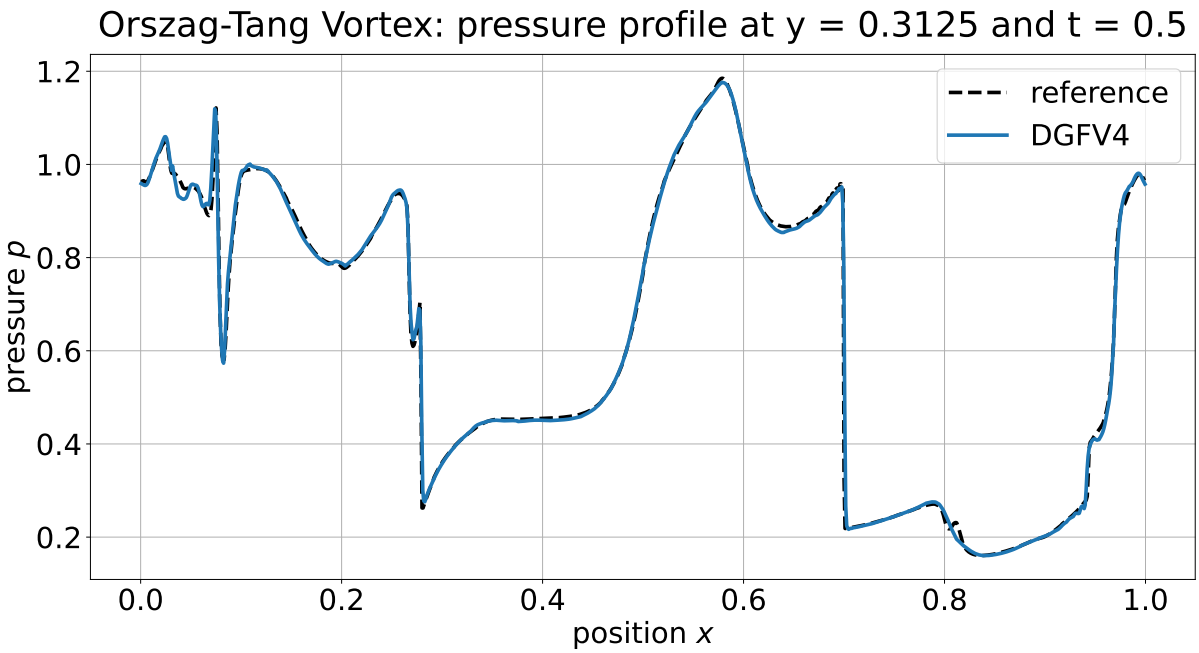

Figure 6.15: Numerical solution of the pressure profiles at $y = 0.3125$ and time $T = 0.5$ are shown for the DGFV4 solver with AMR and a maximum resolution of 1024^2 DOF. The reference solution was computed with the Athena code at a uniform resolution of 2048^2 DOF.

During our numerical experiments we found that without proper divergence cleaning, distinctive grid artifacts appear, which considerably pollute the solution. Analog to Section 6.3.2 we compared the time evolution of the divergence errors with and without divergence control mechanisms. The results shown in Figure 6.16 clearly demonstrate that the hyperbolic divergence cleaning method is effective in confining the divergence errors ensuring a clean, unpolluted MHD flow.

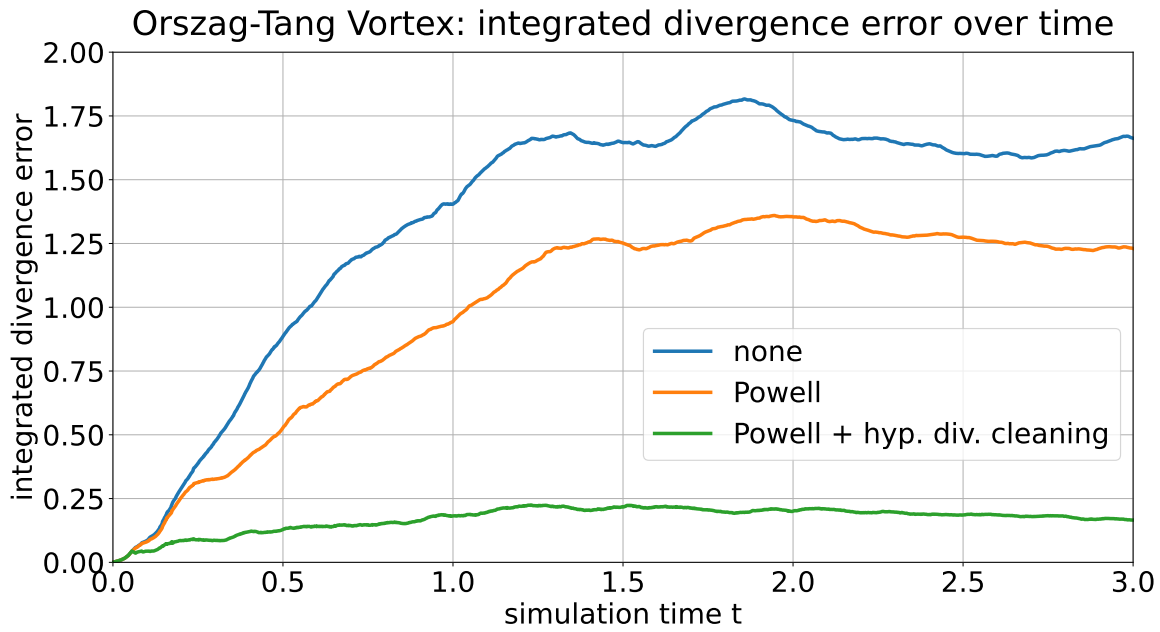

Figure 6.16: Domain-integrated divergence error $\int_\Omega |\nabla \cdot \vec{B}| d\Omega$ as a function of the simulation time for the Orszag-Tang vortex setup. Three runs with the DGFV4 are shown, where the Powell scheme and hyperbolic divergence cleaning are either on or off. The simulation without any divergence correction method activated is labeled as "none".

From Section 3.3 we know that the turbulent Mach number is a measure of flow compression, and since shocks are highly compressive flow structures, a certain correlation with the shock indicator respectively blending factor is expected. Figure 6.17 shows this is indeed the case. Plotted are the turbulent Mach number (left) given by (3.12) and the blending factor (right). Curiously, there is unusual blending activity in smooth flow regions where resolution level changes. Presumably, the smoothness estimator based on solution jumps at element boundaries, as discussed in Section 4.7.1, gets confused due to artificial jumps in the solution introduced by the restriction and prolongation procedures in the AMR routines of FLASH.

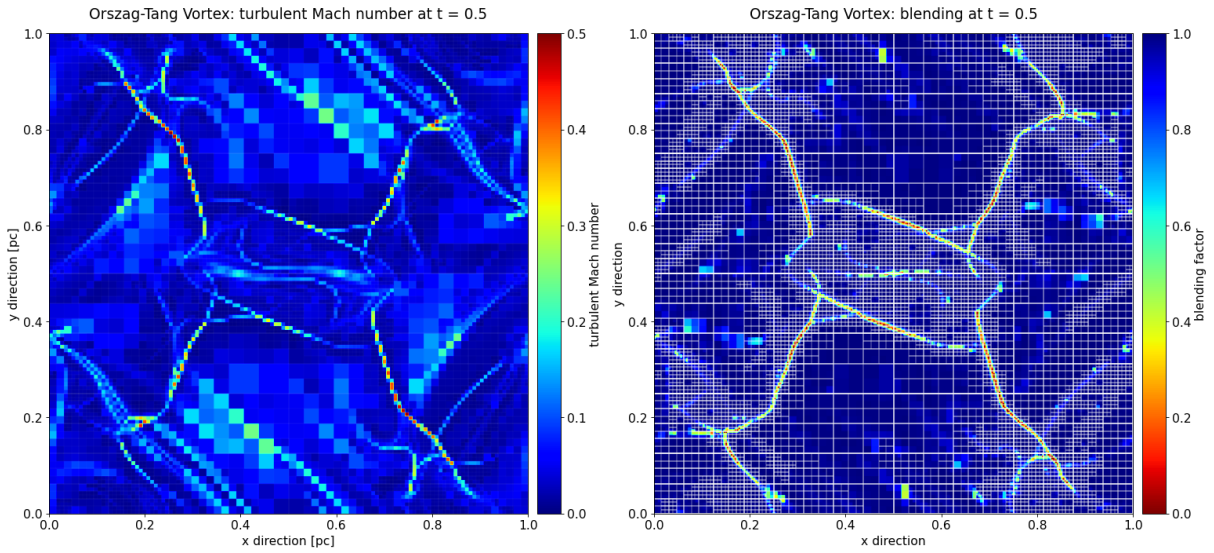

Figure 6.17: Turbulent Mach number (left) given by (3.12) and the blending factor (right) of the Orszag-Tang vortex setup computed with DGFV4 at final simulation time $t = 0.5$. The white lines in the right plot denote the grid lines.

We want to investigate the runtime costs of our DG implementation for a 3D MHD simulation in comparison to the reference solver Bouchut5, a well-tested, second order, split solver for ideal MHD also readily available in FLASH. Our setup of choice is the 3D extension given in Helzel et al. (2011) of the 2D Orszag-Tang vortex. It is directly attained by the following initial velocity:

$$\vec{v} = (-\sin(2\pi\, y)\,(1{+}0.2\,\sin(2\,\pi\, z)), \sin(2\pi\, x)\,(1{+}0.2\,\sin(2\,\pi\, z)), 1{+}0.2\,\sin(2\,\pi\, z))^T. \quad (6.7)$$

The rest of the initial parameters in the original 2D setup are copied along the z-axis. We simulate to final simulation time $T = 0.5$ ensuring a reasonable mix of FV and DG due to blending. We carry out four simulations in total comparing the single-core and multi-core performance of the two solvers. The single-core runs are conducted on the workstation (see Section 5.2) on a uniform grid with DOF of $64^3 \approx 2.6 \times 10^5$. The multi-core tests are carried out on CHEOPS (see Section 5.2) on a uniform grid with DOF of $128^3 \approx 2.1 \times 10^6$ and with 4×24 cores. The runtime results are given in Table 6.3 and Table 6.4. Our DG solver is about five to six times slower than Bouchut5 in both scenarios. Clearly, DGFV4 is at disadvantage with regards to completion times, since it consumes two to three times more timesteps, times four calls considering the four stages of the SSP-RK(4,3) scheme. If we take this into account and calculate the "raw" throughput of the schemes according to (5.3), we see that our scheme is actually quite efficient compared to

Bouchut5. This includes not only the computation of the fluxes and new solution states, but also smoothness estimation and blending.

Table 6.3: Runtimes of Bouchut5 and DGFV4 in FLASH for the 3D Orszag-Tang Vortex setup run with a single core on the workstation. The total amount of DOF is $64^3 \approx 2.6 \times 10^5$.

scheme	runtime [min]	slowdown	#steps	slowdown	TP [10^4 DOF/s]	speedup
Bouchut5	3.58	1.00	126×1	1.0	15.36	1.00
DGFV4	22.70	6.33	322×4	10.2	24.79	1.61

Table 6.4: Runtimes of Bouchut5 and DGFV4 in FLASH for the 3D Orszag-Tang Vortex setup run with 4×24 cores on CHEOPS. The total amount of DOF is $128^3 \approx 2.1 \times 10^6$.

scheme	runtime [min]	slowdown	#steps	slowdown	TP [10^4 DOF/s]	speedup
Bouchut5	3.71	1.00	250×1	1.0	2.45	1.00
DGFV4	18.88	5.09	656×4	10.5	5.06	2.07

The findings in Table 6.3 and Table 6.4 are also mirrored in scaling tests (see Section 5.4) on ODIN (see Section 5.2) with Bouchut5 and DGFV4 in FLASH. We computed the 3D Orszag-Tang Vortex till $T = 0.05$ with a fixed problem size of 256^3 DOF. Figure 6.18 shows that DGFV4 is more efficient than Bouchut5. And as the single-core performance results in Table 6.3 attest this efficiency gain is not manifestly rooted in excessive costs of isolated computations inside the solver. Instead, we identified three facets of Bouchut5, which makes it not so efficient. Firstly, the dimensional splitting of the blocks into sweeps along x-, y-, and z-directions decompactifies the data causing much more costly traffic in and out of CPUs. Secondly, every sweep in each of the three directions causes a full round of guard cell data exchange over MPI. And thirdly, Bouchut5 resorts to a parabolic diffusion method (Marder 1987) for divergence control entailing further data exchange over MPI. In contrast, DGFV4 only needs to update the guard cell layer once at the beginning of each RK stage and has then all the data it needs for computing the new solution.

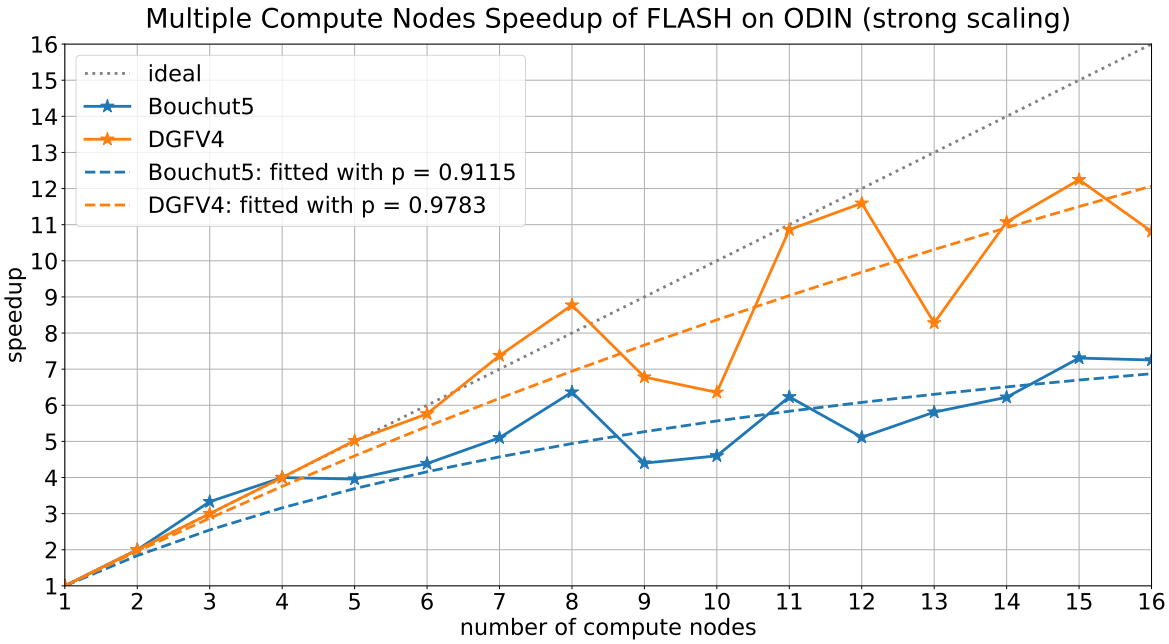

Figure 6.18: Speedup of Bouchut5 and DGFV4 in FLASH on multiple compute nodes on ODIN. The 3D Orszag-Tang Vortex was computed till $T = 0.05$ with a fixed problem size of 256^3 DOF. The fits (dashed lines) are determined with Ahmdal's law (5.4). The volatility of the runtime measurements on ODIN is discussed in Section 5.4.

Magnetic Rotor

We now investigate the 2D MHD rotor problem introduced by Balsara and Spicer (1999). In this setup, a dense disc of fluid rotates within a static fluid background, with a gradually declining velocity layer between the disk edge and the ambient fluid. An initially uniform magnetic field is present, winding up with the disc rotation and containing the dense rotating region through magnetic field tension. The computational domain is set to $\Omega = [0, 1]^2$. Initial pressure and magnetic fields are uniform in the whole domain, with $p = 1$ and $\vec{B} = (5/4\pi, 0, 0)^T$. The central, rotating disc is defined by $r < r_0$ where $r^2 = (x - 0.5)^2 + (y - 0.5)^2$, and $r_0 = 0.1$. Inside the disc, $\rho = 10$, and the disc rotates rigidly with $\vec{v} = (0.5 - y, x - 0.5)v_0/r_0$ with $v_0 = 2$. The background fluid has a density of $\rho = 1$ and is at rest: $\vec{v} = \vec{0}$. In the annulus $r_0 \leq r \leq r_1 = 0.115$, the transition region linearly interpolates between the disc and the background, with $\vec{v} = (0.5-y, x-0.5)v_0 f/r_0$ and $\rho = 1 + 9f$, where $f = (r_1 - r)/(r_1 - r_0)$ is the transition function. The simulation runs until the final time $T = 0.15$. We use outflow boundary conditions.

We present the density contours at $T = 0.15$ in Figure 6.19. The black lines in the left

half the plot highlight the AMR grid bounded between refinement levels of 64^2 cells to 1024^2 cells. The contours of the evolved disc show a sharp and almost noise-free picture of the circular rotation pattern, which is in general a challenge for MHD codes.

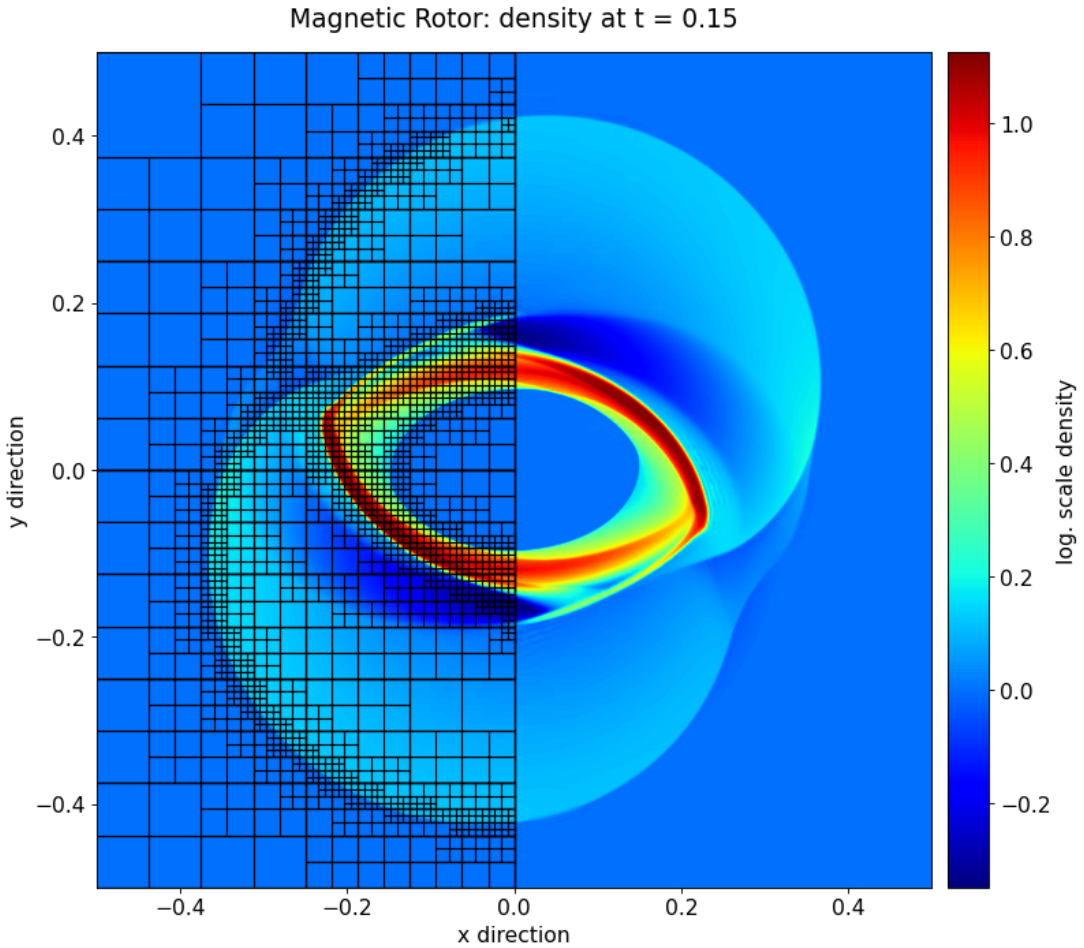

Figure 6.19: Density contours in logarithmic scale of the magnetic rotor at final time $T = 0.15$ calculated with DGFV4 on an dynamic AMR mesh (shown as black lines in the left half of the plane) with a maximum refinement level of $l = 8$ which is equivalent to a maximum resolution of 1024^2 cells.

In Figure 6.20, we plot the local magnetic field divergence error $|\nabla \cdot \vec{B}|$ of the DGFV4 solver at the final simulation time. The error is mostly concentrated in regions around shocks and radially propagates away in all directions (circular ripples) due to the hyperbolic divergence advection mechanism. Preferably, the divergence error gets advected out of the domain but, as clearly visible in the divergence plot, the errors can accumulate in stagnant regions of the domain, which justifies the importance of the damping source term (3.27) as an additional mechanism to dispose of any accrued magnetic field divergence

errors.

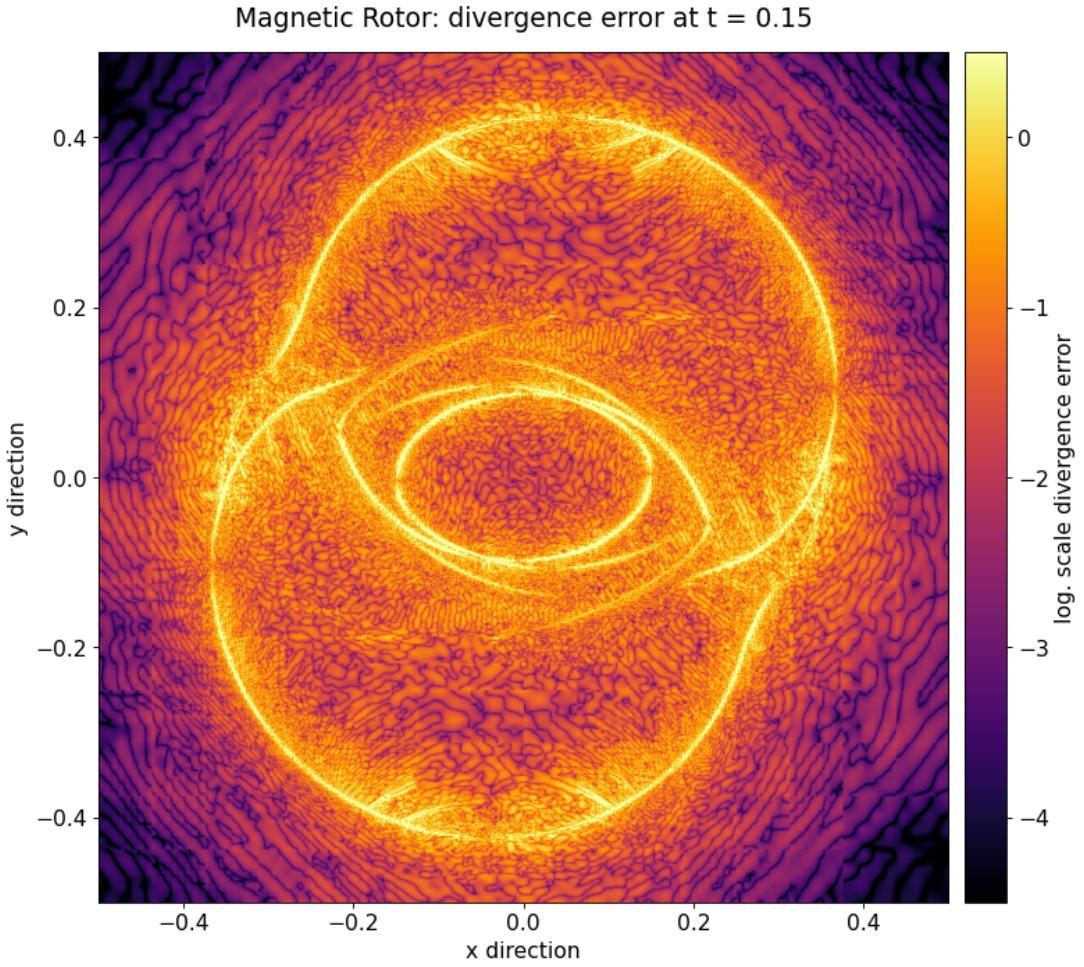

Figure 6.20: Contours of the local divergence error $|\nabla \cdot \vec{B}|$ in logarithmic scale of the magnetic rotor run by the DGFV4 scheme at final time $T = 0.15$.

Analogously to the Orszag-Tang vortex setup above, we plot the turbulent Mach number (left) and the blending factor (right) in Figure 6.21. Besides a few misfirings at non-conforming interfaces, which we discussed above, the blending works as expected. The spurious area of seemingly elevated compressive turbulence in the center (yellow "cross" in the left plot) is a result of low resolution and a rapidly expanding flow causing an excess of positive velocity divergence. We conclude that the way we measure the turbulent Mach number according to (3.12) can yield fallacious results and does not necessarily indicate flow structures, which are troubling for high order methods, like our DG scheme. Nevertheless, we deem it as one useful part in the tool box to track down challenging flow regions.

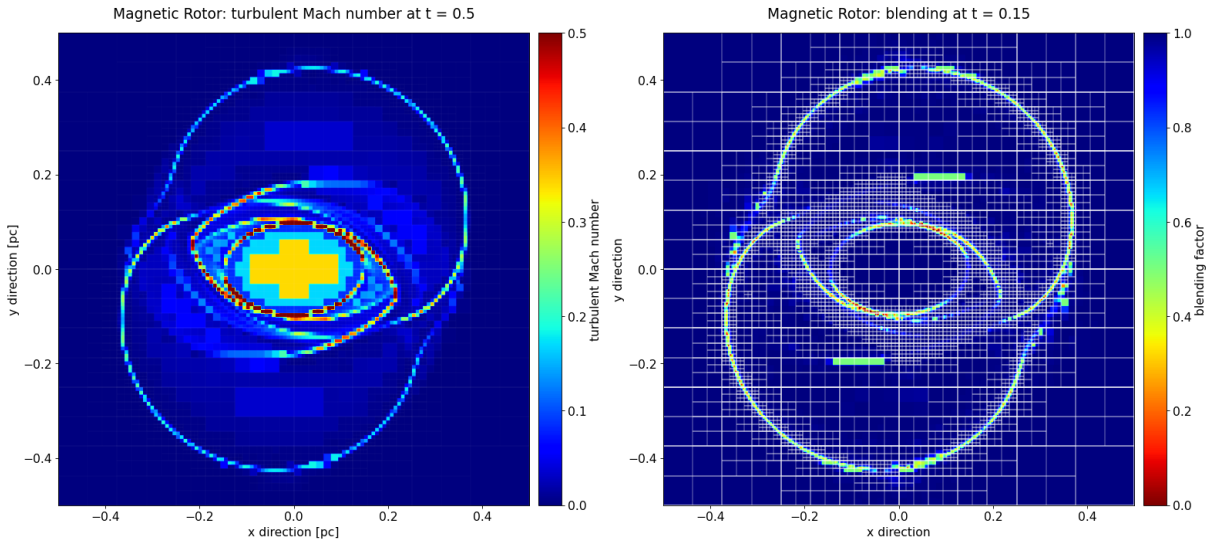

Figure 6.21: Turbulent Mach number (left) given by (3.12) and the blending factor (right) of the magnetic rotor setup computed with DGFV4 at final simulation time $T = 0.15$. The white lines in the right plot denote the grid lines.

MHD Blast

In order to test the shock-capturing performance of our code for a very strong ideal MHD shock problem in 3D, we utilize the 3D blast wave setup of Balsara et al. (2009). The computational domain is set to $\Omega = [0, 1]^3$ with outflow boundary conditions. The background fluid is initially at rest with respect to the grid, where $\vec{v} = \vec{0}$, $\rho = 1$, and with a uniform magnetic field $\vec{B} = (100\sqrt{3}, 100\sqrt{3}, 0)$. The ambient pressure is set to $p = 10^{-1}$, and within a central sphere of radius $r_0 = 0.1$, we set $p = 10^3$ to initialize the blast. This creates an extreme initial shock strength with a pressure ratio of 10^4 in a strongly magnetized background with a plasma-β of $\approx 10^{-5}$. We take $\gamma = 1.4$, and run the simulation until the final simulation time $T = 0.01$. The AMR grid is bounded between refinement levels of 64^3 cells to 256^3 cells.

In Figure 6.22, we show a slice at $z = 0$ of the magnetic pressure at the final time. The black lines in the left half of the plot highlight the AMR grid, while the light gray arrows trace the magnetic field lines of the solution. The originally spherical blast bubble gets stretched along the magnet background field over the course of the simulation as is described by Balsara et al. (2009). Our scheme is able to maintain positivity of the pressure and density in the whole domain and correctly captures the very strong discontinuities, while resolving the complex structures within the blast shell. Note that no oscillations are visible around discontinuities. This test shows the robustness and shock-capturing

performance of our blending scheme for three-dimensional problems involving very strong magnetized shocks.

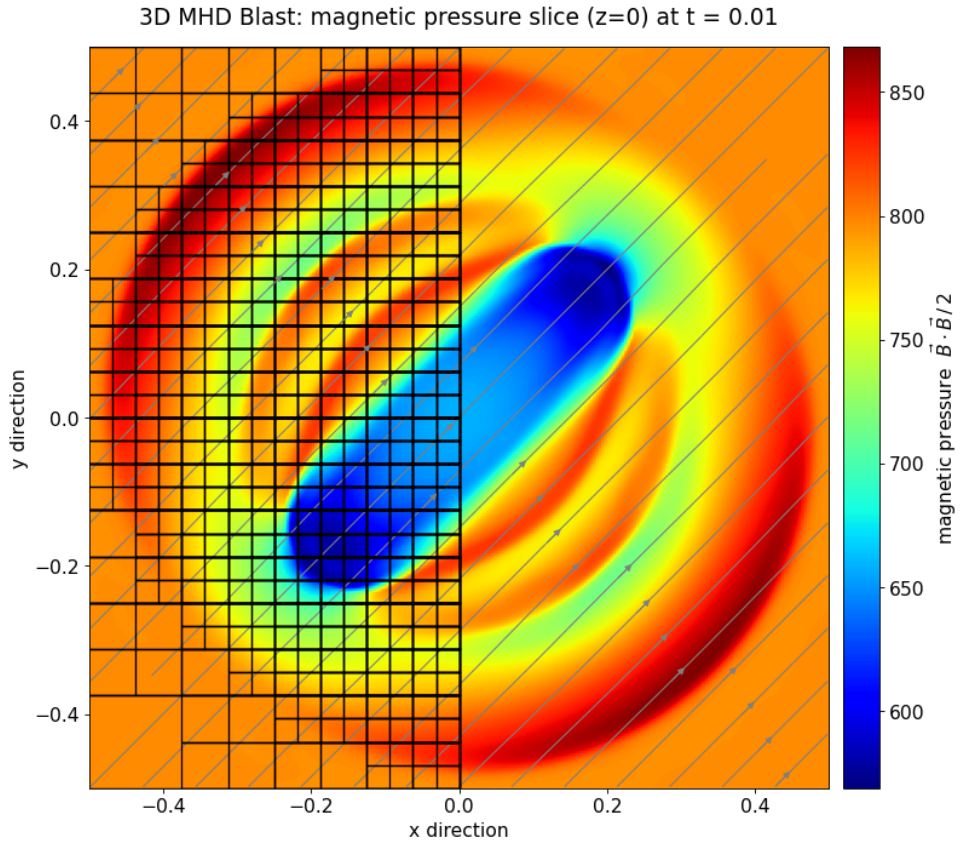

Figure 6.22: Slice of the magnetic pressure at $z = 0$ for the MHD blast wave test solved with the DGFV4 scheme. The black lines on the left side highlight part of the AMR mesh and the gray stream lines indicate the field spanned by the x and y component of the magnetic field.

In Figure 6.23 we plot the turbulent Mach number (left) and the blending factor (right) and confirm the correct functioning of the blending for challenging 3D setups.

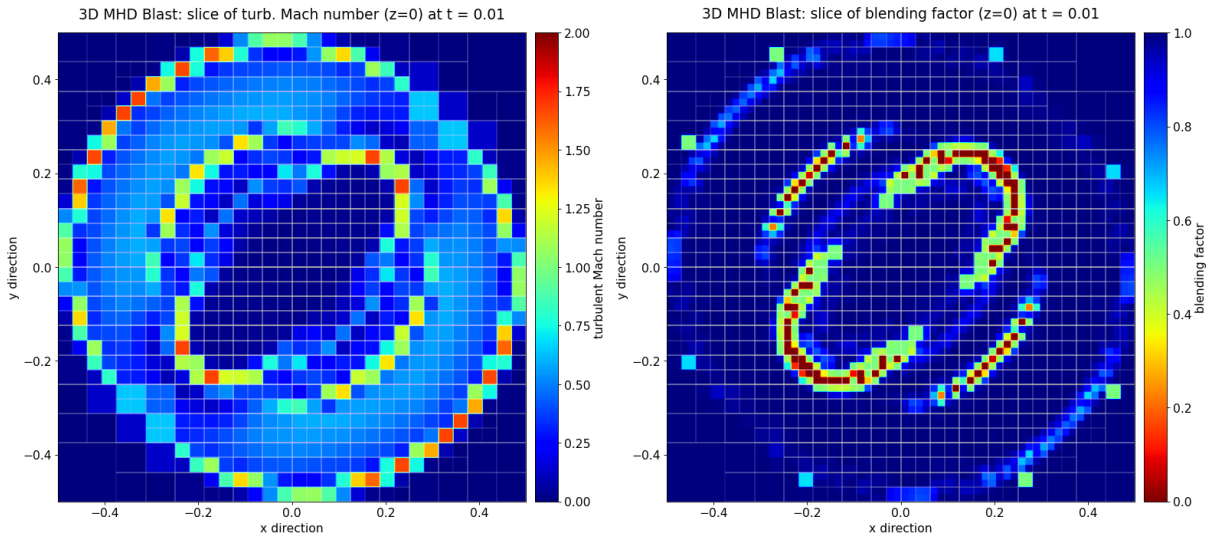

Figure 6.23: Turbulent Mach number (left) given by (3.12) and the blending factor (right) of the 3D MHD blast setup computed with DGFV4 at final simulation time $T = 0.01$. The white lines denote the grid lines.

In Table 6.5 we report the total runtimes of the 3D blast setup simulated with Bouchut5 and DGFV4. The computations are conducted on CHEOPS with 4×24 cores and with AMR. Thus, the problem size is not fixed throughout the simulation, but dynamically grows in alignment with the expanding shock wave. The DOF in both runs increased at equal rate. This allows us to compute an effective throughput $\mathrm{TP}_{\mathrm{eff.}}$ according to (5.3) with an effective $\mathrm{DOF}_{\mathrm{eff.}}$ determined by the maximally allowed resolution level, which in our case amounts to $256^3 \approx 16.8 \times 10^6$. From Table 6.5 we learn that DGFV4 is over two times more efficient than Bouchut5 in terms of effective throughput. This is consistent with the benchmark results presented in Table 6.3 and Table 6.4.

Table 6.5: Runtimes of Bouchut5 and DGFV4 in FLASH for the 3D MHD blast setup run with 4×24 cores on CHEOPS. The computations are conducted with AMR and the effective $\mathrm{DOF}_{\mathrm{eff.}}$ is $256^3 \approx 16.8 \times 10^6$. Details can be found in the text.

scheme	runtime [min]	slowdown	#steps	slowdown	$\mathrm{TP}_{\mathrm{eff.}}$ [10^4 DOF/s]	eff. speedup
Bouchut5	19.25	1.0	358×1	1.0	5.41	1.0
DGFV4	65.40	3.4	786×4	8.8	14.00	2.6

6.3.4 Coupling to Gravity

The FLASH framework provides specialized units for the employment of static gravitational sources emminating from point masses distributed at arbitrary points within the

computational domains or from external background fields. Furthermore, FLASH unites a variety of solvers for the Poisson equations of gravity for mass distributions (self-gravity) based on multipole expansions (Müller and Steinmetz 1995), tree-based Barnes-Hut algorithms (BHTree) (Barnes and Hut 1986), multigrid methods (Ricker 2008) or hybridized schemes (Pfft) (Daley et al. 2012).

In this work, we focus on a recent implementation of the Barnes-Hut algorithm in FLASH for versions 4.0 or later (BHTree/Wunsch) developed and released by (Wünsch et al. 2018). The module utilizes a MPI-parallelized octtree algorithm placed on top of the AMR PARAMESH library (Olson et al. 1999). Following the classical Barnes-Hut procedure, the algorithm computes the local multipole expansion coefficients encoding the gravitational pull of a specific section of the computational domain and communicates to the different processors only those parts of the tree that are needed to perform the tree walk in accordance with the multipole acceptance criterion (MAC). The advantage of this approach is a relatively low memory footprint and good scaling properties. Furthermore, this particular tree code features a general tree-based radiation transport algorithm (TreeRay), which we discuss in the next section. Boundary conditions for gravity can be either isolated or periodic, and they can be specified in each direction independently, using a novel generalization of the Ewald method also introduced in Wünsch et al. (2018).

The Evrard test described by Evrard (1988) investigates the gravitational collapse and subsequent re-bounce of an adiabatic, initially cold sphere. It is generally used to verify energy conservation when hydrodynamics schemes are coupled to gravity (Springel et al. 2001; Wetzstein et al. 2009). The initial conditions consist of a gaseous sphere of mass M, radius R and density profile

$$\rho(r) = \frac{M}{2\pi\, R^2\, r}.$$

The uniform temperature is initialized so that the internal energy per unit mass is

$$E_{\text{int}} = 0.05\, \frac{G\, M}{R}$$

where G is the gravitational constant. The standard values of the above parameters, used also in this work, are $M = R = G = 1$. The Barnes-Hut MAC is set to $\theta_{\text{lim}} = 0.5$ and the AMR refinement levels range from 64^3 cells to 256^3 cells. We computed the reference solution on a uniform grid of 256^3 cells with the PPM solver for hydrodynamics (Colella and Woodward 1984) and the same tree solver for gravity with equal settings as presented

in Wünsch et al. (2018).

Figure 6.24 presents the time evolution of the domain-integrated gravitational, kinetic, internal, and the sum of the three energies. Our results match the reference very well, which confirms the correct coupling to the gravity solver. Since the interface to the gravitational source terms in FLASH is completely generic our findings transfer to any other gravity solver module available in FLASH. Note that the reason for a deviation from energy conservation has been discussed extensively in Wünsch et al. (2018). The source of the inaccuracy is the finite grid resolution, which leads to an error at the point in time where the sphere is most compressed (least resolved) and continues to bounce back.

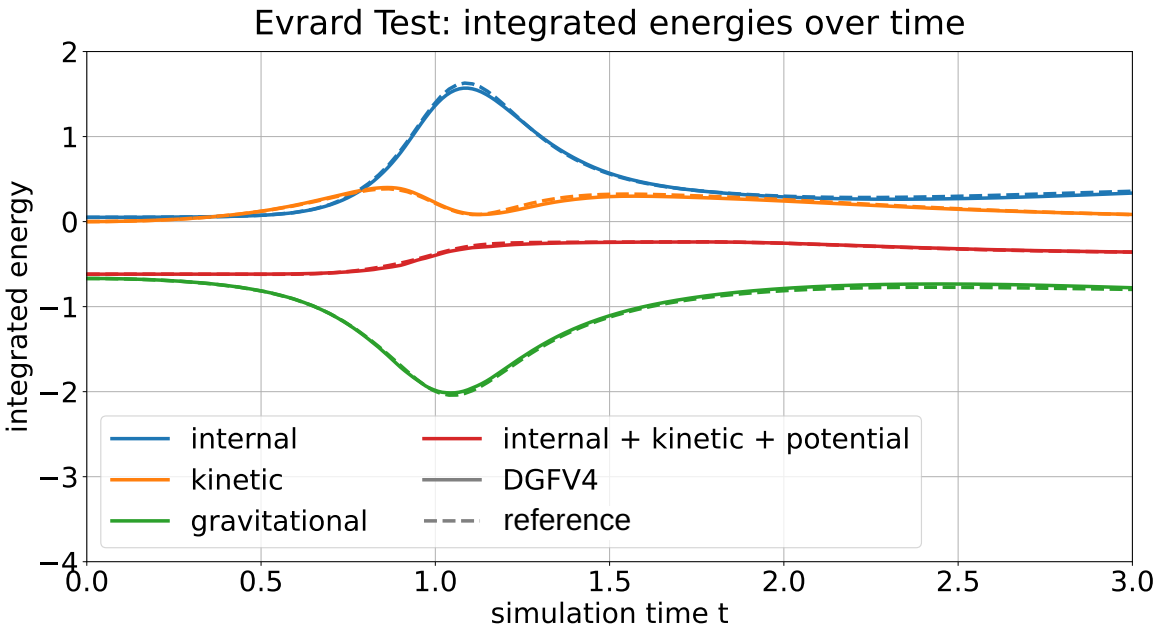

Figure 6.24: Domain-integrated energies over simulation time for the Evrard test run by our DGFV4 solver. The dashed lines show the reference solution computed with PPM on a uniform grid.

6.3.5 Coupling to TreeRay

Turbulent, multi-phase structures within the interstellar medium (ISM) is shaped by the complex and nonlinear interplay between gravity, magnetic fields, heating and cooling, and the radiation and momentum input from stars (Agertz et al. 2013; Walch et al. 2015; Kim et al. 2017). However, treatment of radiative transfer with multiple radiation sources is a critical challenge in moving towards a proper description of star formation and the

complete modeling of interstellar media in general.

TreeRay (Wünsch et al. 2021) is a new radiation transport method combining an octree (Wünsch et al. 2018) with reverse ray tracing implemented in FLASH. Sources of radiation and radiation absorbing gas are mapped on to the tree encoded as emission and absorption coefficients. The tree is traversed for each grid cell and tree nodes are mapped on to rays going in all directions. Finally, a one-dimensional radiation transport equation is solved along each ray. Several physical processes are implemented into the code by providing prescriptions for the absorption and the emission coefficients.

In this work, we employ a simple on-the-spot approximation with only one source emitting a constant number of extremely ultraviolet (EUV) photons per unit time. The Spitzer sphere (Spitzer 1978) is a simple model regarding the interaction of ionizing radiation with absorbing gases. In this model, the EUV radiation from a young, massive star ionizes and heats the surrounding medium, creating a so-called HII region, that is an over-pressured, expanding bubble of photo-ionized, hot gas bounded by a sharp ionization front (Whitworth 1979; Deharveng et al. 2008). The expanding ionization front drives a shock into the surrounding neutral, cold gas, sweeping it up into a dense, warm shell.

Codes of radiation transport methods use this spherical expansion of a HII region as a standard test problem to validate their coupled photoionization and hydrodynamics algorithms. Bisbas et al. (2015) introduced a standard benchmark test in 3D for early-time ($t \leq 0.05$ Myr) and late-time ($t > 0.05$ Myr) expansion phases of the process. In this work, we test our code with the 3D late expansion phase simulation, which we name the Starbench test. The problem is insofar challenging since it involves an intricate combination of fluid dynamics, radiative transfer, microphysical heating, cooling, ionization and recombination.

An analytic approximation for the time evolution of the ionization front (IF) is given by Spitzer (1978)

$$R_{\mathrm{Spitzer}}(t) = R_{\mathrm{S}} \left(1 + \frac{7\, c_{\mathrm{ionized}}\, t}{4\, R_{\mathrm{S}}}\right)^{4/7}, \tag{6.8}$$

where

$$R_{\mathrm{S}} = \left(1 + \frac{3\, \dot{N}\, m_p^2}{4\pi\, \alpha_B\, \rho_0^2}\right)^{1/3} \tag{6.9}$$

is the Strömgren radius (Strömgren 1939), $c_{\mathrm{ionized}} = 12.85\,\mathrm{km\,s^{-1}}$ is the sound speed in the ionized gas inside the isothermal bubble of $T_{\mathrm{ionized}} = 10^4\,\mathrm{K}$. $\dot{N} = 10^{49}\,\mathrm{s^{-1}}$ is the

rate at which the source at the center emits hydrogen ionizing photons ($E_\nu > 13.6\text{eV}$), $m_p = 1.67 \times 10^{24}$ g is the proton mass, and $\alpha_B = 2.7 \times 10^{-13}\text{cm}^3\text{s}^{-1}$ is the recombination coefficient. The density of the surrounding neutral cloud of only hydrogen gas ($\gamma = 5/3$) is taken to be $\rho_0 = 5.21 \times 10^{-21}\text{g}\,\text{cm}^{-3}$ and has a temperature of $T_{\text{neutral}} = 10^3\,\text{K}$. The corresponding sound speed is then $c_{\text{neutral}} = 2.87\,\text{km}\,\text{s}^{-1}$. If, during the simulation, the temperature in the neutral gas exceeds $T_{\text{shell}} = 300\,\text{K}$, it is instantaneously cooled to T_{shell}. Consequently, the shell of shock-compressed neutral gas immediately ahead of the expanding IF is effectively isothermal. The cooling limits the thickness of the shell and makes it resolvable for the numerical hydrodynamics method. The given parameters result in a Strömgren radius of $R_{\text{S}} = 0.3141\,\text{pc}$.

Since the Spitzer solution (6.8) is only valid for the very early expansion phase, Bisbas et al. (2015) proposed a heuristic equation (equation (28) in Bisbas et al. (2015)) giving a good approximation of the position of the ionization front at all times

$$R_{\text{Starbench}} = R_{\text{II}} + \Big(1 - 0.733\,\exp(-t/\text{Myr})\Big)\,(R_{\text{I}} - R_{\text{II}}) \tag{6.10}$$

where R_{I} and R_{II} are solutions to the ordinary differential equations (8) and (12) in Bisbas et al. (2015). For brevity we omit the explicit definition of these and refer to the paper.

We run our simulation to final time $T = 1.5\,\text{Myr}$ and set the domain to $\Omega = [0, 3\,\text{pc}]^3$, which only represents one octant of the whole setup. Since the model is spherically symmetric around the emitting radiation source situated at the $\vec{x} = (0, 0, 0)^T$, we can speedup the simulation considerably. The resolution ranges from 32^3 cells to 128^3 cells. The boundaries of the domain touching the coordinate origin are reflecting walls while the rest are set to outflow. The ionized bubble is expected to occupy most of the domain by the end of the simulation.

Figure 6.25 is a snapshot of the expanding bubble at time $t = 0.8\,\text{Myr}$ showing a density slice at constant $z = 0$. The dark violet region of lower density is completely filled with hot gas, photo-ionized by the radiation emitting point source sitting in the lower left corner. The dashed white circle demarks the ionization front separating the inner, ionized medium from the surrounding, neutral medium of higher density. The dashed circle is computed with the Starbench solution (6.10) confirming excellent agreement between theoretical and numerical results.

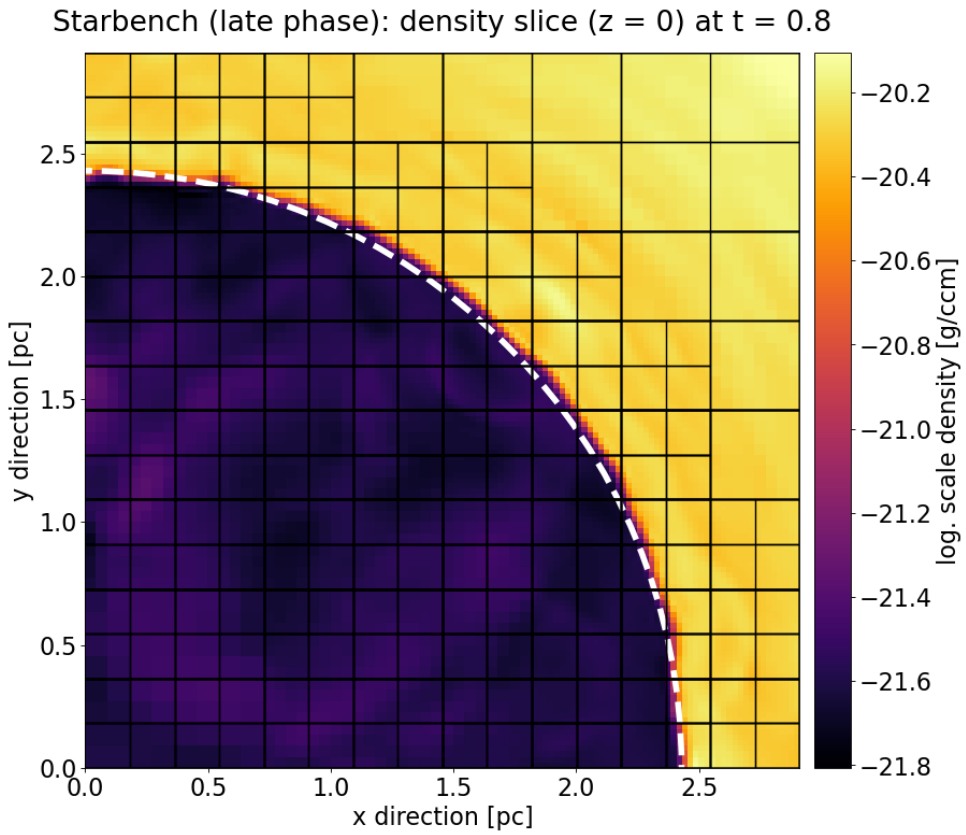

Figure 6.25: Density slice of the Starbench test setup at simulation time $t = 0.8$ Myr computed with our DGFV4 solver and a maximum resolution of 256^3 cells. The dashed circle highlights the Starbench solution given by (6.10). The black lines depict the AMR grid.

Additionally, we plotted the position of the ionization front over the course of our simulation, which we retrieved from snapshots taken at regular time intervals. The position of the IF is determined by calculating the mean radius, where the shell-averaged ionized medium drops to 50 %, signaling the rapid transition to the neutral medium. The result is shown in Figure 6.26 together with the Starbench solution (6.10) (black solid line) and the Spitzer solution (6.8) (gray dashed line). Clearly, our numerical solution matches the Spitzer solution only at very early times, but there is good agreement to the Starbench solution throughout the whole simulation. We ascribe the deviation observable from $t = 1.1\,\mathrm{Myr}$ on-wards to the bubble nearing the boundaries of the domain and the still rather low resolution. Investigations with identical parameters but different solvers in FLASH, such as PPM, revealed the exact same behavior, hence the phenomenon is not rooted in our fluid solver.

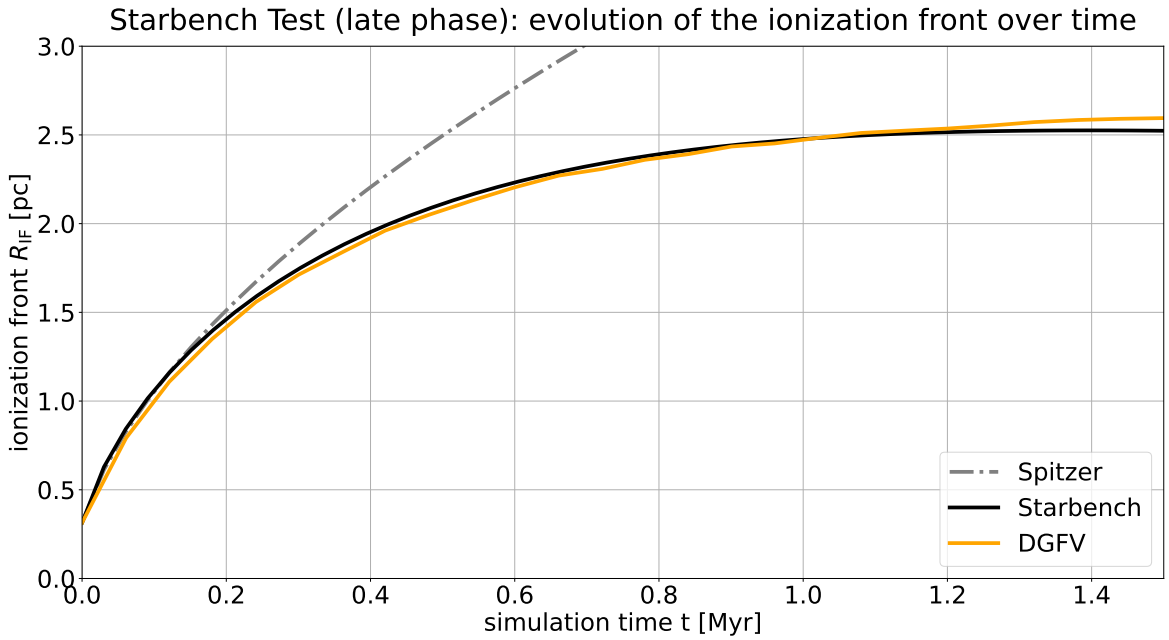

Figure 6.26: Ionization fronts over time computed with the DGFV4 solver. For reference the Spitzer solution (6.8) and the Starbench solution (6.10) are given.

From a technical point of view, this particular test setup is remarkable, since it unites a large number of numerical components into one simulation. The complex interplay of physics and chemistry, especially the correct handling of a very sharp transition of a two-component (ionized/neutral) medium intertwined with ionization, recombination, heating and cooling is a major challenge for every fluid dynamics code.

6.4 Final Remarks

Considering the previous comments in this chapter, whenever we do multi-physics simulation with AMR, gravity, chemistry, etc. in FLASH we are limited to at most formally second order accuracy; even though our fluid solver in its own right defaults to fourth order accuracy in space. In order to take full advantage of the potential benefits of higher order methods in general, all steps in the simulation chain must be designed and executed with high order in mind. According to the authors, this has been done, for example, in the high order DG FLEXI code (Krais et al. 2020). However, as the saying goes “Rome wasn’t built in a day”, our primary goal was to get DG up and running within a comprehensive astrophysics framework. After achieving the first milestone, a gradual shift

towards “pure” high order simulations might be feasible. Clearly, this is only possible with direct feedback by and in close collaboration with the astrophysics community.

Chapter 7

Simulations

7.1 Introduction

In this chapter, we present a number of interesting astrophysical applications, where we investigate the strengths and weaknesses of our DGFV4 solver in FLASH. We seek to learn under what conditions the DG scheme yields superior results and when it underperforms. We show four exemplary astrophysics simulations each with different multi-physics aspects and varying demands on the fluid solver.

Since the standard solver for MHD in FLASH, an unsplit, staggered mesh solver (Lee 2013), is not robust enough for these kind of simulations, we compare our results with the seasoned fluid solvers Bouchut5 (Bouchut et al. 2010), a well-tested, second order, split solver for ideal MHD also readily available in FLASH. For the sake of clarity and to keep this chapter succinct and compact, the reference results are presented in the appendix of this thesis. Respective pointers are given in the text accordingly. As time integrator in DGFV4 we use the third order, four stages SSP-RK(4,3), the same as in the test cases in the chapter before.

7.2 Young Supernova Remnant

The setup is borrowed from Markert et al. (2021), where a simplified model of the Tycho supernova is described. The Type Ia stellar explosion is named after the famous Danish astronomer Tycho Brahe, as he was first to officially report the supernova in the year 1572.

Nowadays, we know that this object, identified as SN 1572, is located in our Galaxy, is over 10,000 lyr away from Earth and has a diameter of 10 pc. Because of its proximity and enormous luminosity, the supernova was so bright that it could be seen with the naked eye at daytime (Rest et al. 2008). In addition to actual astronomical observations, the results in Markert et al. (2021) serve as our reference, since the underlying physics and the setup parameters discussed in this section are identical. In the following, we replicate the setup description with minor adaptations related to our DG solver in FLASH.

Supernova models have been analyzed and discussed for many decades and since they unite a broad range of features such as strong shocks, instabilities and turbulence, they resemble a good test bed for our novel shock capturing approach in combination with AMR. The general sequence of events of the presented supernova simulation is like this: We start with a constant distribution of very low density resembling interstellar media (ISM) that typically fills the space between stars. When a star explodes by turning into a supernova it ejects its own mass at very high speeds into the ISM preceded by a strong shock front heating up the ISM. The ejected mass is rapidly decelerated by the swept-up ISM giving rise to a so-called reverse shock that travels backwards into the cool inner region. The interface, or more precisely the contact discontinuity, between shocked ejecta and shocked ISM is unstable and leads to a layer of slowly growing Rayleigh-Taylor instabilities. This gradually expanding layer, called supernova remnant, is of special interest, since this is where astronomical observations reveal a lot of ongoing physics and chemistry, especially driven by mixing and turbulence.

We adapt the setup descriptions in Chevalier (1982); Fraschetti et al. (2010); Ferrand et al. (2012), where we have the initial (internal) blast energy E and the ejecta mass M given in cgs (centimeter-gram-seconds) units. It is beneficial to convert the given units to convenient simulation units reflecting characteristic dimensions of the physical model at hand. Table 7.1 lists the conversion between cgs and simulation units and table 7.2 lists the initial parameters used in this simulation. The ambient density ρ_a is related to

Table 7.1: Conversion from cgs units to simulation units.

quantity	cgs units	simulation units
mass	g	$\mathrm{M}_\odot = 1.989 \times 10^{33}\,\mathrm{g}$
time	s	$\mathrm{yr} = 3.154 \times 10^{7}\,\mathrm{s}$
length	cm	$\mathrm{pc} = 3.086 \times 10^{18}\,\mathrm{cm}$
temperature	K	K

Table 7.2: Hydrodynamical parameters in cgs units and simulation units.

description	cgs units	simulation units
domain size	$L = 1.543 \times 10^{19}\,\mathrm{cm}$	$L = 5\,\mathrm{pc}$
blast energy	$E = 10^{51}\mathrm{erg}$	$E = 5.2516 \times 10^{-5}\,\mathrm{M}_\odot \frac{\mathrm{pc}^2}{\mathrm{yr}^2}$
ejecta mass	$M = 2.7846 \times 10^{33}\,\mathrm{g}$	$M = 1.4\,\mathrm{M}_\odot$
ambient density	$n_H = 0.13\,\mathrm{cm}^{-3}$	$\rho_a = 2.4539 \times 10^{-3}\,\frac{\mathrm{M}_\odot}{\mathrm{pc}^3}$
ambient temperature	$T = 10^4\,\mathrm{K}$	$p_a = 2.1309 \times 10^{-13}\,\frac{\mathrm{M}_\odot}{\mathrm{pc\,yr}^2}$

the mono-atomic particle (hydrogen) density n_H via $\rho_a = m_u\, n_H$, where m_u is $\frac{1}{12}$ of the mass of a carbon-12 atom. The ambient pressure p_a is calculated from the ideal gas law, i.e. $p_a = n_H\, k_B\, T$ with the Boltzmann constant k_B. There is some ambiguity regarding the ambient gas temperature T. The literature mentions a warm and neutral interstellar medium which is attributed to temperatures between $6 \times 10^3\,\mathrm{K}$ and $10^4\,\mathrm{K}$. The heat capacity ratio is fixed to $\gamma = 5/3$. The simulation time spans a period from $t_0 = 10\,\mathrm{yr}$ to $T = 500\,\mathrm{yr}$. The expansion of the forward shock (eq. (7.3)) is then expected to approximately reach $R_{\mathrm{FS}} = 5\,\mathrm{pc}$, which determines the size of the computational domain $L := 5\,\mathrm{pc}$. Fig. 7.1 depicts a schematic of the simulation setup. Due to the rotational symmetry of the setup it is sufficient to simulate just one octant of the supernova. The

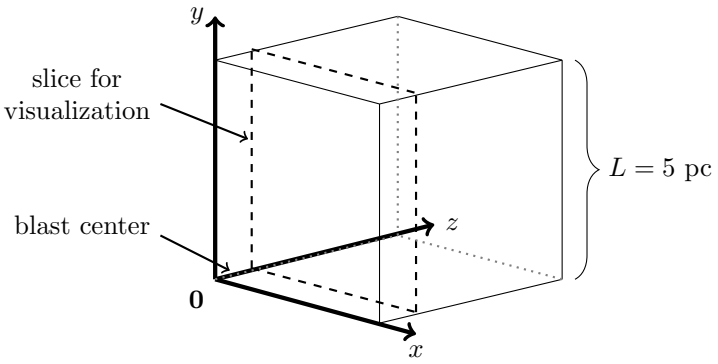

Figure 7.1: Computational domain (cubic box) covering one octant of the supernova model. The faces at the coordinate axes are set to reflecting walls while the opposite sides are set to outflow.

following formulas have been derived in Chevalier (1982) and were adapted to the current setup, i.e. power-law indices of $(s, n) = (0, 7)$. The self-similar solution at initial time $t_0 = 10\,\mathrm{yr}$ within the power law region, respectively the blast center, is given by

$$r(t) = t\,\sqrt{\frac{5}{3}\,\frac{E}{M}} \quad \text{and} \quad \rho(t) = \frac{25}{21\,\pi}\,\frac{E^2}{M}\,t^4\,r(t)^{-7}. \tag{7.1}$$

We initialize the density as

$$\rho_0(\vec{x}) = \rho_a + \rho(t_0) \cdot \begin{cases} 1, & |\vec{x}| \leq r(t_0), \\ \left(\dfrac{|\vec{x}|}{r(t_0)}\right)^{-7}, & |\vec{x}| > r(t_0), \end{cases} \tag{7.2}$$

and the total energy with (6.6), where $p_0 = p_a$, $d = 3$ and $\sigma = \frac{3}{4} r(t_0)$. The initial momentum is $(\rho\,\vec{v})_0 = \vec{0}$. Since we are only interested in the evolution of the instability layer of the supernova, we apply the following rules for mesh refinement and coarsening. The expansion radius of the forward shock over time is given by

$$R_{\mathrm{FS}}(t) = 1.06 \left(\frac{E^2}{M\,\rho_a}\right)^{1/7} t^{4/7}. \tag{7.3}$$

This allows us to assign an adaptive, high resolution shell of maximally refined elements following the remnant as it expands into the computational domain. The inner and outer radii of the shell are estimated as

$$R_{\mathrm{inner}}(t) = 0.7\,R_{\mathrm{FS}}(t) \quad \text{and} \quad R_{\mathrm{outer}}(t) = 1.15\,R_{\mathrm{FS}}(t), \tag{7.4}$$

which have been found adequate via numerical experimentation. Moreover, up to $t = 200\,\mathrm{yr}$ we enforce $R_{\mathrm{inner}} = 0$, which ensures that the first phase of the explosion is well resolved in any case. The refinement levels range from 3 to 7 (FLASH convention), which translates to a FV equivalent resolution from $2^2 \cdot 8 = 16$ up to $2^6 \cdot 8 = 512$ cells in each spatial direction.

We performed a purely hydrodynamical simulation with our 3D (single-level) blending scheme DGFV4 in FLASH, which took 24.26 hours on ODIN with 12×16 cores. Figure 7.2 shows slices of the density (top) and the mass tracer field (or mass scalars) (bottom) at final simulation time of $T = 500\,\mathrm{yr}$. Again, mass scalars are advected alongside the density and are a versatile tool for tracing different mass quantities. For example, heavier elements such as metals are usually swept up by supernova explosions and their distribution gives insights into the intricate structure of supernova remnants. The exact position of the slices shown here are also sketched in Figure 7.1. The shock front partially left the domain, which is not considered a problem since the region of interest, namely the instability layer, is still properly covered. The noise in the density near the plot axes is caused by the "carbuncle" phenomenon, a well-known issue for many numerical schemes at grid-aligned

shocks and discussed, e.g., in Quirk (1997); Pandolfi and D'Ambrosio (2001); Dumbser et al. (2004). The areas of no interest, i.e. the hot outer region as well as the cold inner region, are by default only coarsely resolved by the AMR mesh. Figure A.3 in the appendix shows reference results by the blended multi-level DG schemes (Markert et al. 2021) implemented in NEMO with targeted maximum orders of two (DGFV2), four (DGFV4) and eight (DGFV8). Clearly, an increase in accuracy order leads to a much more detailed remnant structure emphasizing the advantage of higher order schemes in resolving small scale turbulence driven by Rayleigh-Taylor instability. For the most part, the blended fourth order DGFV4 implementations in NEMO and in FLASH are very similar, hence they achieve comparable ranges of scales in their respective solutions. Second order schemes are at disadvantage for this kind of setups considering the smeared out results of DGFV2 in Figure A.3. We note that Bochut5's solutions are very similar to DGFV2. We omitted Bouchut5's results, since they give no further insights.

The blending factor is shown in the top row of Figure 7.3. The band of highly refined elements is clearly visible following the remnant as intended. Two distinctive lines of blending activity trace the front and reverse shocks while there is only minor activity in the remnant effectively stabilizing the DG scheme. The bottom plot in Figure 7.3 shows the slice of the turbulent Mach number (3.12) justifying the correct blending activity of our scheme. At shocks the velocity divergence is always very strong causing high turbulent Mach numbers. The remnant, on the other hand, is dominated by subsonic turbulence at the boundary between incompressible and compressible flow ($\mathcal{T} \approx 0.3$ according to Section 3.3) with small patches of higher compressibility ($\mathcal{T} \approx 0.5$). The good performance of higher order DG in these turbulent lower-mach regimes has already been extensively discussed in the literature, e.g. Gassner and Beck (2013); Beck et al. (2014); Garai et al. (2015); Bauer et al. (2016); Flad and Gassner (2017). We ascribe the spuriously high values in turbulent Mach number and the intensified blending activity in the inner regions to the extremely under-resolved solutions. Nevertheless, the under-resolution near the center does not negatively impact the dynamics in the remnant. Hence, we see no reason to increase the resolution there and sacrifice execution speed.

Figure 7.4 shows a section of an actual false color image (left) of the Tycho supernova remnant shot by Eriksen (2011) and the Chandra X-ray Observatory operated by NASA. Low-energy X-rays (red) in the image show expanding debris (remnant) from the explosion and high energy X-rays (blue) show the forward shock. The right side in Figure 7.4 depicts the column sum of density Schlieren of our simulation with DGFV4 in FLASH.

Considering that SN 1572 is roughly 450 yr old, we took the snapshot at respective simulation time of 450 yr and montaged the astronomical observation side by side with the density schlieren plot. The positions of the forward shocks match nicely, indicating that the model despite its simplicity already describes the dynamical processes reasonably well. The remnant of the SN 1572 seems to cover a much larger region; even penetrates the forward shock. Presumably, the resolution in our simulation is not high enough to allow for sufficient growth of the remnant.

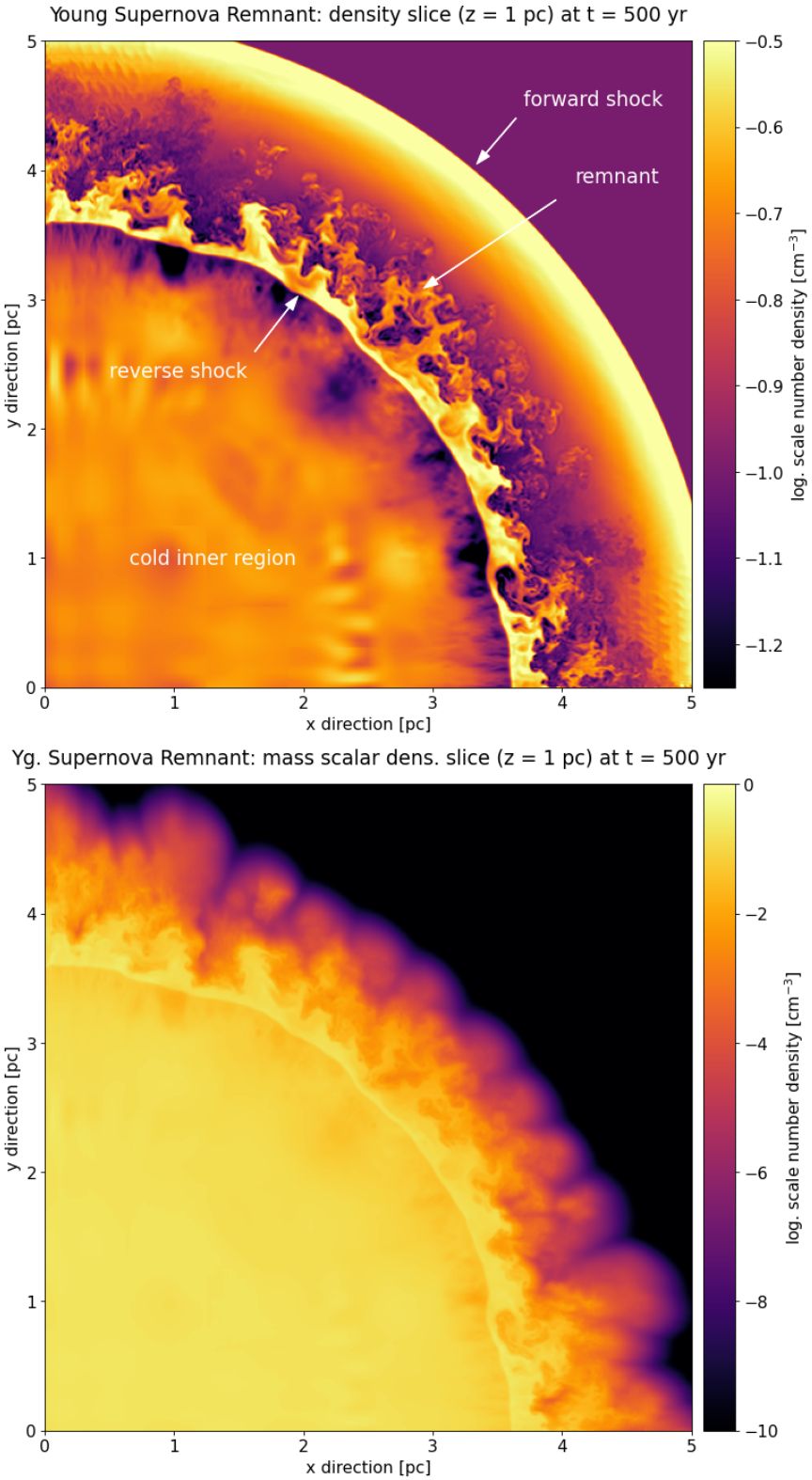

Figure 7.2: Slices of density (top) and mass scalars (bottom) at z = 1 pc and final simulation time $T = 500$ yr of the young supernova remnant simulation carried out by the DGFV4 scheme in FLASH. The noise in the density near the plot axes are caused by the carbuncle phenomenon, a well-known issue for many numerical schemes at grid-aligned shocks.

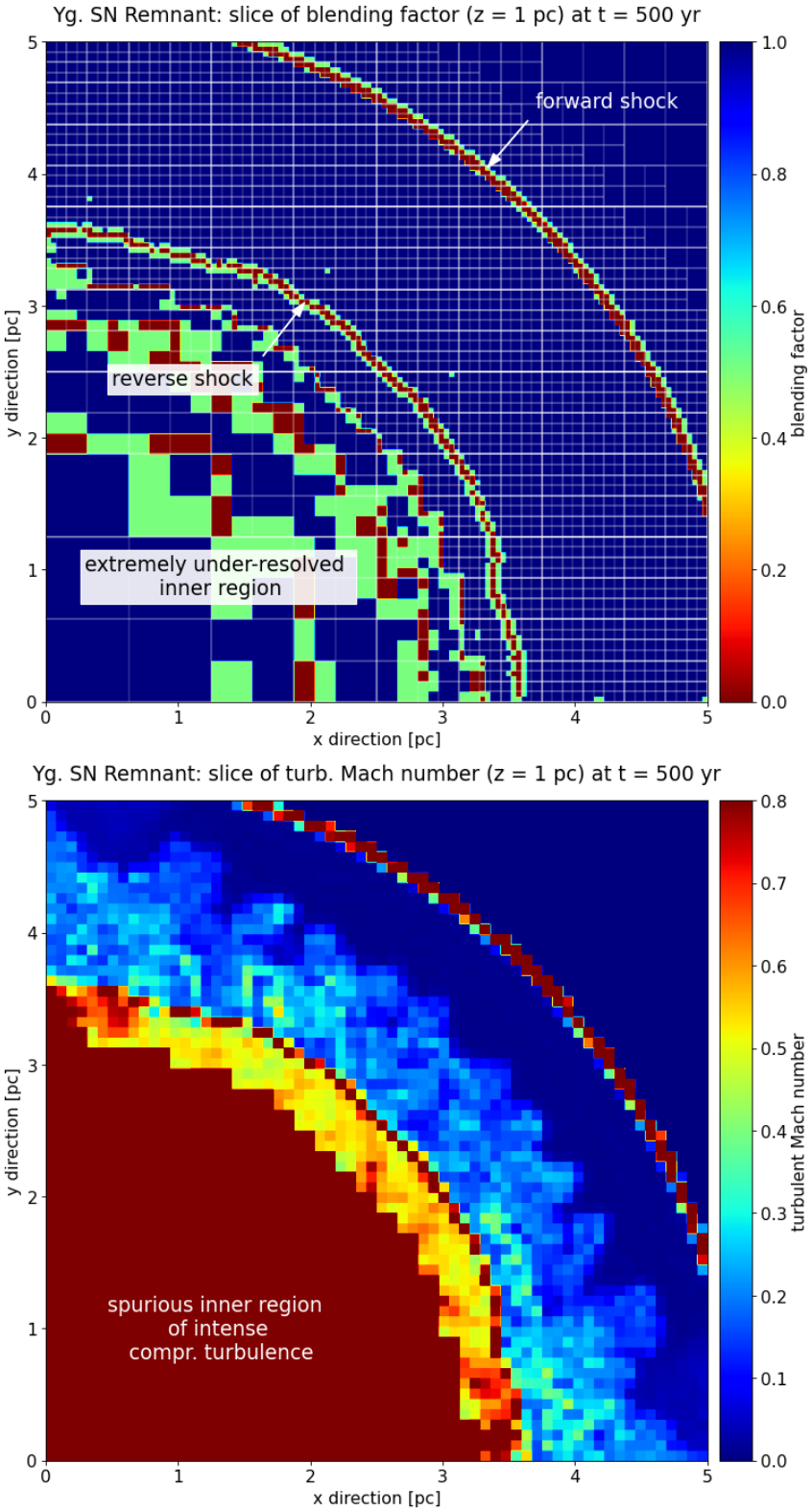

Figure 7.3: Slices of blending factor (top) and turbulent Mach number (bottom) at z = 1 pc and final simulation time $T = 500$ yr of the young supernova remnant simulation carried out by the DGFV4 scheme in FLASH. The visible numerical artifacts in the inner region are discussed in the text.

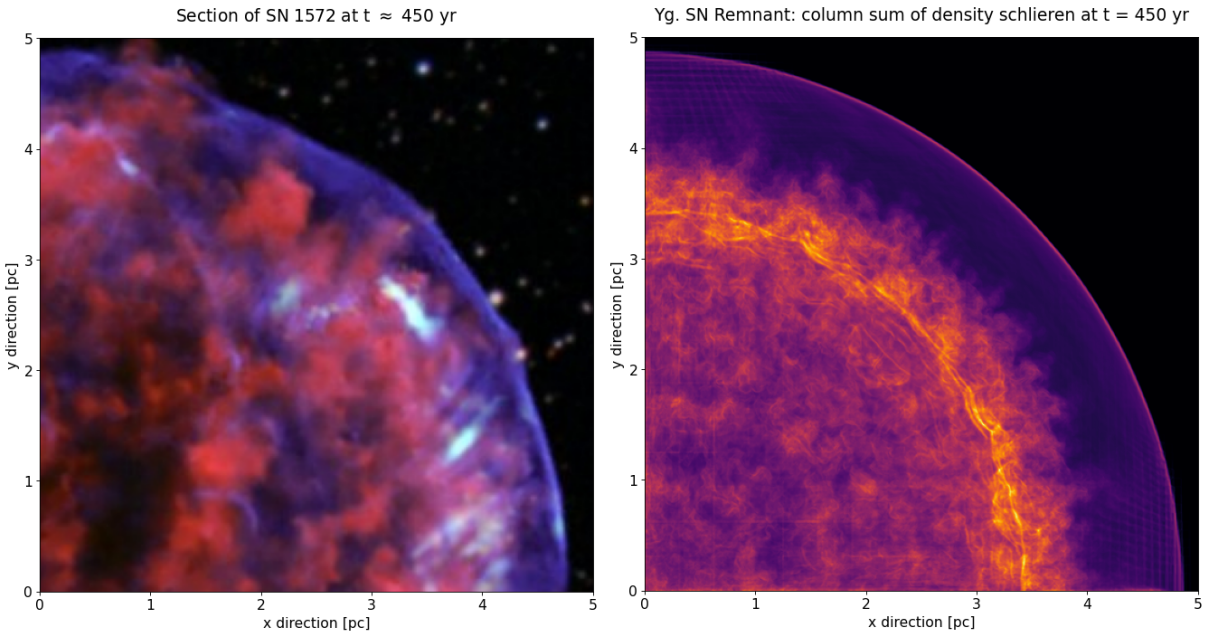

Figure 7.4: Comparison of a false-color photo (left) of SN 1572 and a density schlieren (column sum) plot (right) computed from our simulation data at time $t = 450$ yr. The photo shown here is only segment of a much larger photo. Image source: Eriksen (2011).

7.3 Differentially Rotating MHD Torus

Here, we present results of a three-dimensional global MHD simulation of magneto-rotationally instabilities (MRI) growing in a differentially rotating torus initially threaded by a toroidal magnetic field (Okada et al. 1989). The setup description and some of the results have also been published in Markert et al. (2022). Here, we replicate the setup description, append additional plots and discuss further insights. After several rotation periods the inner region of the torus becomes turbulent due to the growth of the MRI fueling efficient angular momentum transport processes (Velikhov 1959; Chandrasekhar 1960; Balbus and Hawley 1990). Magnetic loops emerge by the buoyant rise of magnetic flux sheets from the interior of the torus (Machida et al. 1999). Owing to this angular momentum re-distribution, the torus becomes flattened to a disk, where interstellar material gradually accretes to the massive source of gravity at the center. Consequently, this process leads to various interesting phenomena such as X-ray flares and jet formation.

Such setups are usually modeled with the ideal MHD equations in cylindrical coordinates (r, ϕ, z), but in this work we resort to the Cartesian coordinate system (x, y, z). Following Machida et al. (1999), we adopt an equilibrium model of the magnetized torus in order to

initialize the setup. The scales of the model are completely determined by the following parameters: gravitational constant G, mass of the central object M, initial angular momentum of the torus L, median radius of the torus r_0, characteristic density of the torus ρ_0, magnetic field strength of the initial toroidal magnetic field B_ϕ and the initial ratio of gas pressure to magnetic pressure β_0. For convenience we set all aforementioned parameters to unity: $G = M = L = r_0 = \rho_0 = B_\phi = \beta_0 = 1$. We assume the polytropic equation of state $p = K\,\rho^\gamma$ with constant $K = 0.05$ and $\gamma = 5/3$. The inviscid, non-resistive torus is embedded in non-rotating, non-self-gravitating ambient gas ($\rho = 10^{-5}$) and evolves adiabatically. Radiative cooling effects are neglected. For the static gravitational field caused by the central mass, we use the Newtonian potential provided by the *PointMass* module. The Alfvén wave speed is calculated according to Okada et al. (1989) as

$$c^2_{\text{Alfvén}} = \frac{2\,K}{\beta_0}\Big(\rho\,r^2\Big)^{\gamma-1} \quad \text{with } r^2 = x^2 + y^2 \tag{7.5}$$

and the sound speed c_{sound} is defined by (3.10). We transform the equation of motion (Machida et al. 1999) to the potential form

$$\psi_{\text{torus}} = \text{const.} = -\frac{G\,M}{r} + \frac{L^2}{2\,r^2} + \frac{c^2_{\text{sound}}}{\gamma-1} + \frac{\gamma\,c^2_{\text{Alfvén}}}{2\,(\gamma-1)}\,\Bigg|_{r\,=\,r_0} \tag{7.6}$$

and obtain an expression for the density distribution of the initial torus

$$\rho_{\text{torus}} = \left[\frac{\max\Big(0, \psi_{\text{torus}} + G\,M/R - L^2/(2\,r^2)\Big)}{K\,\gamma/(\gamma-1)\Big(1 + r^{2\,(\gamma-1)}/\beta_0\Big)}\right]^{1/(\gamma-1)}. \tag{7.7}$$

The torus rotates differentially according to the Keplerian velocity $v_\phi = \sqrt{G\,M/r}$. The computational domain is set to $\Omega = [-7, 7]^3$ with outflow boundary conditions at all faces. The AMR grid is bounded between refinement levels of 64^3 cells to 256^3 cells. The first row in Figure 7.5 shows what the initial torus looks like in our setup.

After several periods, regions of dominating magnetic pressure ($\beta < 1$) emerge, where magnetic flux buoyantly escapes from the disk leading to violent flaring activities. The loop-like structures, similar to those in the solar corona, are also visible in our simulation after eight rotation periods as shown in Figure 7.5 in the second and third row. This setup shows that the DGFV4 solver is capable of simulating complex MHD flow configurations under the influence of a static gravitational potential, giving rise to magnetically driven

structures over several spatial scales.

Figure 7.6 depicts slices of the turbulent Mach number $\mathcal{T}$ (left) and the blending factor (right). Evidently, the disk is only moderately compressible with low $\mathcal{T}$ in the outer regions ($r \approx 5$) and only needs minor blending. The compressibility increases considerably towards the center and the outer boundary of the disk is clearly marked by a sharp disruption of the velocity field in radial direction. This also visible in the left plot of Figure 7.7, showing slices of the actual velocity field. Although, this region is not a "classical" shock, blending is still heavily triggered to stabilize the computation.

Another interesting aspect of this simulation is observable in Figure 7.7 on the right side showing the velocity field in the x-z plane. Magneto-rotational forces throw a jet of material at high speed from the disk's center in opposite directions. The flow at the center is generally very challenging in such simulation, and our DG scheme is not robust enough, and falls back to the low order FV scheme. The study of such jets is subject of active research and even today not all processes involved in the complex dynamics in observed jets are fully understood. Hence, there is an urgent need for very accurate, robust fluid schemes, which preserve angular momentum and also ensure high fidelity in modeling the intricate magneto-rotational dynamics at the center, where the speed of rotation is enormous.

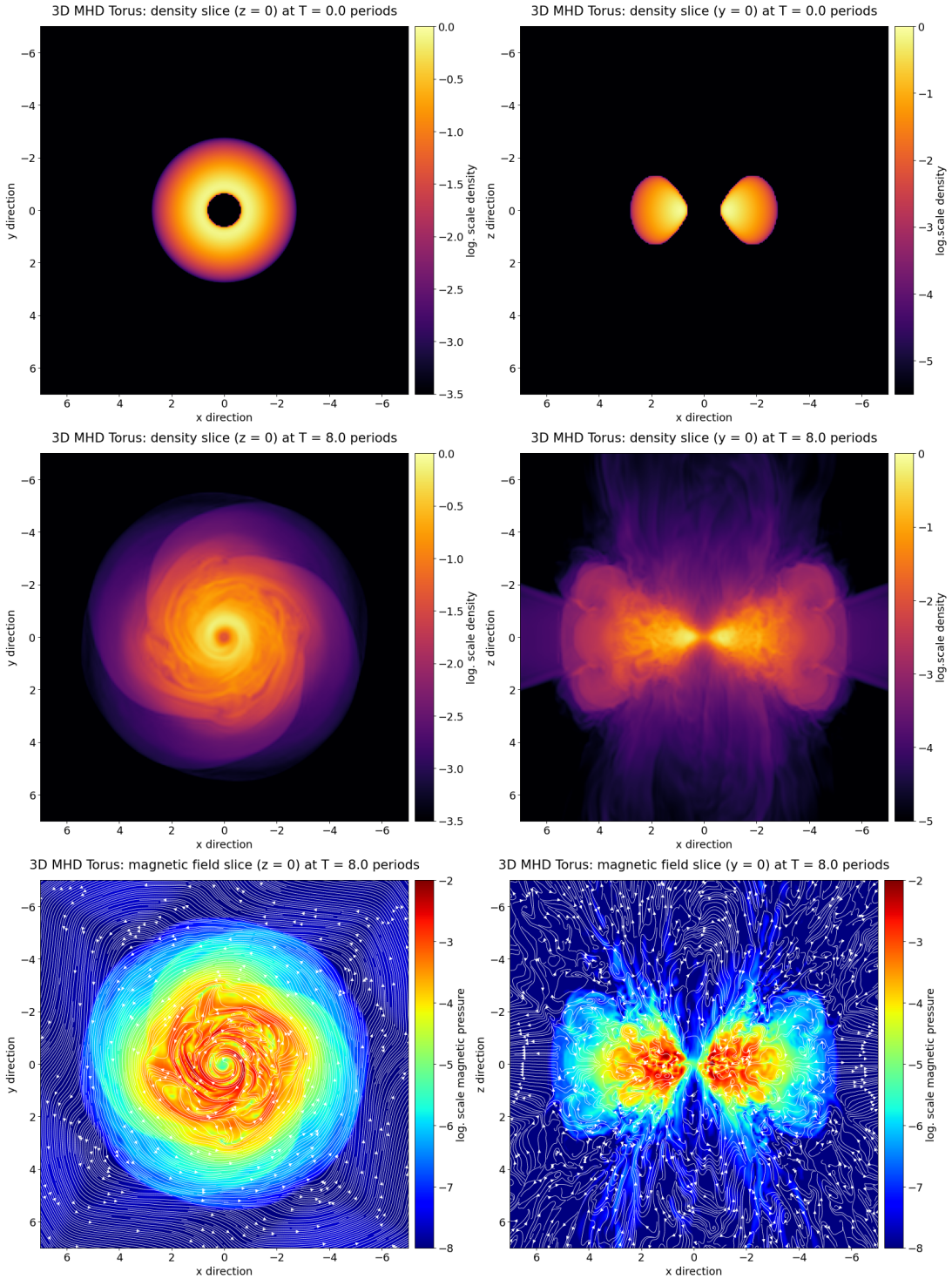

Figure 7.5: MHD torus simulation at the initial time (top row) and after 8 periods carried out by our DGFV4 solver in FLASH. Shown are slices of the density (middle row) and magnetic field (bottom row) in x-y and x-z plane. Image source: Markert et al. (2022).

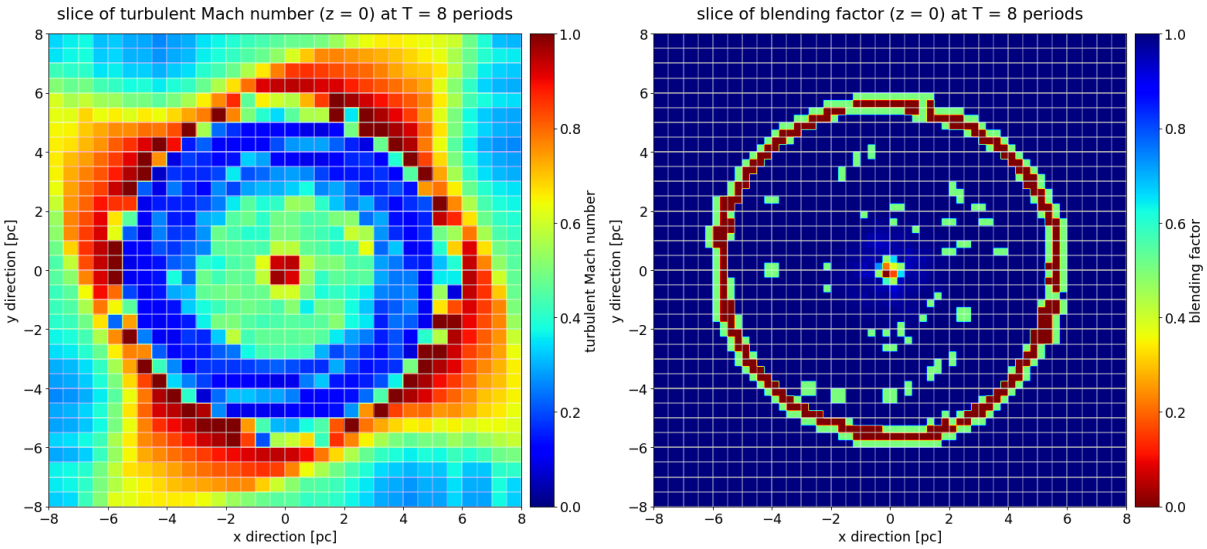

Figure 7.6: Slices of turbulent Mach number and blending factor in the x-y plane for the MHD torus simulation after 8 periods carried out by our DGFV4 solver in FLASH.

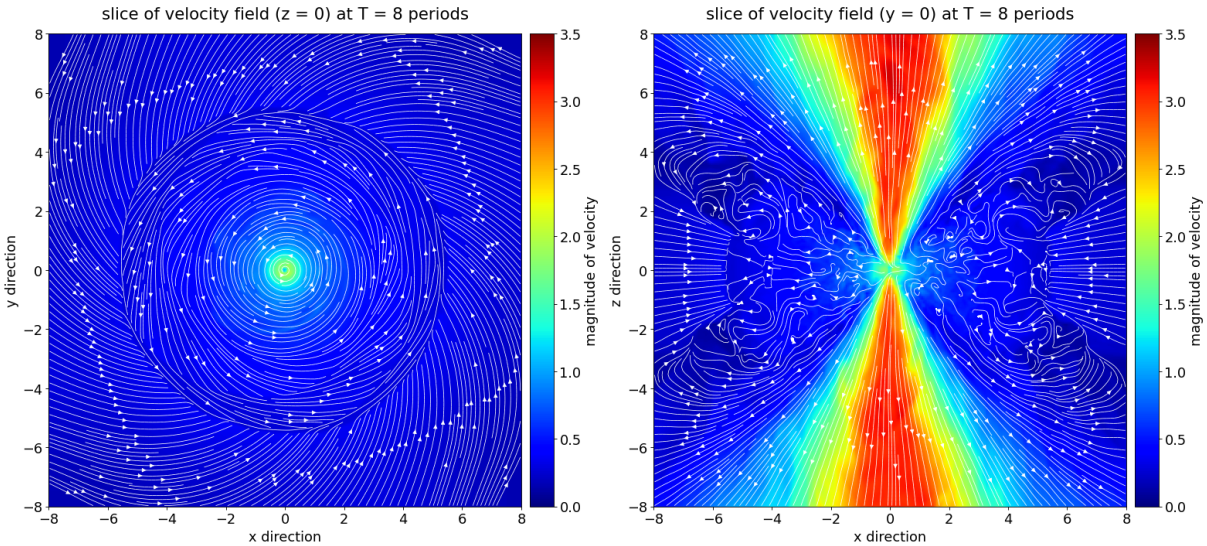

Figure 7.7: Slices of velocity field in x-y and x-z planes for the MHD torus simulation after 8 periods carried out by our DGFV4 solver in FLASH.

In the following, we present a small comparative study of this setup. We run a total of six simulations: three different resolutions of 64^3, 128^3, and 256^3 cells each with the two MHD solvers Bouchut5 and DGFV4 in FLASH. All runs are conducted on the ODIN cluster with 16 á 16 cores. This gives a total cpu count (#cpus) of 256 cores.

The results at final simulation time are shown in Figure A.4 and Figure A.5 in appendix A.4. The DGFV4 solver shows finer structures and more scales in the disk at every

resolution level compared to Bouchut5 and is capable to resolve magneto-rotationally driven outflows at lowest resolution levels.

In Table 7.3 we list the runtimes of all six simulations. Bouchut5 is a split solver and thus in general has a larger time step (about a factor of 2) compared to unsplit schemes. Hence, the number of time steps (#steps) are lower for Bouchut5 compared to DGFV4, which results in overall faster completion times for the same resolution. To further assess our implementation, we also compute the throughput (TP) according to (5.3). Note, that the MPI-parallelized code scales better with increasing resolution resulting in higher TP for both schemes. A drawback of DG schemes compared to pure FV discretizations is that the spatial operator spectra are more stiff and hence need more stable time integration methods, such as the high-order RK method used in this comparison. This means, that the DGFV4 discretization has to be called four times per time step, resulting in longer runtimes. If we compare, however, the TP of both schemes, we see that our DGFV4 implementation is actually faster than the Bouchut5 implementation as we have already seen in other runtime comparisons in this thesis.

Table 7.3: Runtimes of the MHD torus setup for the Bouchut5 and our DGFV4 solver implemented in FLASH. All simulations ran on the cluster ODIN on 16 nodes á 16 cores. The definition for the computational throughput (TP) is given by eqn. (5.3).

solver	DOF	runtime [h]	slowdown	#steps	slowdown	TP [10^4 DOF/s]	speedup
Bouchut5	64^3	0.03	1.0	940×1	1.0	0.91	1.0
	128^3	0.41	1.0	3012×1	1.0	1.66	1.0
	256^3	6.14	1.0	8374×1	1.0	2.48	1.0
DGFV4	64^3	0.08	2.67	1736×4	7.4	2.56	2.8
	128^3	1.18	2.88	4808×4	6.4	3.71	2.2
	256^3	17.24	2.81	13898×4	6.6	5.87	2.4

7.4 Expansion of HII Region into Fractal Medium

Radiation driven feedback from massive stars is believed to be a key element in the evolution of molecular clouds. In this work, we utilize our new DGFV4 solver to simulate the dynamical effects of D-type expansion powered by a single O7 star emitting ionizing photons at $10^{49}\,\mathrm{s}^{-1}$ located at the center of a molecular cloud. The cloud consists of 10^4 solar masses and stretches over a region of 6.4 pc in each direction, so the volume-weighted mean density of the cloud amounts to $\rho_0 = 0.62 \times 10^{-21}\,\mathrm{g\,cm}^{-3}$. The setup description and

some of the results have also been published in Markert et al. (2022). Here, we replicate the setup, append additional plots and expand on further insights.

It is a well established fact that molecular clouds are rich in internal structure probably driven by pure turbulence (Klessen 2011; Girichidis et al. 2011) and subscribe to a statistically self-similar fractal structure (Stutzki et al. 1998). Our goal is to model the development of a molecular cloud having initial fractal dimension of $\mathcal{D} = 2.6$ with a standard deviation of the approximately log-normal PDF of $\sigma_{\mathrm{STD}} = 0.38$. With these parameters the cloud is initially dominated by small-scale structures, which are then quickly overrun by the advancing ionization front, thereby producing neutral pillars protruding into the HII region. The ionized gas within the expanding hot bubble blows out through a large number of small holes between these pillars. These regions are termed pillar-dominated (Walch et al. 2012). Fractals with mass-size relations similar to those observed by Larson (1981) can be created in Fourier space (Shadmehri and Elmegreen 2011), so we can freely configure the fractal dimension $\mathcal{D}$ of the initial cloud. A detailed account on how to construct such a fractal density distribution is presented in Walch et al. (2012).

The aspects of radiation physics are handled by the TreeRay module in FLASH which we described in Section 6.3.5. Besides minor changes in setup parameters given below, this setup uses the same parameters as in Section 6.3.5. In contrast to Section 6.3.5, we simulate the whole expanding bubble, but keep the same maximum resolution at 128^3 cells. Hence, the radiation source is at the center of the cubic computational domain $\Omega = [-6.4\,\mathrm{pc}, 6.4\,\mathrm{pc}]^3$ and all boundaries are set to outflow. Radiative cooling effects of the molecular cloud itself are neglected. The result at final simulation time $T = 1\,\mathrm{Myr}$ can be seen on the right in Figure 7.8. The plot shows the column density in z-direction uncovering the aforementioned pillars as a product of the turbulent-like density distribution tearing apart the shell structure. A comparable example in our Galaxy is the Rosette nebula, which is an expanding HII region powered by a cluster of several O-type stars located in the outer zones of a giant molecular cloud (Wang et al. 2008, 2009). A false-color image of the nebula taken by Corban with the Hubble Space Telescope (ESA/ESO/NASA) is shown in Figure 7.8 on the left side.

The simulation took 13.41 hours with 4×16 cores on ODIN and went through 1243 timesteps. It is important to note that the lion's share of the total runtime was occupied by the physics computations: 1.49 hours (11 %) were spend on the hydrodynamics (DGFV4 solver) and 11.74 hours (87.5 %) were consumed by physics, i.e. Gravity and TreeRay.

thus, a fluid solver that needs less total timesteps for such simulations is at advantage with regards to completion times. At each additional timestep the expensive physics computation are fired up, eventually adding up to much longer runtimes. Bouchut5's computation took only 424 timesteps and finished in 4.42 hours. A remedy to this issue could be to apply sub-cycling in the fluid solver and to just compute the expensive physics every couple of timesteps.

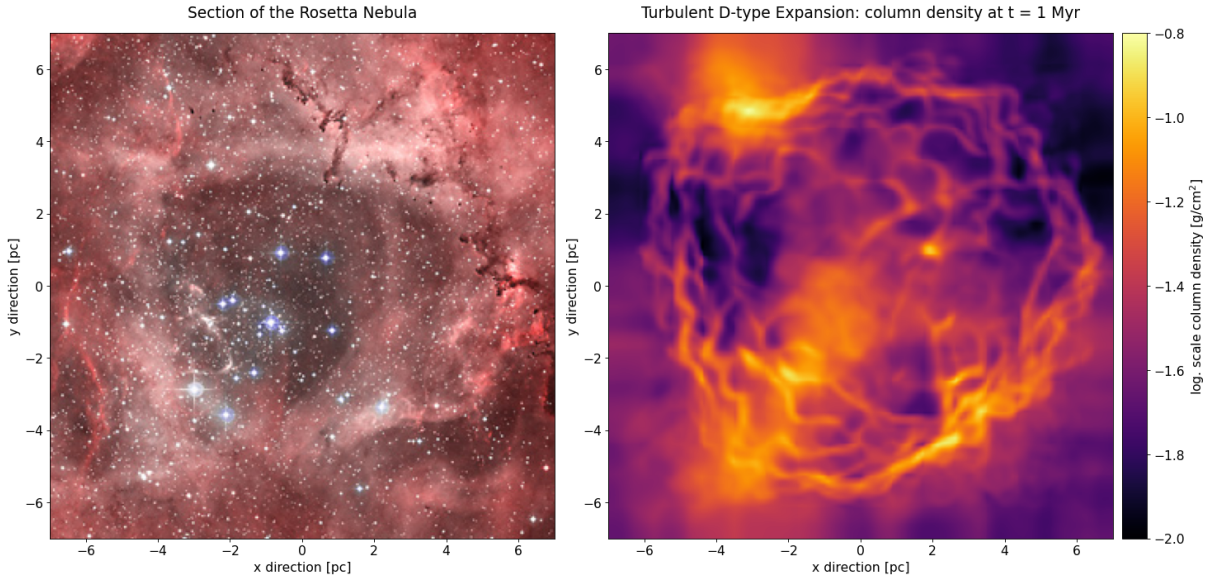

Figure 7.8: Observed turbulent D-type expansion within the Rosette nebula (left, Image source: Corban) and simulation within a fractal medium run by our DGFV4 solver. The false-color image on the left shows only a section of the much larger nebula with a radius of 20 pc taken by the Hubble Space Telescope (ESA/ESO/NASA). The right side shows the column density along the z-axis: $\int_{z_{\min}}^{z_{\max}} \rho(x, y, z)\, \mathrm{d}z$. Image source: Markert et al. (2022).

The result of the reference simulation by Bouchut5 with identical setup parameters is shown in Figure A.6 in appendix A.5. The density plots by our DGFV4 solver seems a bit diffuser than with Bouchut5. This suggest that the simulation is dominated by the FV method in our blending scheme. To investigate this further, we plot slices of the density (top left), together with temperature (top right), turbulent Mach number (bottom left) and blending factor (bottom right) shown in Figure 7.9. Evidently, the most interesting flow structure, the expanding, turbulent shell, gets only resolved by the dissipative FV scheme. The activated blending (wide red circle) directly correlates with regions of supersonic turbulent Mach number. As discussed before, higher order DG schemes are not robust enough to compute under this regime and cannot act out their superior resolution capabilities. Even though this setup combines multi-species and

radiation physics in a very tough shock-turbulence regime, this simulation nevertheless shows that our new "hardened" DG implementation in FLASH is capable of running such complex multi-physics applications in astrophysical settings.

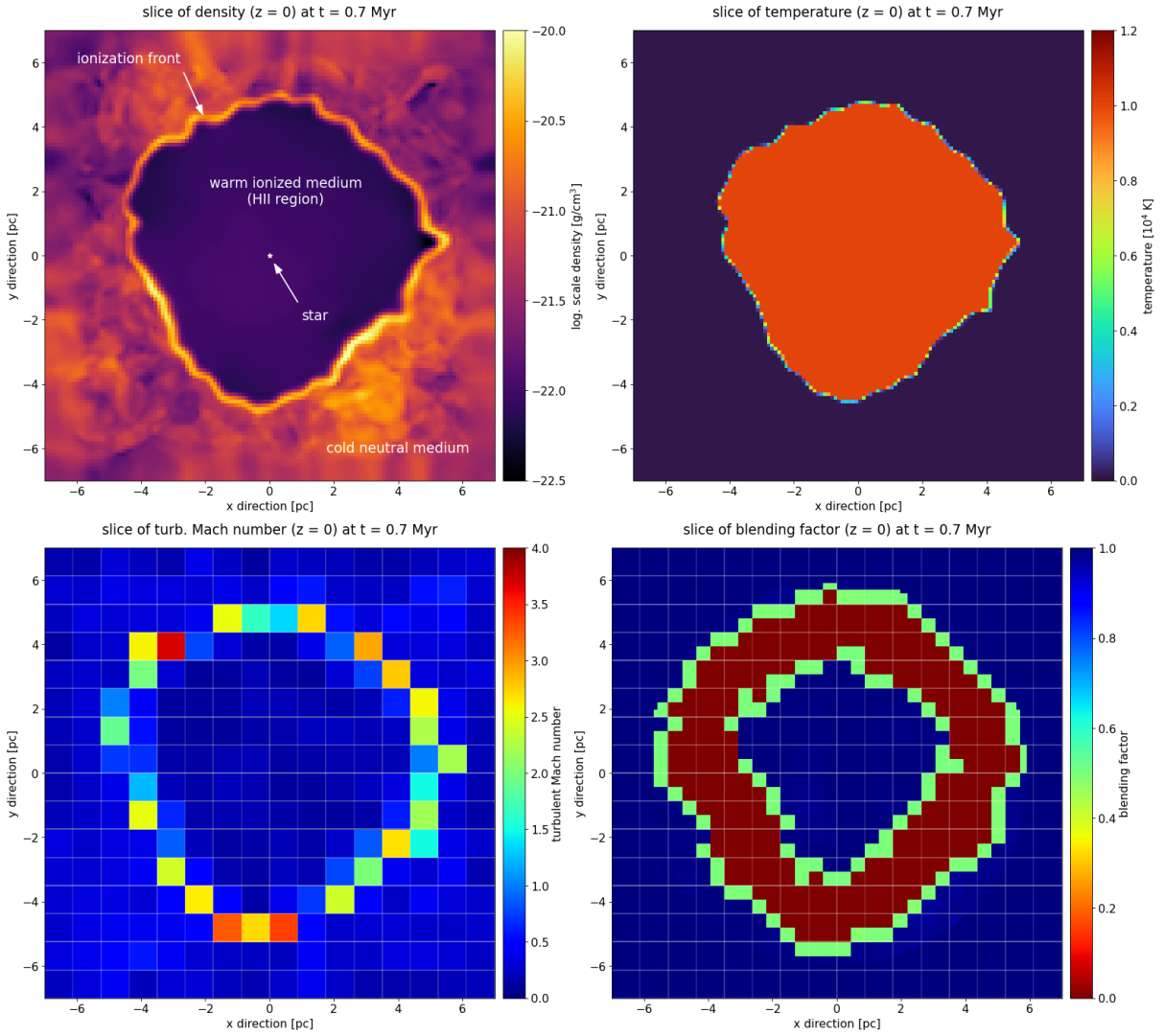

Figure 7.9: Turbulent D-type expansion within a fractal medium simulated by our DGFV4 solver in FLASH. Shown are slices of density, temperature, turbulent Mach number and blending factor along the x-y plane at $t = 0.7$ Myr.

On a final note, we also document the simulation results for an extension to this setup, where in addition to the expanding HII region, the radiating O-type star emits strong stellar winds blowing away all matter in its vicinity. The effect is a near-vacuum bubble around the star, which is an additional challenge for our DGFV4 scheme. It should come as no real surprise that this simulation is also dominated by the FV method in our

blending scheme. The results in form of density slices are shown in Figure A.7 by our scheme and Figure A.8 by Bouchut5 in appendix A.6. Nonetheless, we put on record that our DG implementation in FLASH is able to correctly carry out such setups, albeit yielding more diffusive results compared to the reference solver at this point.

7.5 Molecular Cloud in Hot Galactic Wind

Even though the physical origin of hot galactic outflows is still not exactly understood, it is well established that they significantly contribute to the matter transfer cycle in star forming molecular clouds within galaxies. Hence, modeling the physical dynamics and chemical processes in galactic clouds under the impact of supersonic cosmic winds is another cornerstone in a comprehensive theory of star formation. Based on previous work by Girichidis et al. (2021), we aim to perform an idealized simulation of a wind tunnel experiment simulating an in-situ molecular hydrogen formation scenario of a turbulent, warm, self-gravitating cloud with a number density of 1 cm^{-3} of mainly atomic hydrogen and exposed to a magnetized, ionized, supersonic, hot wind. Strong magnetic fields influence the gas dynamics and can redirect the flow, where the orientation of the field in relation to the prevailing wind direction plays a critical role. In our case, the initial magnetic field is parallel to the wind direction. The setup involves non-equilibrium chemistry via a chemical network, which includes the abundances of ionized (H^+), atomic (H) and molecular (H^2) hydrogen, carbon monoxide (CO), singly ionized carbon (C^+), and free electrons. The aspect of background heating through UV radiation is locally attenuated by TreeRay in optically thick regions. The exact details are documented in Girichidis et al. (2021).

The outline of the setup is as follows. The computational domain is an elongated box with dimensions $[-100, 900] \times [-100, 100] \times [-100, 100]$ pc with maximum AMR resolution of $640 \times 128 \times 128$ cells, in which a spherical cloud with a temperature of 700 K and radius 50 pc is placed at the origin of the domain. The hot, fully ionized, ambient gas surrounding the cloud has a uniform total number density of $10^{-3}\,\mathrm{cm}^{-3}$ and a temperature of 10^6 K. The total mass of the background gas integrates to 870 solar masses. The cloud is initially in pressure equilibrium with the ambient medium with all hydrogen in the neutral atomic phase. The density of the cloud is close to uniform with a small asymmetric perturbation to promote symmetry breaking. In addition to the density perturbation, a small turbulent velocity field is applied in the cloud. The cloud is also initially permeated by a turbulent,

divergence free magnetic field with a root-mean-square field strength of a 10^{-6} G, equal to the magnetic background field.

Evidently, in this setup there are a lot of unmentioned, rather complex, astrophysical technicalities involved; only fully comprehensible by experts in the field. Thankfully, the simulations in Girichidis et al. (2021) were also conducted with FLASH and the main author granted us free access to their complete set of original setup files, allowing us to readily commence the simulations ourselves without further ado. We want to emphasize again that our DG solver in FLASH is indeed "plug-and-play", since we could carry out this unfamiliar setup in a straightforward manner.

In Figure 7.10 we show the column sum of total density in log-scale of the churned up cloud after 20 Myr together with column sums of the mass weighted abundances H^+, H, H_2, and CO. The reference solution conducted with Bouchut5 is shown in Figure A.9 in appendix A.7. On large scales, both solutions are comparable with similar amounts of H_2 and CO accumulated at identical simulation times. As observed before in Section 7.4, the solution by DGFV4 is rather diffusive. In Figure 7.11 we plot slices of the turbulent Mach number (top) and the blending factor (bottom) along the x-y plane, which confirms the correlation of very compressive flows with excessive triggering of the blending. We conclude that with our current state-of-the-art high order DG implementation in FLASH, this setup will not profit with regards to getting a larger range of spatial scales with equal DOF compared to established FV methods in FLASH.

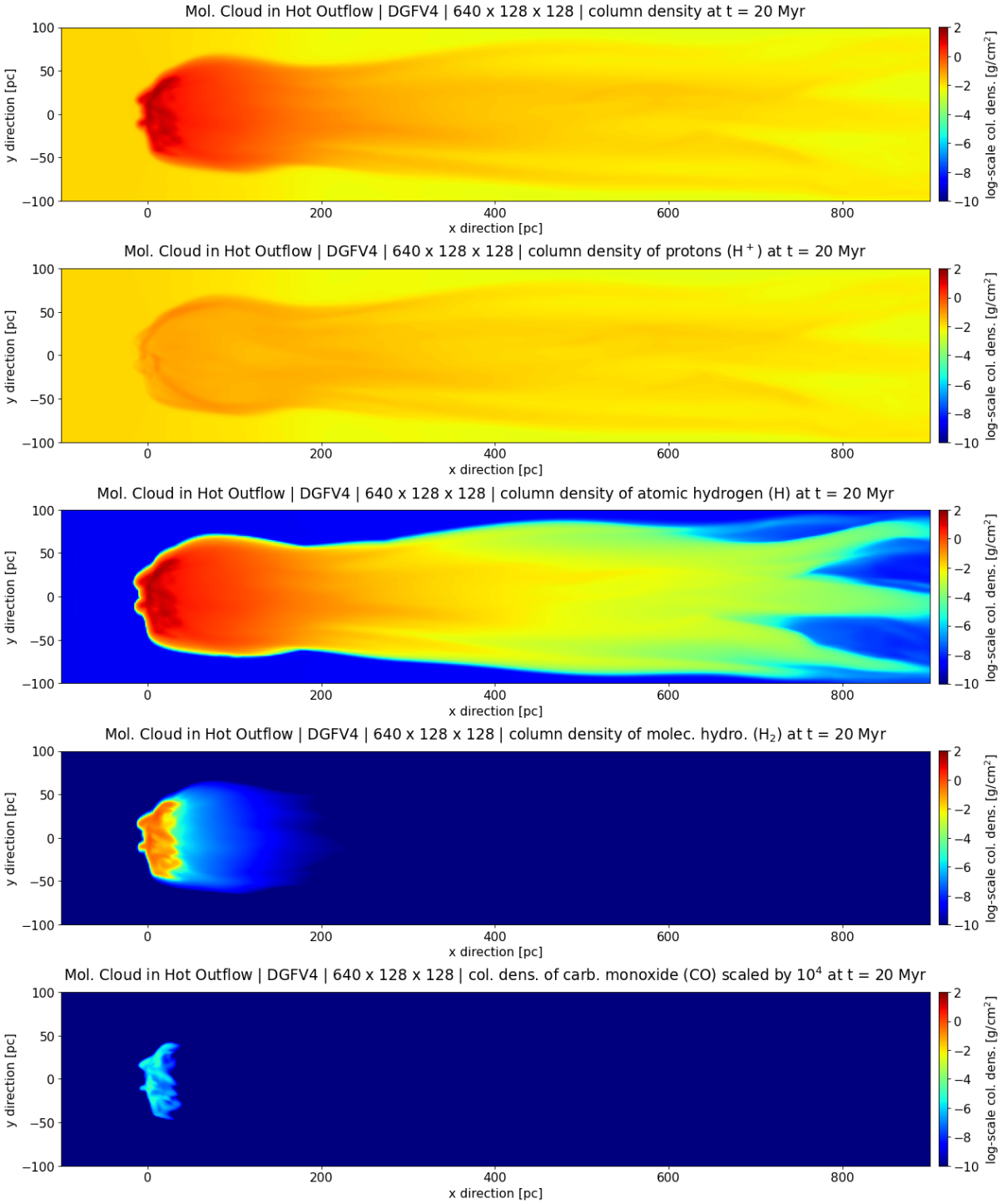

Figure 7.10: Molecular cloud within a jet of hot galactic outflow run by the DGFV4 solver. Shown is the column density along the z-axis for the total mass and different species.

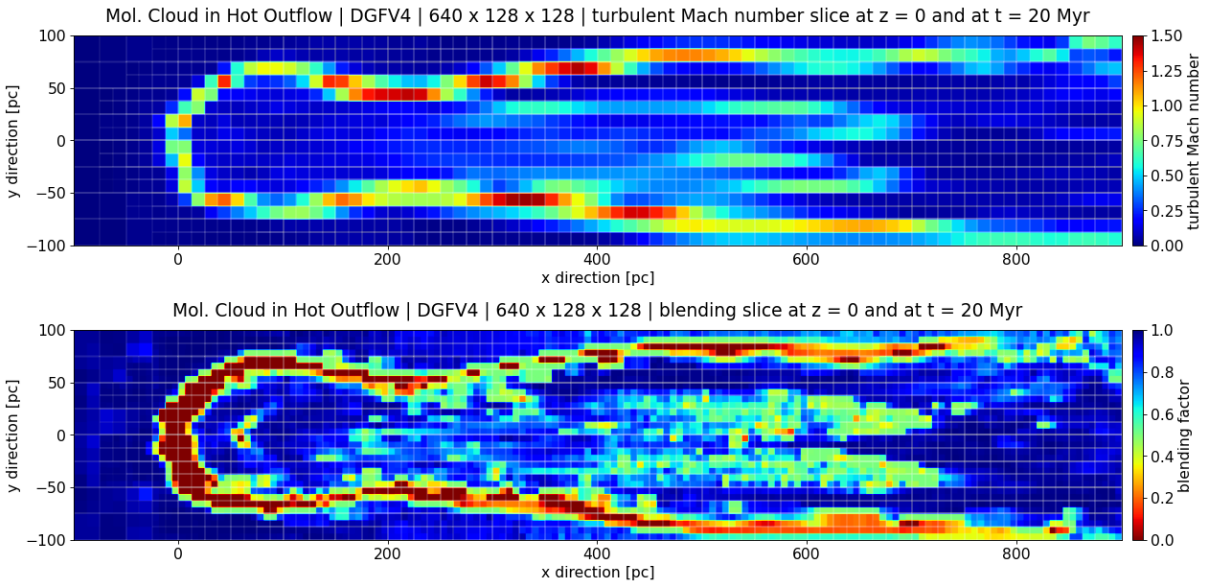

Figure 7.11: Molecular cloud within a jet of hot galactic outflow run by the DGFV4 solver. Shown are slices along the x-y plane of the turbulent Mach number and the blending factor at $t = 20$ Myr.

On a final note, we present an example in our Universe with similar characteristics as the shredded cloud setup shown here. The so-called Smith's cloud in Figure 7.12, discovered by Smith (1963), has dimensions of roughly 3 kpc by 1 kpc and contains over a million of solar masses (Lockman et al. 2008). The cloud is moving towards the disk of our Galaxy with high velocity hitting gas in our Galaxy's outskirts, which gives it the shape resembling a comet with a tail. The false-color image was taken by Saxton (2008) with the Green Bank Telescope.

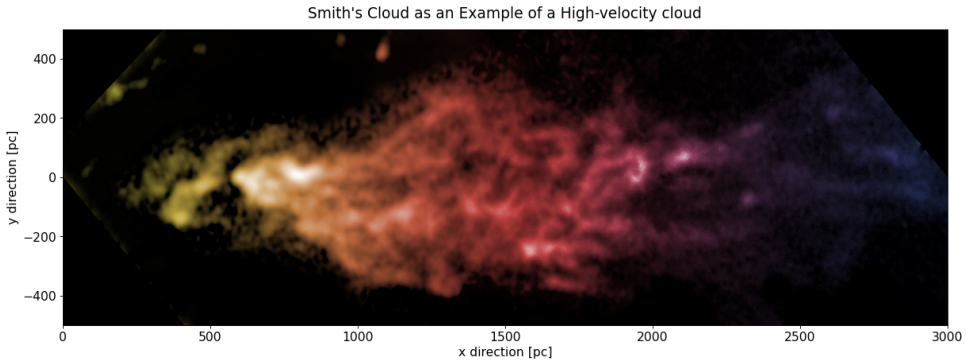

Figure 7.12: False-color image of Smith's Cloud as an example of a high-velocity cloud hitting gas in our Galaxy's outskirts, which gives it the shape resembling a comet with a tail. The image shown here was rotated and cut to size in order to fit the page formatting. Image source: Saxton (2008).

Chapter 8

Conclusion & Outlook

8.1 Recapitulation

In engineering applications, discontinuous Galerkin (DG) schemes have been proven to be a powerful and flexible class of high order methods for solving differential equations such as the hyperbolic conservation laws of hydrodynamics discussed Chapter 2. However, the potential benefits of DG for applications in astrophysical contexts is still relatively unexplored in its entirety. Of course, there have already been published several studies surveying DG for astrophysical flows in the past. But the adoption of DG by the astrophysics community is just beginning to gain traction and integration of DG into established, multi-physics simulation frameworks for comprehensive astrophysical modeling is still lacking. It is our firm believe, that the full potential of a novel fluid solver only shows under the pressure of real-world simulations with all aspects of multi-physics, challenging flow configurations, resolution and runtime constrains, and scalability on high-performance clusters involved. Thus, we saw the pressing need to propel DG from the well-trodden path of cataloguing test results under "optimal" laboratory conditions to the harsh environment of large-scale astrophysics simulations. In Chapter 3 we outlined the various aspects of our envisaged simulations and drew conclusions what a new DG solver has to bring along to be useful.

Consequently, the core of this thesis is the development and deployment of a robust DG scheme for the ideal MHD equations on 3D Cartesian grids with AMR for astrophysical flows. We chose to implement our new scheme within the venerable simulation frame-

work FLASH, with a specific focus on multi-physics problems in astrophysics. It entailed modifications of the vanilla DG scheme to make it fit seamlessly within FLASH in such a way, that all other physics modules can be naturally coupled without additional implementation overhead. A key ingredient is that our DG scheme uses mean value data organized into blocks - the central data structure in FLASH. Having the opportunity to work on mean values, allowed us to rely on a rock-solid TVD FV scheme as "backup" when our high order DG chocked in cases where the flow configuration got too fierce. Finding ways to combine the two schemes in a fail-safe manner without loosing primary conservation while still maintaining high order accuracy for smooth, well-resolved flows involves a series of careful considerations, which we documented at length in Chapter 4. The result of our work is a shock capturing scheme - a hybrid between FV and DG - with smooth transitions between low and high order fluxes according to solution smoothness estimators. Along the way, we also investigated the benefits and pitfalls of integrating end-to-end entropy stability into our numerical schemes, especially with focus on highly compressible turbulent flows.

In Chapter 5 we introduced our Fortran-based CFD code NEMO specifically suited for rapid prototyping numerical schemes. The code is capable of running small- to medium-sized simulations leveraging hybrid parallelization with OpenMP and MPI, and supports AMR powered by the P4EST octree library. Moreover, NEMO was an integral part in designing and testing the aforementioned shock capturing scheme by filling the gap between the rigidity of pre-existing DG codes and the overwhelming complexity of full-stack astrophysics frameworks. Tweaking our implementation for maximum execution speed and near-perfect scalability were, however, not our main goal, even though DG has a huge potential in this regard. First and foremost, we focused our efforts on robustness and versatility.

After a long series of design iterations, fine-tuning and putting the scheme to the acid test with NEMO, we were finally confident enough to successfully transplant our scheme into FLASH. In Chapter 6 we give implementation details and discussed a long range of stringent test cases showing that the our DG scheme works properly and is fully connected to all multi-physics modules. Step by step, the complexity of the test cases is increased by using setups that require increasingly complex physical modules and features. Finally, a fully coupled multi-physics simulation setup shows that the novel DG solver in FLASH is ready for use in full-stack astrophysics simulations and thus ready for assessments and investigations under real-world conditions.

Finally, we ventured into a broad range of astrophysics applications each with different multi-physics aspects and varying demands on the fluid solver. In Chapter 7 we show and discuss our results of five exemplary setups, from the rapid expansion of a supernova to chemical conversions in cosmic clouds under the impact of supersonic winds.

Based on the experience we gained from the simulations we can now answer our research questions raised in the introduction of this thesis.

8.2 Revisiting the Scientific Objectives

Hinchliffe's rule, named after particle physicist Ian Hinchliffe, states that if a research paper's title is in the form of a "yes-no" question, the answer to that question will most likely be "no" (Peon 1988). Among journalists this adage is also commonly known as Betteridge's law of headlines. It is based on the assumption that if the authors were confident that the answer was "yes", they would have presented it as an assertion; by presenting it as a question, they are not accountable for whether it is correct or not. Of course, this polemic statement is controversially disputed (Shieber 2015) and should not be taken too seriously. Nevertheless, in every banter there is a dash of truth.

In the context of this thesis, we formulated four basic questions serving as guard rails on the way towards a robust and accurate DG scheme for challenging astrophysics applications. All in all, we can confidently answer all four questions with "yes" whilst taking into account some compromises we accepted in order to reach our goals.

The first objective aims at the question if there are high order DG schemes which can successfully carry out compressible turbulent flows.

1.) Can we successfully run a high order DG method in transonic compressible inviscid flow regimes?

We can report that all standard and entropy stable fourth order DG methods eventually crash for a specifically designed test case except for the entropy stable Flux Differencing DG variant with Gauss quadrature rule. The failure of the standard DG variants was expected, since they are already known for their susceptibility to aliasing even in weakly compressible turbulence simulations. The impracticalness of the entropy stable DG variants with Lobatto quadrature for compressible flows is disappointing, since its straightforward derivation, simple implementation and reasonable runtime performance

are great arguments in its favor. Although the entropy stable Flux Differencing Gauss DG is surprisingly robust, it has its drawbacks. The entropy boundary projection, necessary for entropy consistency, is detrimental in the proximity of shocks, since it greatly amplifies the oscillations of the already "troubled" solution polynomial. Moreover, benchmarks reveals that the scheme is up to six times slower than the standard DG methods. Consequently, we aim for the standard method with Gauss quadrature as baseline for our high order DG scheme. Instabilities due to aliasing or general under-resolvement are effectively handled by our convex blending scheme without unacceptably smearing out the solution. We conclude that assuring entropy stability on a discrete level alone is not enough for high order methods in tackling under-resolved, compressive flows and can be even detrimental. Thus, we relaxed our insistence on perfect entropy consistency in our codes. We did not observe any significant falsities in our numerical solutions due to excessive accumulation of numerical entropy errors.

2.) Can we devise a shock-capturing scheme for high order DG, which is capable of resolving strong shocks in a robust and sharp manner while preserving high order accuracy in smooth, well-resolved flows?

We devised a novel convex blending scheme combing the robustness of a second order TVD FV scheme with the fidelity of a fourth order DG method. We show that our shock capturing approach is capable of handling strong shocks in a robust manner and yields sharp solution profiles. Furthermore, the blending scheme is high order in well-resolved flows and effectively stabilizes the high order DG in under-resolved, compressive flows. We assume that the dissipative contribution of the FV scheme is sufficient to compensate for spurious entropy growth caused by the DG scheme. The divergence constraint by the MHD equations is naturally been taken care of due to hyperbolic divergence cleaning.

3.) Can we fully integrate a "hardened" DG method into a popular, actively maintained multi-physics framework for large-scale astrophysics applications, such as FLASH?

We successfully integrated our convex blending scheme into the simulation framework FLASH and verified the correct coupling with all aspects of multi-physics by a large range of stringent test cases. Moreover, we showed that the runtime performance is reasonable. That is, on average our solver is five times slower compared to other established fluid solvers in FLASH. Even though our implementation is faster in terms of raw throughput,

it is at disadvantage with regards to the tighter timestep constraints and the application of multi-stage RK schemes.

4.) Is our new DG method capable of carrying out comprehensive multi-physics simulations?

Our new solver module in FLASH is capable of running a broad range of astrophysics simulations with different aspects of multi-physics and varying demands on the solver. Even though our solver is able to robustly carry out simulations with predominantly super- to hypersonic turbulent flows, it is not superior with regards to solution quality or wider ranges of larger to smaller scales as is expected from high order DG schemes. The high order component in our blending scheme is not robust enough to withstand the pressure of such highly demanding flow configurations and the fallback FV method completely takes over, which is by construction much more dissipative. This observation is in line with similar studies, for example, conducted by Ma et al. (2015). The multi-physics aspects in the expanding HII region, the stellar feedback simulation and the cloud shredded by hot cosmic winds are modeled correctly by our solver as we confirmed by comparing with solutions from another second order FV fluid solver available in FLASH. However, our solutions are visibly smeared out compared to the other solver. Evidently, the reference solver is more sophisticated than our FV scheme and was especially designed for this kind of simulations. On the other hand, for simulations with significant portions of sub- to transsonic flows, namely the supernova explosion and the rotating MHD torus, our high order DG method gives considerably more scales in the solution compared to second order schemes at the same level of resolution. Blending is only triggered when really needed to stabilize DG; be it in the proximity of shocks or occasionally, when under-resolved flow structures trip up DG.

We conclude that with our new convex blending scheme we can achieve superior results with regards to the range of spatial scales for sub- to transsonic turbulent flows and with isolated strong shocks. Multi-physics, such as gravity, radiative transfer and chemistry networks function according to specification. Our implementation of a "hardened" DG method in FLASH is ready for use and further assessment by astrophysicists.

8.3 Follow-up Research

Guided by the answers to our research objectives, we propose three promising directions for future research efforts that help to push forward the utilization of high order DG in general and for astrophysics applications in particular.

1.) Augmenting the existing convex blending scheme

Evidently, there is a lot of room for improvement in our proposed blending scheme. At first, a better FV scheme with more accurate reconstruction methods and more sophisticated Riemann solvers will directly enhance resolution of flow structures in regions where blending is frequently triggered. However, this measure might reduce the effectiveness in stabilizing the DG scheme, since the fallback FV method becomes less dissipative. Secondly, smarter, physics-aware shock indicators might discern more precisely where and when to activate blending. Better indicators in turn will pave the way for (re-)introducing more sophisticated DG schemes such as the entropy stable Flux Differencing DG with Gauss quadrature. More stable DG schemes allow to increase the accuracy order, which in turn might benefit, for example, the supernova simulation and the rotating MHD torus setup, presented in this thesis.

2.) Using invariant domain preserving DG schemes

Initiated by Guermond et al. (2018, 2019); Kuzmin (2020), recent developments in researching robust DG schemes, called invariant domain preserving (IDP) DG, might be an answer to supersonic turbulence simulations. IDP DG is based on the same fundamental idea of blending a robust monotone scheme with an accurate high order method, but infers the blending strength from positivity constraints and minimum-maximum principles, for example, in density and pressure. Entropy consistency can be naturally guaranteed (Kuzmin 2021) and limiting can be precisely adjusted on subcell level resulting in very sharp shock fronts and detailed small scale flow structures even for challenging setups (Pazner 2021; Rueda-Ramírez et al. 2022b).

Our solver module in FLASH is written in a modular fashion and offers a simple interface, where a huge variety of DG implementations can potentially plug in. If IDP DG turns out to significantly improve the solution quality of strongly compressible turbulence simulation, our FLASH module can readily host such a new solver.

3.) Developing an efficient DG solver for a diffusive ISM model

A fundamental disadvantage of all proposed substitution and blending schemes in this thesis and in the literature, is the inevitable loss of high order accuracy in the solution as soon as the limiter is triggered. Additional conserved quantities, like angular momentum, are generally not preserved by the monotone fallback method. Furthermore, it is questionable if the simultaneous application of multiple fluid solvers within the same simulation is a rigorous way to do numerics. Which weak solution does such a switching or blending scheme converges to? Most certainly not to the superposition of the individual solutions by both schemes, especially considering the high nonlinearity of the hyperbolic conservation laws in question. Frankly speaking, it is an open secret, and is has been mentioned in several places throughout this thesis, that the actual purpose of such monotone fallback schemes is to produce "dirty" but "cheap" dissipation in order to smoothen respectively regularize the solution. In the spirit of artificial viscosity, it is more veracious to directly include diffusion into the fluid model in the first place. These equations are then rigorously discretized by the high order DG machinery leading to a clean and performant implementation, provided restrictive timestep constraints can be efficiently handled by appropriate timestepping approaches, e.g., Dumbser et al. (2008); Gassner et al. (2008); Lörcher et al. (2008); Hidalgo and Dumbser (2011). The idea of operating high order schemes at the resolution limit, as we have done in this thesis, is probably ill-fated and disregards the true power of high order DG schemes with spectral-like accuracy.

Diffusion in ISM models is not an unheard-of phenomenon in the astrophysics community, e.g., Medvedev et al. (2017); Uchaikin and Sibatov (2019); Wang et al. (2021), but still raises suspicion among many astrophysicists. But in fact, this is shortsighted since all of their popular FV methods introduce dissipation through the back door anyway. Any semi-discrete formulation of a robust FV scheme can be rearranged such that the evolution equations split into convection and diffusion terms. Hence, in reality the purely inviscid Euler equations are never solved, but a viscous approximation instead. Of course, for such schemes the viscosity parameter ν scales with the grid parameter, $\nu \propto \Delta x^k$ for $k > 0$, and in the continuum limit, $\Delta x \to 0$, the vanishing viscosity solution of the original inviscid equations is reached. As a matter of fact, Svärd (2018) argues in favor of a compressible diffusive flow model and gives a couple of solid arguments based on physical deliberations. Finally, we want to remind you that the very pioneers in CFD, VonNeumann and Richtmyer (1950), already saw the necessity to stabilize their computations with the proper amount of dissipation.

Appendix A

Appendix

A.1 Satisfying the Element Entropy Condition via Convex Blending (proof-of-concept)

Since we know that the FV scheme with appropriate interface fluxes satisfies the entropy condition (2.15), we can utilize the blending scheme (4.100) in order to compensate for spurious entropy growth induced by the DG scheme:

$$(1-\alpha_q)\,\overline{\dot{\Delta\mathcal{S}}}_q^{\mathrm{FV}} + \alpha_q\,\overline{\dot{\Delta\mathcal{S}}}_q^{\mathrm{DG}} \leq 0. \tag{A.1}$$

Provided that $\overline{\dot{\Delta\mathcal{S}}}_q^{\mathrm{FV}} \leq 0$ we define α_q to be

$$\alpha_q = \frac{\epsilon + \left|\overline{\dot{\Delta\mathcal{S}}}_q^{\mathrm{FV}}\right|}{\epsilon + \left|\overline{\dot{\Delta\mathcal{S}}}_q^{\mathrm{FV}}\right| + \tau_{\mathrm{e}}\ \max\left(0, \overline{\dot{\Delta\mathcal{S}}}_q^{\mathrm{DG}}\right)} \tag{A.2}$$

with a free parameter $\tau_{\mathrm{e}} \in (1, \infty)$, which amplifies the sensitivity of the indicator and a compensation factor $\epsilon = 10^{-20}$ avoiding numerical problems in case $\left|\overline{\dot{\Delta\mathcal{S}}}_q^{\mathrm{FV}}\right| \to 0$. Figure A.1 shows the evolution of the entropy production rate for various DG variants stabilized via the blending from above. The parameter τ_{e} had to be tuned individually for each simulation in order to run till the final simulation time $T = 10$. The results shown here are only a proof-of-concept and demand further investigations.

A.1. Satisfying the Element Entropy Condition via Convex Blending (proof-of-concept)

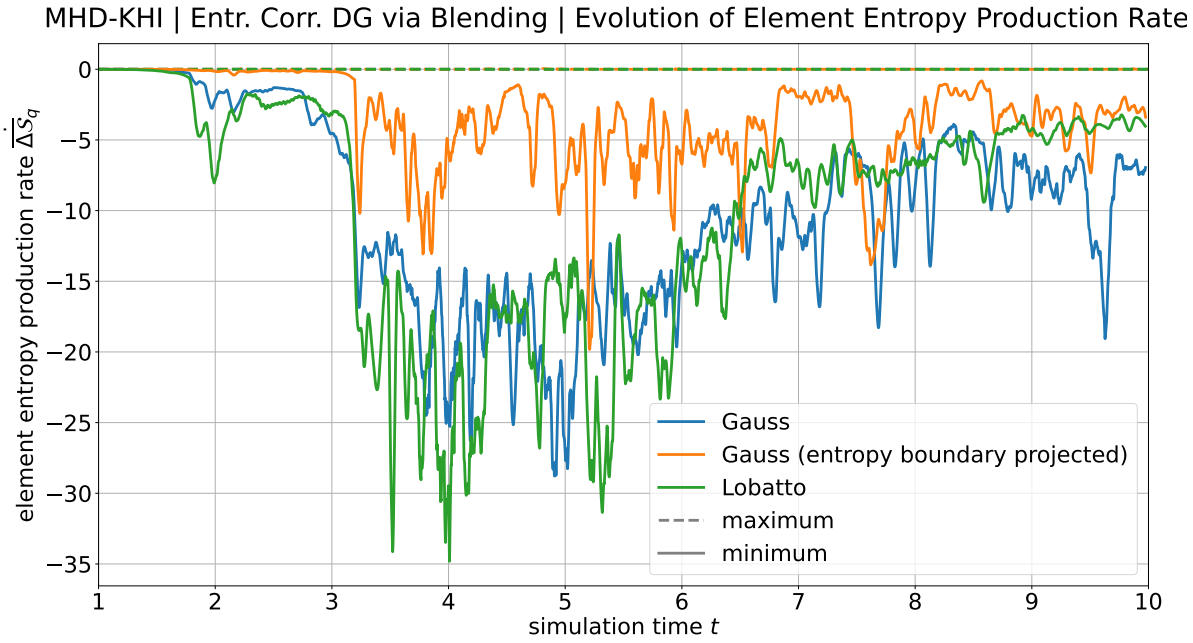

Figure A.1: Evolution of the entropy production rates of MHD-KHI setup and various DG variants stabilized via blending according to approximated entropy errors in each DG element.

A.2 3D MHD Blast run by Bouchut5

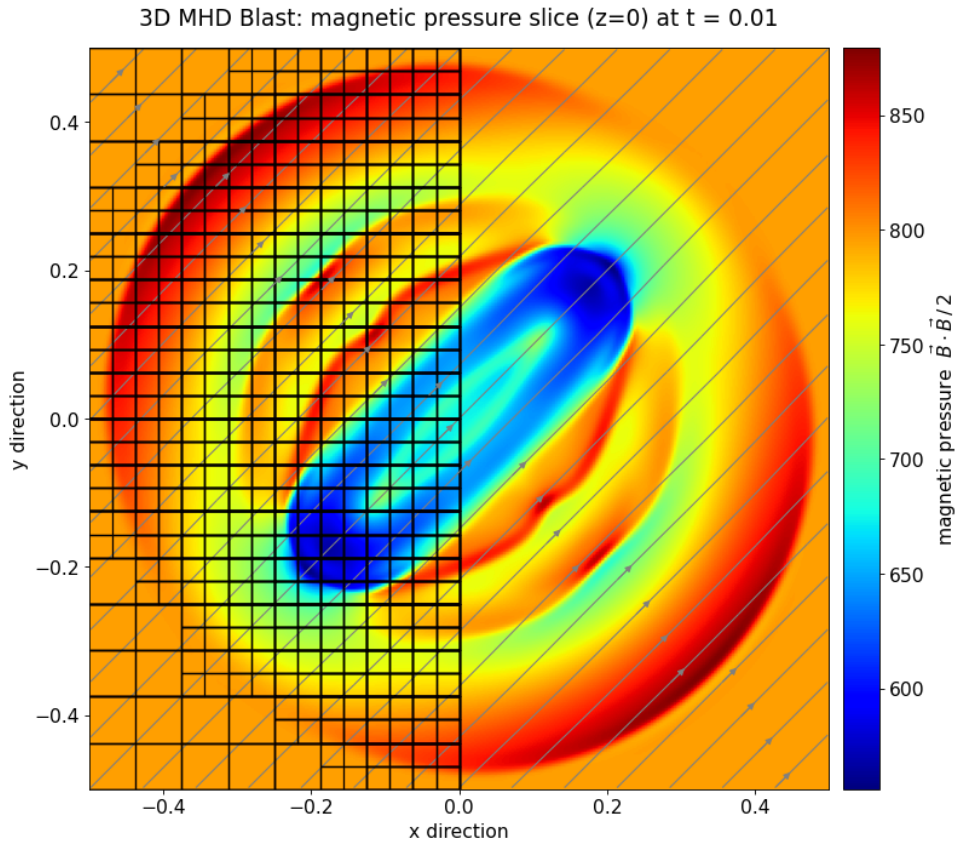

Figure A.2: Slice of the magnetic pressure at constant $z = 0$ solved with the Bouchut5 scheme. The black lines on the left side highlight part of the AMR mesh and the gray stream lines indicate the field spanned by the x- and y-component of the magnetic field. The maximum resolution level is set to 256^3 DOF.

A.3 Young Supernova Remnant from Nemo

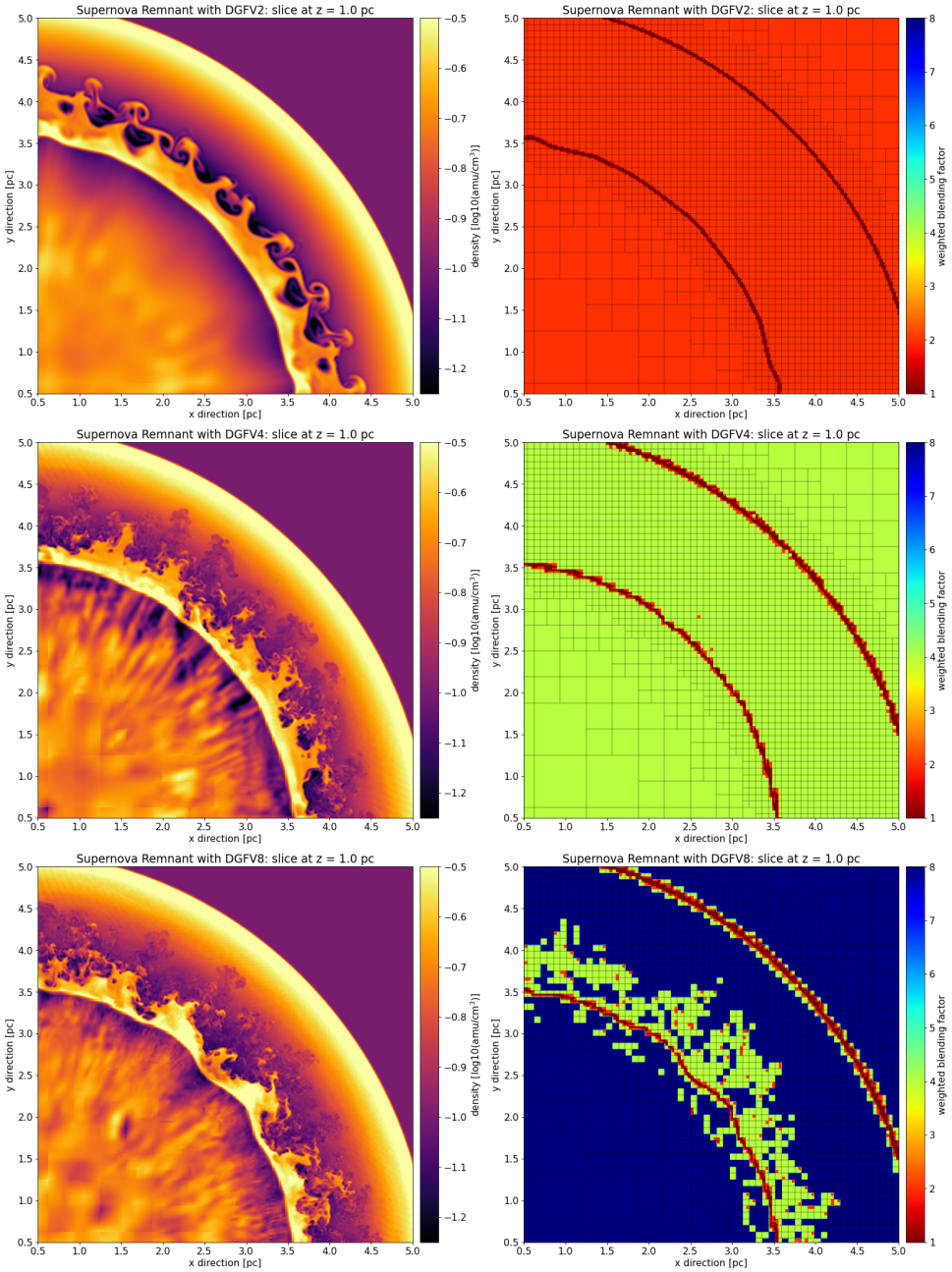

Figure A.3: *left column:* 2D density slice (see figure 7.1) showing the instability region respectively remnant of the supernova at $T = 500\,\mathrm{yr}$ simulated with the (from top to bottom) DGFV2, DGFV4 and DGFV8 scheme. *right column:* 2D slice of the weighted blending factors of the respective blending schemes at $T = 500\,\mathrm{yr}$. The black lines correspond to the element boundaries of the Cartesian non-conforming mesh. Image source: Markert et al. (2021).

A.4 Comparative Study for the MHD Torus Setup

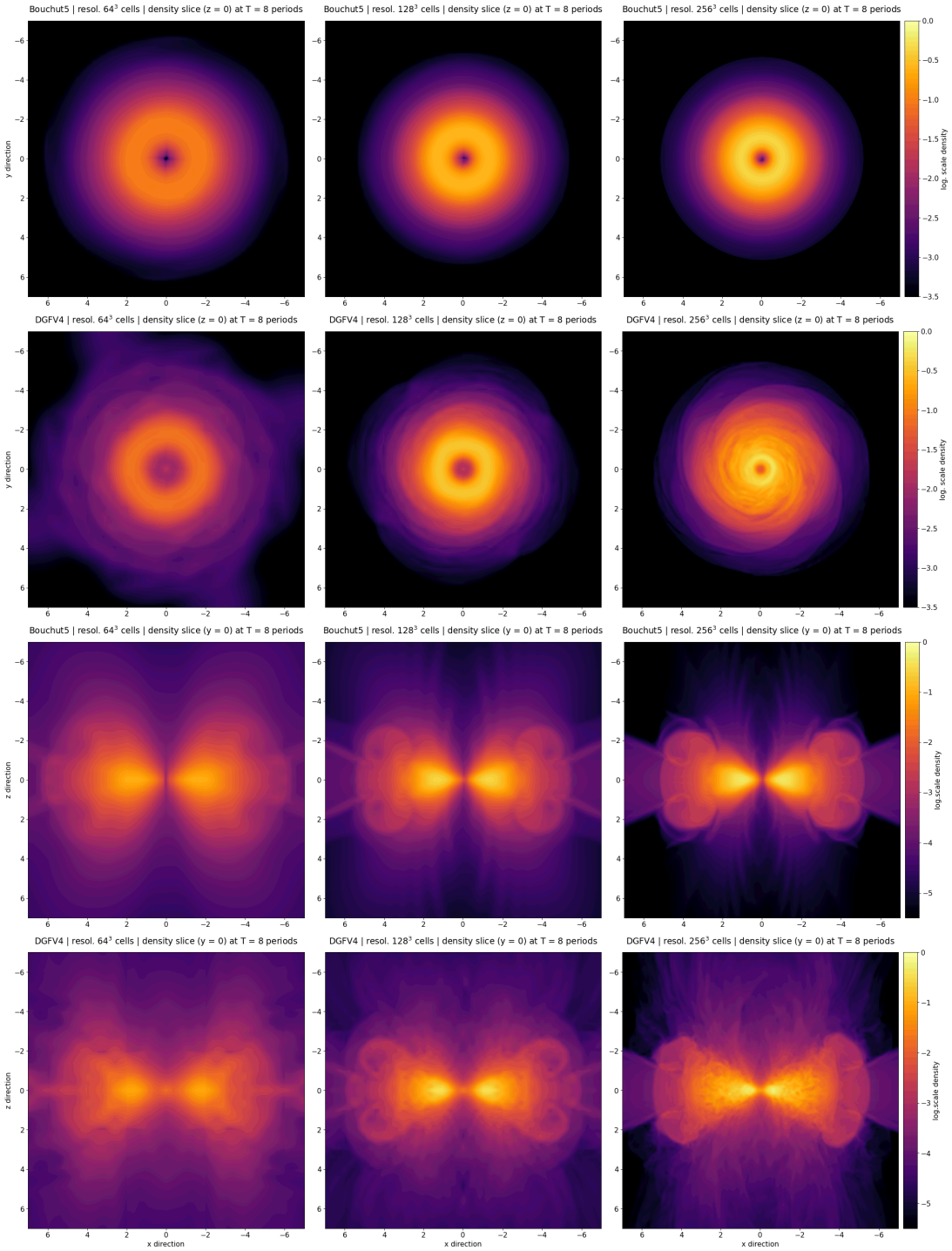

Figure A.4: MHD torus after 8 periods run for max. resolutions (from left to right) 64^3, 128^3, 256^3 and two fluid solvers Bouchut5 (rows 1,3) and DGFV4 (rows 2,4). Shown are log-scale density slices in the x-y plane (rows 1,2) and in the x-z plane (rows 3,4). Image source: Markert et al. (2022).

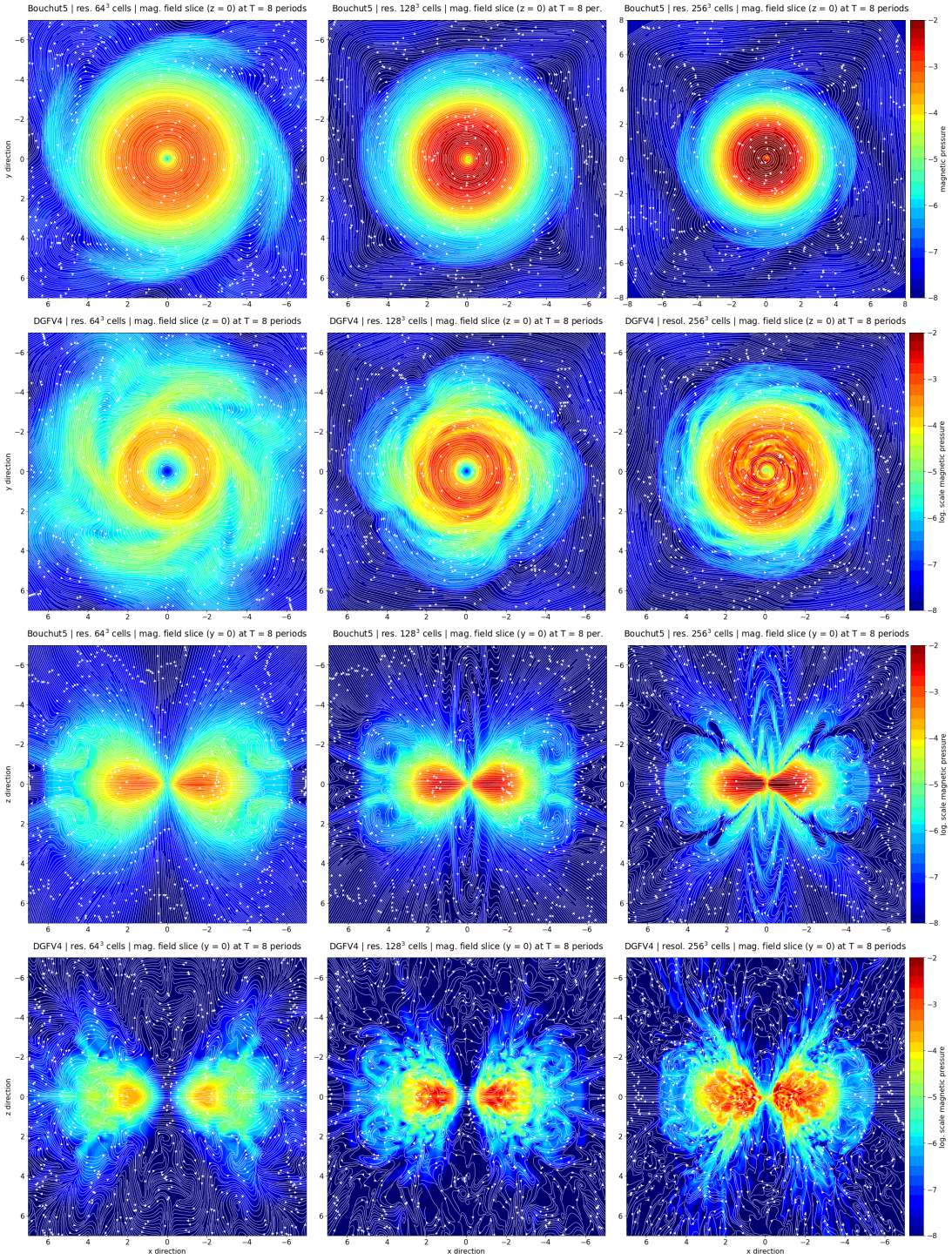

Figure A.5: MHD torus after 8 periods run for max. resolutions (from left to right) 64^3, 128^3, 256^3 and two fluid solvers Bouchut5 (rows 1,3) and DGFV4 (rows 2,4). Shown are log-scale magnetic pressure slices in the x-y plane (rows 1,2) and in the x-z plane (rows 3,4). The streamlines in white denote the magnetic field lines. Image source: Markert et al. (2022).

A.5 Turbulent Expanding HII Region with Bouchut5

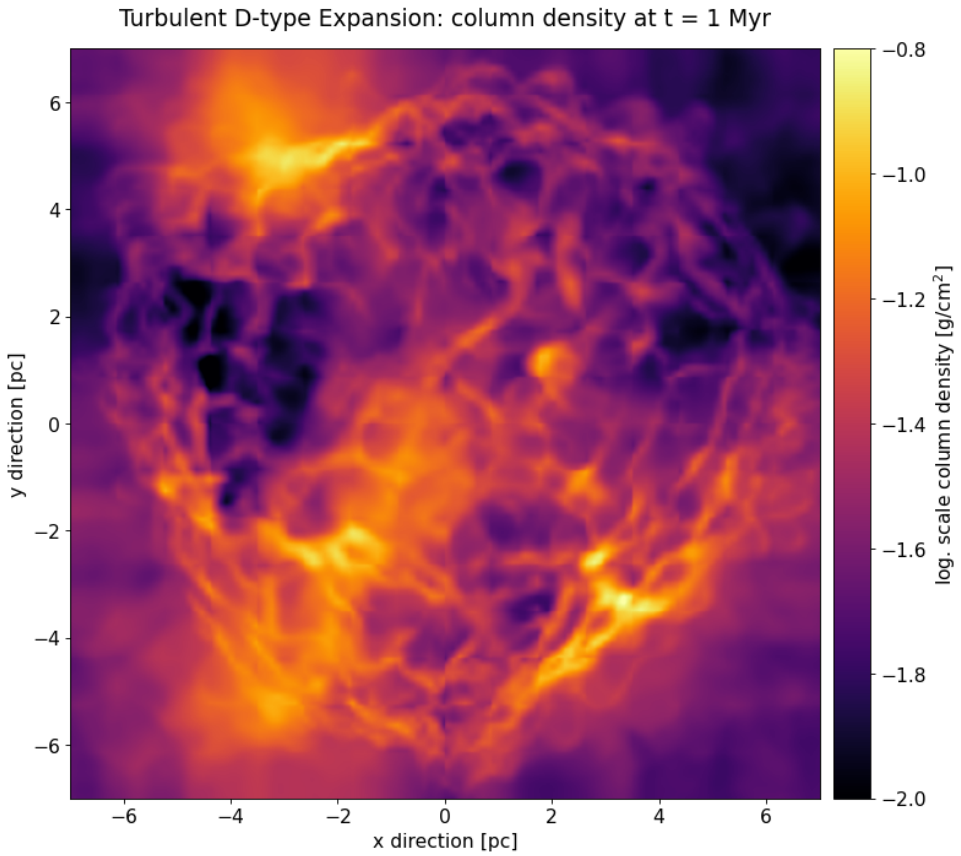

Figure A.6: Turbulent D-type expansion into a fractal medium run by Bouchut5 solver at final simulation time $T = 1$ Myr. Shown is the column density along the z-axis.

A.6 Stellar Feedback into Turbulent ISM

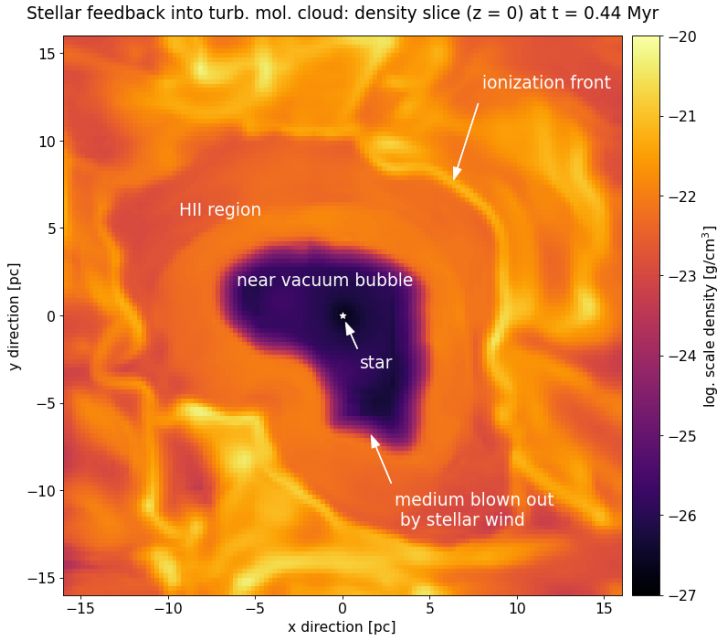

Figure A.7: Stellar feedback simulation run by the DGFV4 solver. Shown is the density slice along the x-y plane with annotations of the various flow features observable in the plot.

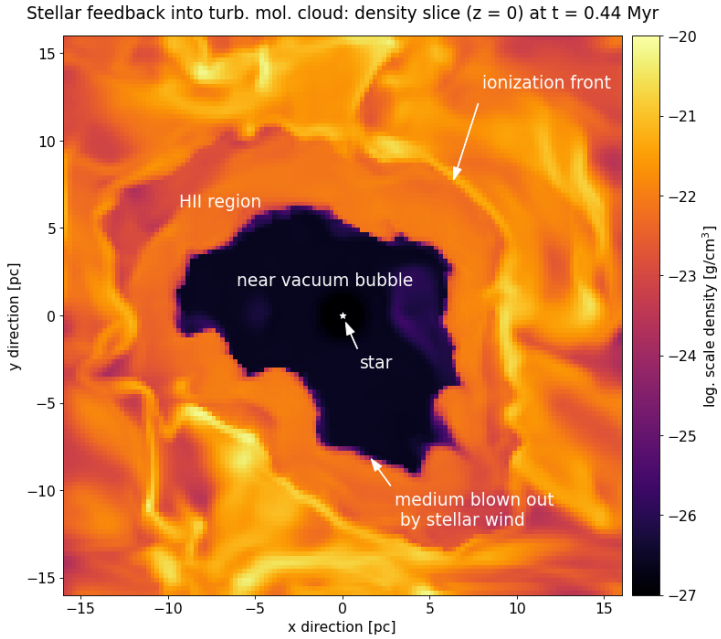

Figure A.8: Stellar feedback simulation run by the Bouchut5 solver. Shown is the density slice along the x-y plane with annotations of the various flow features observable in the plot.

A.7 Cloud in Hot Galactic Wind with Bouchut5

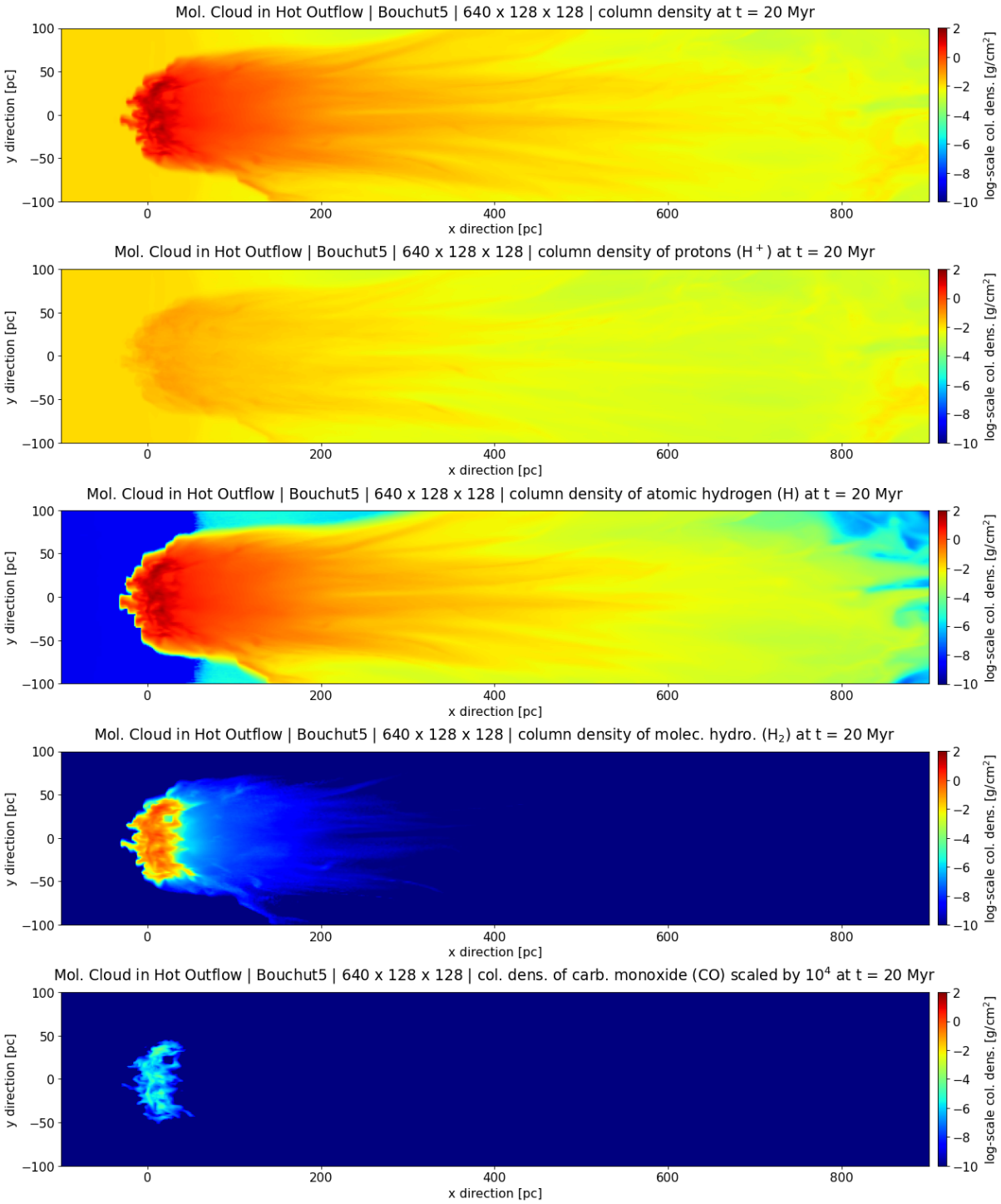

Figure A.9: Molecular cloud within a jet of hot galactic outflow run by the Bouchut5 solver. Shown is the column density along the z-axis for the total mass and different species.

Bibliography

Hesam Abbassi, Farzad Mashayek, and Gustaaf B Jacobs. Shock capturing with entropy-based artificial viscosity for staggered grid discontinuous spectral element method. *Computers & Fluids*, 98:152–163, 2014.

Remi Abgrall. A general framework to construct schemes satisfying additional conservation relations. application to entropy conservative and entropy dissipative schemes. *Journal of Computational Physics*, 372:640–666, 2018.

Milton Abramowitz and Irene A. Stegun. *Handbook of Mathematical Functions: with Formulas, Graphs, and Mathematical Tables*. Dover Publications, 1965.

Oscar Agertz, Ben Moore, Joachim Stadel, Doug Potter, Francesco Miniati, Justin Read, Lucio Mayer, Artur Gawryszczak, Andrey Kravtsov, Åke Nordlund, et al. Fundamental differences between sph and grid methods. *Monthly Notices of the Royal Astronomical Society*, 380(3):963–978, 2007.

Oscar Agertz, Andrey V Kravtsov, Samuel N Leitner, and Nickolay Y Gnedin. Toward a complete accounting of energy and momentum from stellar feedback in galaxy formation simulations. *The Astrophysical Journal*, 770(1):25, 2013.

Mark Ainsworth. Dispersive and dissipative behaviour of high order discontinuous galerkin finite element methods. *Journal of Computational Physics*, 198(1):106–130, 2004.

Gene M Amdahl. Validity of the single processor approach to achieving large scale computing capabilities. In *Proceedings of the April 18-20, 1967, spring joint computer conference*, pages 483–485, 1967.

John David Anderson Jr. *Fundamentals of aerodynamics*. Tata McGraw-Hill Education, 2010.

Steven A Balbus and John F Hawley. A powerful local shear instability in weakly magnetized disks: I. linear analysis. In *Bulletin of the American Astronomical Society*, volume 22, page 1209, 1990.

Dinshaw S Balsara. Second-order-accurate schemes for magnetohydrodynamics with divergence-free reconstruction. *The Astrophysical Journal Supplement Series*, 151(1): 149, 2004.

Dinshaw S Balsara. Higher-order accurate space-time schemes for computational astrophysics—part i: finite volume methods. *Living reviews in computational astrophysics*, 3(1):1–138, 2017.

Dinshaw S Balsara and Roger Käppeli. Von neumann stability analysis of globally divergence-free rkdg schemes for the induction equation using multidimensional riemann solvers. *Journal of Computational Physics*, 336:104–127, 2017.

Dinshaw S Balsara and Daniel S Spicer. A staggered mesh algorithm using high order godunov fluxes to ensure solenoidal magnetic fields in magnetohydrodynamic simulations. *Journal of Computational Physics*, 149(2):270–292, 1999.

Dinshaw S Balsara, Tobias Rumpf, Michael Dumbser, and Claus-Dieter Munz. Efficient, high accuracy ader-weno schemes for hydrodynamics and divergence-free magnetohydrodynamics. *Journal of Computational Physics*, 228(7):2480–2516, 2009.

Wolfgang Bangerth, Carsten Burstedde, Timo Heister, and Martin Kronbichler. Algorithms and data structures for massively parallel generic adaptive finite element codes. *ACM Transactions on Mathematical Software*, 38(2):14:1–14:28, 2011.

Josh Barnes and Piet Hut. A hierarchical o (n log n) force-calculation algorithm. *nature*, 324(6096):446–449, 1986.

Timothy Barth and Dennis Jespersen. The design and application of upwind schemes on unstructured meshes. In *27th Aerospace sciences meeting*, page 366, 1989.

Francesco Bassi, Andrea Crivellini, Stefano Rebay, and Marco Savini. Discontinuous Galerkin solution of the Reynolds-averaged Navier–Stokes and k–ω turbulence model equations. *Computers & Fluids*, 34(4-5):507–540, 2005.

Matthew R Bate, Ian A Bonnell, and Nigel M Price. Modelling accretion in protobinary systems. *Monthly Notices of the Royal Astronomical Society*, 277(2):362–376, 1995.

Andreas Bauer, Kevin Schaal, Volker Springel, Praveen Chandrashekar, Rüdiger Pakmor, and Christian Klingenberg. Simulating turbulence using the astrophysical discontinuous galerkin code tenet. In *Software for Exascale Computing-SPPEXA 2013-2015*, pages 381–402. Springer, 2016.

Andrea D. Beck, Thomas Bolemann, David Flad, Hannes Frank, Gregor J. Gassner, Florian Hindenlang, and Claus-Dieter Munz. High-order discontinuous galerkin spectral element methods for transitional and turbulent flow simulations. *International Journal for Numerical Methods in Fluids*, 76(8):522–548, 2014.

William H Beyer. Crc standard mathematical tables and formulae. *Boca Raton*, 1991.

Stefano Bianchini and Alberto Bressan. Vanishing viscosity solutions of nonlinear hyperbolic systems. *Annals of mathematics*, pages 223–342, 2005.

Thomas G Bisbas, TJ Haworth, RJR Williams, Jonathan Mackey, Pascal Tremblin, AC Raga, SJ Arthur, C Baczynski, JE Dale, T Frostholm, et al. Starbench: the d-type expansion of an h ii region. *Monthly Notices of the Royal Astronomical Society*, 453(2):1324–1343, 2015.

Rupak Biswas, Karen D Devine, and Joseph E Flaherty. Parallel, adaptive finite element methods for conservation laws. *Applied Numerical Mathematics*, 14(1-3):255–283, 1994.

K. Black. A conservative spectral element method for the approximation of compressible fluid flow. *Kybernetika*, 35(1):133–146, 1999.

Andreas Bleuler and Romain Teyssier. Towards a more realistic sink particle algorithm for the ramses code. *Monthly Notices of the Royal Astronomical Society*, 445(4):4015–4036, 2014.

Marvin Bohm. *An entropy stable nodal discontinuous Galerkin method for the resistive MHD equations*. PhD thesis, Universität zu Köln, 2018.

Marvin Bohm, Andrew R. Winters, Gregor J. Gassner, Dominik Derigs, Florian Hindenlang, and Joachim Saur. An entropy stable nodal discontinuous Galerkin method for the resistive MHD equations. Part I: Theory and numerical verification. *Journal of Computational Physics*, July 2018. doi: 10.1016/j.jcp.2018.06.027. URL https://doi.org/10.1016/j.jcp.2018.06.027.

Marvin Bohm, Sven Schermeng, Andrew R. Winters, Gregor J. Gassner, and Gustaaf B. Jacobs. Multi-element SIAC filter for shock capturing applied to high-order discontinuous galerkin spectral element methods. *Journal of Scientific Computing*, 81(2):820–844, August 2019.

François Bouchut, Christian Klingenberg, and Knut Waagan. A multiwave approximate riemann solver for ideal mhd based on relaxation ii: numerical implementation with 3 and 5 waves. *Numerische Mathematik*, 115(4):647–679, 2010.

TJ Boyd, TJM Boyd, and JJ Sanderson. *The physics of plasmas*. Cambridge University Press, 2003.

Jeremiah U Brackbill and Daniel C Barnes. The effect of nonzero ∇ b on the numerical solution of the magnetohydrodynamic equations. *Journal of Computational Physics*, 35(3):426–430, 1980.

Alberto Bressan, Graziano Crasta, and Benedetto Piccoli. Well-posedness of the cauchy problem for n$\times$ n systems of conservation laws. In *Memoirs AMS, no. 694, American Mathematical Society*. Citeseer, 1997.

Moysey Brio and Cheng Chin Wu. An upwind differencing scheme for the equations of ideal magnetohydrodynamics. *Journal of computational physics*, 75(2):400–422, 1988.

Keaton J Burns, Geoffrey M Vasil, Jeffrey S Oishi, Daniel Lecoanet, and Benjamin P Brown. Dedalus: A flexible framework for numerical simulations with spectral methods. *Physical Review Research*, 2(2):023068, 2020.

Carsten Burstedde and Johannes Holke. A tetrahedral space-filling curve for nonconforming adaptive meshes. *SIAM Journal on Scientific Computing*, 38(5):C471–C503, 2016.

Carsten Burstedde and Johannes Holke. Coarse mesh partitioning for tree-based amr. *SIAM Journal on Scientific Computing*, 39(5):C364–C392, 2017.

Carsten Burstedde, Lucas C. Wilcox, and Omar Ghattas. `p4est`: Scalable algorithms for parallel adaptive mesh refinement on forests of octrees. *SIAM Journal on Scientific Computing*, 33(3):1103–1133, 2011. doi: 10.1137/100791634.

Carsten Burstedde, Donna Calhoun, Kyle T. Mandli, and Andy R. Terrel. Forestclaw: Hybrid forest-of-octrees AMR for hyperbolic conservation laws. In Michael Bader, Arndt Bode, Hans-Joachim Bungartz, Michael Gerndt, Gerhard R. Joubert, and Frans Peters, editors, *Parallel Computing: Accelerating Computational Science and Engineering (CSE)*, volume 25 of *Advances in Parallel Computing*, pages 253–262. IOS Press, 2014. doi: 10.3233/978-1-61499-381-0-253.

Claudio Canuto, M Yousuff Hussaini, Alfio Quarteroni, and Thomas A Zang. *Spectral methods: fundamentals in single domains*. Springer Science & Business Media, 2007.

Claudio Canuto, M Yousuff Hussaini, Alfio Quarteroni, A Thomas Jr, et al. *Spectral methods in fluid dynamics*. Springer Science & Business Media, 2012.

M. Carpenter and C. Kennedy. Fourth-order 2N-storage Runge-Kutta schemes. *Appl. Num. Math.*, 35:48–56, 2000.

M. Carpenter, T. Fisher, E. Nielsen, and S. Frankel. Entropy stable spectral collocation schemes for the Navier-Stokes equations: Discontinuous interfaces. *SIAM Journal on Scientific Computing*, 36(5):B835–B867, 2014.

Noel Chalmers, Lilia Krivodonova, and Ruibin Qin. Relaxing the cfl number of the discontinuous galerkin method. *SIAM Journal on Scientific Computing*, 36(4):A2047–A2075, 2014.

Jesse Chan. On discretely entropy conservative and entropy stable discontinuous Galerkin methods. *Journal of Computational Physics*, 362:346–374, 2018.

S Chandrasekhar. The stability of non-dissipative couette flow in hydromagnetics. *Proceedings of the National Academy of Sciences of the United States of America*, 46(2): 253, 1960.

Praveen Chandrashekar. Kinetic energy preserving and entropy stable finite volume schemes for compressible Euler and Navier-Stokes equations. *Communications in Computational Physics*, 14:1252–1286, 11 2013.

Praveen Chandrashekar and Christian Klingenberg. Entropy stable finite volume scheme for ideal compressible mhd on 2-d cartesian meshes. *SIAM Journal on Numerical Analysis*, 54(2):1313–1340, 2016.

Praveen Chandrashekar, Juan Pablo Gallego-Valencia, and Christian Klingenberg. A runge–kutta discontinuous galerkin scheme for the ideal magnetohydrodynamical model. In *XVI International Conference on Hyperbolic Problems: Theory, Numerics, Applications*, pages 335–344. Springer, 2016.

Tianheng Chen and Chi-Wang Shu. Entropy stable high order discontinuous Galerkin methods with suitable quadrature rules for hyperbolic conservation laws. *Journal of Computational Physics*, 345:427–461, 2017.

Roger A Chevalier. Self-similar solutions for the interaction of stellar ejecta with an external medium. *The Astrophysical Journal*, 258:790–797, 1982.

Elisabetta Chiodaroli, Camillo De Lellis, and Ondřej Kreml. Global ill-posedness of the isentropic system of gas dynamics. *Communications on Pure and Applied Mathematics*, 68(7):1157–1190, 2015.

Rudolf Clausius. On the application of the mechanical theory of heat to the steam-129 engine." as found in: Clausius, r.(1865). *The Mechanical Theory of Heat–with its Applications*, 130, 1856a.

Rudolf Clausius. X. on a modified form of the second fundamental theorem in the mechanical theory of heat. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 12(77):81–98, 1856b.

Bernardo Cockburn and Chi-Wang Shu. The Runge-Kutta discontinuous Galerkin method for conservation laws V: Multidimensional systems. *Journal of Computational Physics*, 141(2):199 – 224, 1998. ISSN 0021-9991. doi: DOI:10.1006/jcph.1998.5892. URL http://www.sciencedirect.com/science/article/B6WHY-45J593T-6N/2/34d62c09260f2d7d0af3099d456a3b72.

Bernardo Cockburn, Sunchung Hou, and Chi-Wang Shu. The Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws. IV: The multidimensional case. *Mathematics of Computation*, 54(190):545–581, April 1990.

Bernardo Cockburn, George E Karniadakis, and Chi-Wang Shu. *Discontinuous Galerkin methods: theory, computation and applications*, volume 11. Springer Science & Business Media, 2012.

Phillip Colella and Paul R Woodward. The piecewise parabolic method (ppm) for gas-dynamical simulations. *Journal of computational physics*, 54(1):174–201, 1984.

Phillip Colella, Daniel T Graves, TJ Ligocki, DF Martin, D Modiano, DB Serafini, and B Van Straalen. Chombo software package for amr applications design document. *Available at the Chombo website: http://seesar. lbl. gov/ANAG/chombo/(September 2008)*, 2009.

David C Collins, Hao Xu, Michael L Norman, Hui Li, and Shengtai Li. Cosmological adaptive mesh refinement magnetohydrodynamics with enzo. *The Astrophysical Journal Supplement Series*, 186(2):308, 2010.

John Corban. False-color image of the rosetta nebula taken with hubble space telescope (esa/eso/nasa). https://esahubble.org/projects/fits_liberator/fitsimages/john_corban_12/.

Richard Courant, Kurt Friedrichs, and Hans Lewy. Über die partiellen differenzengleichungen der mathematischen physik. *Mathematische annalen*, 100(1):32–74, 1928.

Constantine M Dafermos. *Hyperbolic conservation laws in continuum physics*, volume 3. Springer, 2005.

James E Dale and Ian Bonnell. Ionizing feedback from massive stars in massive clusters: fake bubbles and untriggered star formation. *Monthly Notices of the Royal Astronomical Society*, 414(1):321–328, 2011.

Christopher Daley, Marcos Vanella, Anshu Dubey, Klaus Weide, and Elias Balaras. Optimization of multigrid based elliptic solver for large scale simulations in the flash code. *Concurrency and Computation: Practice and Experience*, 24(18):2346–2361, 2012.

Andreas Dedner, Friedemann Kemm, Dietmar Kröner, C-D Munz, Thomas Schnitzer, and Matthias Wesenberg. Hyperbolic divergence cleaning for the mhd equations. *Journal of Computational Physics*, 175(2):645–673, 2002.

L Deharveng, B Lefloch, S Kurtz, D Nadeau, M Pomares, J Caplan, and Annie Zavagno. Triggered massive-star formation on the borders of galactic h ii regions-iv. star formation at the periphery of sh2-212. *Astronomy & Astrophysics*, 482(2):585–596, 2008.

Dominik Derigs, Andrew R Winters, Gregor J Gassner, and Stefanie Walch. A novel high-order, entropy stable, 3d amr mhd solver with guaranteed positive pressure. *Journal of Computational Physics*, 317:223–256, 2016.

Dominik Derigs, Andrew R Winters, Gregor J Gassner, and Stefanie Walch. A novel averaging technique for discrete entropy-stable dissipation operators for ideal MHD. *Journal of Computational Physics*, 330:624–632, 2017.

Dominik Derigs, Andrew R Winters, Gregor J Gassner, Stefanie Walch, and Marvin Bohm. Ideal glm-mhd: About the entropy consistent nine-wave magnetic field divergence diminishing ideal magnetohydrodynamics equations. *Journal of Computational Physics*, 364:420–467, 2018.

Bruno Després and Emmanuel Labourasse. Angular momentum preserving cell-centered lagrangian and eulerian schemes on arbitrary grids. *Journal of Computational Physics*, 290:28–54, 2015.

Veselin A Dobrev, Tzanio V Kolev, and Robert N Rieben. High-order curvilinear finite element methods for lagrangian hydrodynamics. *SIAM Journal on Scientific Computing*, 34(5):B606–B641, 2012.

Jack Dongarra, Thomas Herault, and Yves Robert. Fault tolerance techniques for high-performance computing. In *Fault-tolerance techniques for high-performance computing*, pages 3–85. Springer, 2015.

Bruce T Draine and Christopher F McKee. Theory of interstellar shocks. *Annual review of astronomy and astrophysics*, 31(1):373–432, 1993.

Michael Dumbser and Raphael Loubere. A simple robust and accurate a posteriori sub-cell finite volume limiter for the discontinuous galerkin method on unstructured meshes. *Journal of Computational Physics*, 319:163–199, August 2016. doi: 10.1016/j.jcp.2016.05.002. URL https://doi.org/10.1016/j.jcp.2016.05.002.

Michael Dumbser and Raphaël Loubère. A simple robust and accurate a posteriori sub-cell finite volume limiter for the discontinuous galerkin method on unstructured meshes. *Journal of Computational Physics*, 319:163–199, 2016.

Michael Dumbser, Jean-Marc Moschetta, and Jérémie Gressier. A matrix stability analysis of the carbuncle phenomenon. *Journal of Computational Physics*, 197(2):647–670, 2004.

Michael Dumbser, Cedric Enaux, and Eleuterio F Toro. Finite volume schemes of very high order of accuracy for stiff hyperbolic balance laws. *Journal of Computational Physics*, 227(8):3971–4001, 2008.

Michael Dumbser, Olindo Zanotti, Raphael Loubere, and Steven Diot. A posteriori subcell limiting of the discontinuous galerkin finite element method for hyperbolic conservation laws. *Journal of Computational Physics*, 278:47–75, December 2014.

Michael Dumbser, Francesco Fambri, Maurizio Tavelli, Michael Bader, and Tobias Weinzierl. Efficient implementation of ader discontinuous galerkin schemes for a scalable hyperbolic pde engine. *axioms*, 7(3):63, 2018.

Dale R Durran. *Numerical methods for fluid dynamics: With applications to geophysics*, volume 32. Springer Science & Business Media, 2010.

John Edward Dyson and David A Williams. *The physics of the interstellar medium*. CRC Press, 2020.

Gerhard Dziuk. *Theorie und Numerik partieller Differentialgleichungen*. de Gruyter, 2010.

Danijela Efnusheva, Ana Cholakoska, and Aristotel Tentov. A survey of different approaches for overcoming the processor-memory bottleneck. *AIRCC's International Journal of Computer Science and Information Technology*, 9(2):151–163, 2017.

Bruce G Elmegreen and John Scalo. Interstellar turbulence i: observations and processes. *Annu. Rev. Astron. Astrophys.*, 42:211–273, 2004.

Björn Engquist and Stanley Osher. One-sided difference approximations for nonlinear conservation laws. *Mathematics of Computation*, 36(154):321–351, 1981.

Bjorn Engquist, Ami Harten, and Stanley Osher. A high order essentially non-oscillatory shock capturing method. In *Large Scale Scientific Computing*, pages 197–208. Springer, 1987.

K. Eriksen. Long exposure observation of the tycho supernova remnant by chandra x-ray observatory (nasa/cxc/rutgers). http://chandra.harvard.edu/photo/2011/tycho/, 2011.

Charles R Evans and John F Hawley. Simulation of magnetohydrodynamic flows-a constrained transport method. *The Astrophysical Journal*, 332:659–677, 1988.

August E Evrard. Beyond n-body-3d cosmological gas dynamics. *Monthly Notices of the Royal Astronomical Society*, 235:911–934, 1988.

E Falgarone, TG Phillips, and Christopher K Walker. The edges of molecular clouds-fractal boundaries and density structure. *The Astrophysical Journal*, 378:186–201, 1991.

E Falgarone, TH Troland, RM Crutcher, and G Paubert. Cn zeeman measurements in star formation regions. *Astronomy & Astrophysics*, 487(1):247–252, 2008.

Gilles Ferrand, Anne Decourchelle, and Samar Safi-Harb. Three-dimensional simulations of the thermal x-ray emission from young supernova remnants including efficient particle acceleration. *The Astrophysical Journal*, 760(1):34, 2012.

Katia M Ferriere. The interstellar environment of our galaxy. *Reviews of Modern Physics*, 73(4):1031, 2001.

Joel H Ferziger, Milovan Perić, and Robert L Street. *Computational methods for fluid dynamics*, volume 3. Springer, 2002.

Travis C Fisher and Mark H Carpenter. High-order entropy stable finite difference schemes for nonlinear conservation laws: Finite domains. *Journal of Computational Physics*, 252: 518–557, 2013.

Travis C Fisher, Mark H Carpenter, Jan Nordström, Nail K Yamaleev, and Charles Swanson. Discretely conservative finite-difference formulations for nonlinear conservation laws in split form: Theory and boundary conditions. *Journal of Computational Physics*, 234:353–375, 2013.

Ulrik S Fjordholm, Siddhartha Mishra, and Eitan Tadmor. Arbitrarily high-order accurate entropy stable essentially nonoscillatory schemes for systems of conservation laws. *SIAM Journal on Numerical Analysis*, 50(2):544–573, 2012.

David Flad and Gregor Gassner. On the use of kinetic energy preservaing DG-scheme for large eddy simulation. *Journal of Computational Physics*, 350:782–795, 2017.

Joseph E Flaherty, Lilia Krivodonova, Jean-Francois Remacle, and Mark S Shephard. Aspects of discontinuous galerkin methods for hyperbolic conservation laws. *Finite Elements in Analysis and Design*, 38(10):889–908, 2002.

Federico Fraschetti, Romain Teyssier, Jean Ballet, and Anne Decourchelle. Simulation of the growth of the 3d rayleigh-taylor instability in supernova remnants using an expanding reference frame. *Astronomy & Astrophysics*, 515:A104, 2010.

Lucas Friedrich, Andrew R. Winters, David C. Del Rey Fernández, Gregor J. Gassner, Matteo Parsani, and Mark H. Carpenter. An entropy stable h/p non-conforming discontinuous Galerkin method with the summation-by-parts property. *Journal of Scientific Computing*, 77(2):689–725, May 2018. URL https://doi.org/10.1007/s10915-018-0733-7.

Lucas Friedrich, Gero Schnücke, Andrew R Winters, David C Fernández, Gregor J Gassner, and Mark H Carpenter. Entropy stable space–time discontinuous galerkin schemes with summation-by-parts property for hyperbolic conservation laws. *Journal of Scientific Computing*, 80(1):175–222, 2019.

Sébastien Fromang, Patrick Hennebelle, and Romain Teyssier. A high order godunov scheme with constrained transport and adaptive mesh refinement for astrophysical magnetohydrodynamics. *Astronomy & Astrophysics*, 457(2):371–384, 2006.

Bruce Fryxell, Kevin Olson, Paul Ricker, FX Timmes, Michael Zingale, DQ Lamb, Peter MacNeice, Robert Rosner, JW Truran, and H Tufo. Flash: An adaptive mesh hydrodynamics code for modeling astrophysical thermonuclear flashes. *The Astrophysical Journal Supplement Series*, 131(1):273, 2000.

Evghenii Gaburov and Keigo Nitadori. Astrophysical weighted particle magnetohydrodynamics. *Monthly Notices of the Royal Astronomical Society*, 414(1):129–154, 2011.

Juan Pablo Gallego Valencia. *On Runge-Kutta discontinuous Galerkin methods for compressible Euler equations and the ideal magneto-hydrodynamical model.* PhD thesis, Universität Würzburg, 2017.

Anirban Garai, Laslo Diosady, Scott Murman, and Nateri Madavan. Dns of flow in a low-pressure turbine cascade using a discontinuous-galerkin spectral-element method. In *Turbo Expo: Power for Land, Sea, and Air*, volume 56642, page V02BT39A023. American Society of Mechanical Engineers, 2015.

Thomas A Gardiner and James M Stone. An unsplit godunov method for ideal mhd via constrained transport. *Journal of Computational Physics*, 205(2):509–539, 2005.

G. Gassner. A skew-symmetric discontinuous Galerkin spectral element discretization and its relation to SBP-SAT finite difference methods. *SIAM Journal on Scientific Computing*, 35(3):A1233–A1253, 2013.

G.J. Gassner, A.R. Winters, and David A. Kopriva. Split form nodal discontinuous Galerkin schemes with summation-by-parts property for the compressible Euler equations. *Journal of Computational Physics*, 327:39–66, 2016a.

Gregor Gassner and David A Kopriva. A comparison of the dispersion and dissipation errors of gauss and gauss–lobatto discontinuous galerkin spectral element methods. *SIAM Journal on Scientific Computing*, 33(5):2560–2579, 2011.

Gregor Gassner, Frieder Lörcher, and C-D Munz. A discontinuous galerkin scheme based on a space-time expansion ii. viscous flow equations in multi dimensions. *Journal of Scientific Computing*, 34(3):260–286, 2008.

Gregor Gassner, Marc Staudenmaier, Florian Hindenlang, Muhammed Atak, and Claus-Dieter Munz. A space–time adaptive discontinuous galerkin scheme. *Computers & Fluids*, 117:247–261, August 2015.

Gregor J. Gassner. A kinetic energy preserving nodal discontinuous Galerkin spectral element method. *International Journal for Numerical Methods in Fluids*, 76(1):28–50, 2014.

Gregor J. Gassner and Andrea D. Beck. On the accuracy of high-order discretizations for underresolved turbulence simulations. *Theoretical and Computational Fluid Dynamics*, 27(3–4):221–237, 2013.

Gregor J Gassner and Andrew R Winters. A novel robust strategy for discontinuous galerkin methods in computational fluid mechanics: Why? when? what? where? *Frontiers in Physics*, page 612, 2021.

Gregor J Gassner, Florian Hindenlang, and Claus-Dieter Munz. A runge-kutta based discontinuous galerkin method with time accurate local time stepping. In *Adaptive High-Order Methods in Computational Fluid Dynamics*, pages 95–118. World Scientific, 2011.

Gregor J. Gassner, Andrew R. Winters, and David A. Kopriva. A well balanced and entropy conservative discontinuous Galerkin spectral element method for the shallow water equations. *Applied Mathematics and Computation*, 272, Part 2:291–308, 2016b.

Gregor J Gassner, Andrew R Winters, and David A Kopriva. Split form nodal discontinuous galerkin schemes with summation-by-parts property for the compressible euler equations. *Journal of Computational Physics*, 327:39–66, 2016c.

Gregor J Gassner, Andrew R Winters, and David A Kopriva. Split form nodal discontinuous galerkin schemes with summation-by-parts property for the compressible euler equations. *Journal of Computational Physics*, 327:39–66, 2016d.

Gregor J. Gassner, Andrew R. Winters, Florian J. Hindenlang, and David A. Kopriva. The BR1 scheme is stable for the compressible Navier–Stokes equations. *Journal of Scientific Computing*, Apr 2018. ISSN 1573-7691. doi: 10.1007/s10915-018-0702-1. URL https://doi.org/10.1007/s10915-018-0702-1.

A Gatto, S Walch, M-M Mac Low, T Naab, P Girichidis, SCO Glover, R Wünsch, RS Klessen, PC Clark, C Baczynski, et al. Modelling the supernova-driven ism in different environments. *Monthly Notices of the Royal Astronomical Society*, 449(1): 1057–1075, 2015.

Sam Geen, Joakim Rosdahl, Jeremy Blaizot, Julien Devriendt, and Adrianne Slyz. A detailed study of feedback from a massive star. *Monthly Notices of the Royal Astronomical Society*, 448(4):3248–3264, 2015.

Philipp Girichidis, Christoph Federrath, Robi Banerjee, and Ralf S Klessen. Importance of the initial conditions for star formation–i. cloud evolution and morphology. *Monthly Notices of the Royal Astronomical Society*, 413(4):2741–2759, 2011.

Philipp Girichidis, Thorsten Naab, Stefanie Walch, and Thomas Berlok. The in situ formation of molecular and warm ionized gas triggered by hot galactic outflows. *Monthly Notices of the Royal Astronomical Society*, 505(1):1083–1104, 2021.

Simon CO Glover and Paul C Clark. Molecular cooling in the diffuse interstellar medium. *Monthly Notices of the Royal Astronomical Society*, 437(1):9–20, 2014.

Simon CO Glover and Mordecai-Mark Mac Low. Simulating the formation of molecular clouds. i. slow formation by gravitational collapse from static initial conditions. *The Astrophysical Journal Supplement Series*, 169(2):239, 2007.

Edwige Godlewski and Pierre-Arnaud Raviart. *Numerical approximation of hyperbolic systems of conservation laws*, volume 118. Springer Science & Business Media, 2013.

Sergei Konstantinovich Godunov. A difference scheme for numerical solution of discontinuous solution of hydrodynamic equations. *Math. Sbornik*, 47:271–306, 1959.

Sergei Konstantinovich Godunov. An interesting class of quasilinear systems. In *Dokl. Acad. Nauk SSSR*, volume 139, pages 521–523, 1961.

SK Godunov. Symmetric form of the equations of magnetohydrodynamics. *Numerical Methods for Mechanics of Continuum Medium*, 1:26–34, 1972.

Sigal Gottlieb, Chi-Wang Shu, and Eitan Tadmor. Strong stability-preserving high-order time discretization methods. *SIAM review*, 43(1):89–112, 2001.

Ayoub Gouasmi, Karthik Duraisamy, and Scott M Murman. Formulation of entropy-stable schemes for the multicomponent compressible euler equations. *Computer Methods in Applied Mechanics and Engineering*, 363:112912, 2020.

Jean-Luc Guermond, Murtazo Nazarov, Bojan Popov, and Ignacio Tomas. Second-order invariant domain preserving approximation of the euler equations using convex limiting. *SIAM Journal on Scientific Computing*, 40(5):A3211–A3239, 2018.

Jean-Luc Guermond, Bojan Popov, and Ignacio Tomas. Invariant domain preserving discretization-independent schemes and convex limiting for hyperbolic systems. *Computer Methods in Applied Mechanics and Engineering*, 347:143–175, 2019.

Thomas Guillet, Rüdiger Pakmor, Volker Springel, Praveen Chandrashekar, and Christian Klingenberg. High-order magnetohydrodynamics for astrophysics with an adaptive mesh refinement discontinuous galerkin scheme. *Monthly Notices of the Royal Astronomical Society*, 485(3):4209–4246, 2019.

Wei Guo, Ramachandran D. Nair, and Xinghui Zhong. An efficient WENO limiter for discontinuous galerkin transport scheme on the cubed sphere. *International Journal for Numerical Methods in Fluids*, 81(1):3–21, September 2015.

Ami Harten. High resolution schemes for hyperbolic conservation laws. *Journal of computational physics*, 135(2):260–278, 1997.

Ami Harten, Peter D Lax, C David Levermore, and William J Morokoff. Convex entropies and hyperbolicity for general euler equations. *SIAM journal on numerical analysis*, 35 (6):2117–2127, 1998.

Amiram Harten. On the symmetric form of systems of conservation laws with entropy. *Journal of Computational Physics*, 49:151–164, 1983.

Amiram Harten, Peter D Lax, and Bram van Leer. On upstream differencing and godunov-type schemes for hyperbolic conservation laws. *SIAM review*, 25(1):35–61, 1983.

Xijun He, Dinghui Yang, Xueyuan Huang, and Xiao Ma. A numerical dispersion-dissipation analysis of discontinuous galerkin methods based on quadrilateral and triangular elements. *Geophysics*, 85(3):T101–T121, 2020.

Christiane Helzel, James A Rossmanith, and Bertram Taetz. An unstaggered constrained transport method for the 3d ideal magnetohydrodynamic equations. *Journal of Computational Physics*, 230(10):3803–3829, 2011.

Patrick Hennebelle and Edith Falgarone. Turbulent molecular clouds. *The Astronomy and Astrophysics Review*, 20(1):1–58, 2012.

Sebastian Hennemann, Andrés M Rueda-Ramírez, Florian J Hindenlang, and Gregor J Gassner. A provably entropy stable subcell shock capturing approach for high order split form dg for the compressible euler equations. *Journal of Computational Physics*, 426:109935, 2021.

J. S. Hesthaven and T. Warburton. *Nodal Discontinuous Galerkin Methods*. Springer, 2008.

Jan Hesthaven and Robert Kirby. Filtering in legendre spectral methods. *Mathematics of Computation*, 77(263):1425–1452, 2008.

Jan S Hesthaven and Tim Warburton. *Nodal discontinuous Galerkin methods: algorithms, analysis, and applications*. Springer Science & Business Media, 2007.

Arturo Hidalgo and Michael Dumbser. Ader schemes for nonlinear systems of stiff advection–diffusion–reaction equations. *Journal of Scientific Computing*, 48(1):173–189, 2011.

Florian J Hindenlang, Gregor J Gassner, Christoph Altmann, Andrea Beck, Marc Staudenmaier, and Claus-Dieter Munz. Explicit discontinuous galerkin methods for unsteady problems. *Computers & Fluids*, 61:86–93, 2012.

Johannes Holke, David Knapp, and Carsten Burstedde. An optimized, parallel computation of the ghost layer for adaptive hybrid forest meshes. *SIAM Journal on Scientific Computing*, 43(6):C359–C385, 2021.

DA Hubber, S Walch, and AP Whitworth. An improved sink particle algorithm for sph simulations. *Monthly Notices of the Royal Astronomical Society*, 430(4):3261–3275, 2013.

Tobin Isaac and Matthew G. Knepley. Support for non-conformal meshes in PETSc's DMPlex interface. *ACM Transactions on Mathematical Software*. URL https://arxiv.org/abs/1508.02470.

Tobin Isaac, Carsten Burstedde, Lucas C. Wilcox, and Omar Ghattas. Recursive algorithms for distributed forests of octrees. *SIAM Journal on Scientific Computing*, 37(5): C497–C531, 2015. doi: 10.1137/140970963.

Farzad Ismail and Philip L Roe. Affordable, entropy-consistent euler flux functions ii: Entropy production at shocks. *Journal of Computational Physics*, 228(15):5410–5436, 2009.

Shi Jin. Runge-kutta methods for hyperbolic conservation laws with stiff relaxation terms. *Journal of Computational Physics*, 122(1):51–67, 1995.

Fredrik Johansson and Marc Mezzarobba. Fast and rigorous arbitrary-precision computation of gauss–legendre quadrature nodes and weights. *SIAM Journal on Scientific Computing*, 40(6):C726–C747, 2018.

Oğuzhan Kadaifçiler. Overcoming the memory wall. 2017.

Smadar Karni. Multicomponent flow calculations by a consistent primitive algorithm. *Journal of Computational Physics*, 112(1):31–43, 1994.

George Em Karniadakis and Spencer J. Sherwin. *Spectral/hp Element Methods for Computational Fluid Dynamics*. Oxford University Press, 2005.

George Em Karniadakis, George Karniadakis, and Spencer Sherwin. *Spectral/hp element methods for computational fluid dynamics*. Oxford University Press on Demand, 2005.

Carl T Kelley. *Solving nonlinear equations with Newton's method*. SIAM, 2003.

Friedemann Kemm. On the origin of divergence errors in mhd simulations and consequences for numerical schemes. *Communications in Applied Mathematics and Computational Science*, 8(1):1–38, 2013.

A. Kenevsky. *High-Order Implicit-Explicit Runge-Kutta Time Integration Schemes and Time-Consistent Filtering in Spectral Methods*. Ph.d. dissertation, Brown University, May 2006.

Lawrence E. Kidder, Scott E. Field, Francois Foucart, Erik Schnetter, Saul A. Teukolsky, Andy Bohn, Nils Deppe, Peter Diener, François Hébert, Jonas Lippuner, Jonah Miller, Christian D. Ott, Mark A. Scheel, and Trevor Vincent. Spectre: A task-based discontinuous galerkin code for relativistic astrophysics. *Journal of Computational Physics*, 335:84–114, 2017.

Jeong-Gyu Kim, Woong-Tae Kim, Eve C Ostriker, and M Aaron Skinner. Modeling uv radiation feedback from massive stars. i. implementation of adaptive ray-tracing method and tests. *The Astrophysical Journal*, 851(2):93, 2017.

R. M. Kirby and G.E. Karniadakis. De-aliasing on non-uniform grids: Algorithms and applications. *Journal of Computational Physics*, 191:249–264, 2003.

Robert M Kirby and Spencer J Sherwin. Aliasing errors due to quadratic nonlinearities on triangular spectral/hp element discretisations. *Journal of engineering mathematics*, 56(3):273–288, 2006.

Ralf S Klessen. Star formation in molecular clouds. *EAS Publications Series*, 51:133–167, 2011.

A. Klöckner, T. Warburton, and J. S. Hesthaven. Viscous shock capturing in a time-explicit discontinuous galerkin method. *Mathematical Modelling of Natural Phenomena*, 6(3):57–83, 2011.

Doyle Knight. *Elements of numerical methods for compressible flows*, volume 19. Cambridge University Press, 2006.

D. A. Kopriva, S. L. Woodruff, and M. Y. Hussaini. Computation of electromagnetic scattering with a non-conforming discontinuous spectral element method. *International Journal for Numerical Methods in Engineering*, 53(1):105–122, 2002.

David A. Kopriva. *Implementing Spectral Methods for Partial Differential Equations*. Scientific Computation. Springer, May 2009a.

David A Kopriva. *Implementing spectral methods for partial differential equations: Algorithms for scientists and engineers*. Springer Science & Business Media, 2009b.

David A. Kopriva. Stability of overintegration methods for nodal discontinuous Galerkin spectral element methods. *Journal of Scientific Computing*, 76(1):426–442, December 2017.

David A Kopriva and Edwin Jimenez. An assessment of the efficiency of nodal discontinuous galerkin spectral element methods. In *Recent Developments in the Numerics of Nonlinear Hyperbolic Conservation Laws*, pages 223–235. Springer, 2013.

David A Kopriva, Florian J Hindenlang, Thomas Bolemann, and Gregor J Gassner. Free-stream preservation for curved geometrically non-conforming discontinuous galerkin spectral elements. *Journal of Scientific Computing*, 79(3):1389–1408, 2019.

Viktor Pavlovich Korobeinikov. *Problems of point blast theory*. Springer Science & Business Media, 1991.

Nico Krais, Andrea Beck, Thomas Bolemann, Hannes Frank, David Flad, Gregor Gassner, Florian Hindenlang, Malte Hoffmann, Thomas Kuhn, Matthias Sonntag, et al. Flexi: A high order discontinuous galerkin framework for hyperbolic–parabolic conservation laws. *Computers & Mathematics with Applications*, 2020.

Lilia Krivodonova. Limiters for high-order discontinuous galerkin methods. *Journal of Computational Physics*, 226(1):879–896, 2007.

Lilia Krivodonova and Ruibin Qin. An analysis of the spectrum of the discontinuous galerkin method. *Applied Numerical Mathematics*, 64:1–18, 2013.

Lilia Krivodonova, Jianguo Xin, J-F Remacle, Nicolas Chevaugeon, and Joseph E Flaherty. Shock detection and limiting with discontinuous galerkin methods for hyperbolic conservation laws. *Applied Numerical Mathematics*, 48(3-4):323–338, 2004.

Mark R Krumholz, Christopher F McKee, and Richard I Klein. Embedding lagrangian sink particles in eulerian grids. *The Astrophysical Journal*, 611(1):399, 2004.

Stanislav N Kružkov. First order quasilinear equations in several independent variables. *Mathematics of the USSR-Sbornik*, 10(2):217, 1970.

Dmitri Kuzmin. Slope limiting for discontinuous galerkin approximations with a possibly non-orthogonal taylor basis. *International Journal for Numerical Methods in Fluids*, 71(9):1178–1190, 2012.

Dmitri Kuzmin. Monolithic convex limiting for continuous finite element discretizations of hyperbolic conservation laws. *Computer Methods in Applied Mechanics and Engineering*, 361:112804, 2020.

Dmitri Kuzmin. Entropy stabilization and property-preserving limiters for p1 discontinuous galerkin discretizations of scalar hyperbolic problems. *Journal of Numerical Mathematics*, 29(4):307–322, 2021.

Leon Lapidus and George F Pinder. *Numerical solution of partial differential equations in science and engineering*. John Wiley & Sons, 2011.

Richard B Larson. Turbulence and star formation in molecular clouds. *Monthly Notices of the Royal Astronomical Society*, 194(4):809–826, 1981.

Peter Lax and Burton Wendroff. Systems of conservation laws. Technical report, LOS ALAMOS NATIONAL LAB NM, 1959.

Peter D Lax. Weak solutions of nonlinear hyperbolic equations and their numerical computation. *Communications on pure and applied mathematics*, 7(1):159–193, 1954.

Peter D Lax. *Hyperbolic systems of conservation laws and the mathematical theory of shock waves*. SIAM, 1973.

Peter D Lax and Robert D Richtmyer. Survey of the stability of linear finite difference equations. *Communications on pure and applied mathematics*, 9(2):267–293, 1956.

Dongwook Lee. A solution accurate, efficient and stable unsplit staggered mesh scheme for three dimensional magnetohydrodynamics. *Journal of Computational Physics*, 243: 269–292, 2013.

FG Lether and PR Wenston. Minimax approximations to the zeros of pn (x) and gauss-legendre quadrature. *Journal of computational and applied mathematics*, 59(2):245–252, 1995.

Randall J LeVeque. *Numerical methods for conservation laws*, volume 214. Springer, 1992.

Randall J. LeVeque. *Finite Volume Methods for Hyperbolic Problems*. Cambridge University Press, 2002.

Randall J LeVeque. *Finite difference methods for ordinary and partial differential equations: steady-state and time-dependent problems*. SIAM, 2007.

Randall J LeVeque et al. *Finite volume methods for hyperbolic problems*, volume 31. Cambridge university press, 2002.

Fengyan Li, Liwei Xu, and Sergey Yakovlev. Central discontinuous galerkin methods for ideal mhd equations with the exactly divergence-free magnetic field. *Journal of Computational Physics*, 230(12):4828–4847, 2011.

Pak Shing Li, Andrew J Cunningham, Brandt L Gaches, Richard I Klein, Mark R Krumholz, Aaron T Lee, Christopher F McKee, Anna L Rosen, Aaron Skinner, et al. Orion2: A magnetohydrodynamics code for star formation. *Journal of Open Source Software*, 6(68):3771, 2021.

Xu-Dong Liu, Stanley Osher, and Tony Chan. Weighted essentially non-oscillatory schemes. *Journal of computational physics*, 115(1):200–212, 1994.

Yong Liu, Chi-Wang Shu, and Mengping Zhang. Entropy stable high order discontinuous Galerkin methods for ideal compressible MHD on structured meshes. *Journal of Computational Physics*, 354:163–178, February 2018a.

Yong Liu, Chi-Wang Shu, and Mengping Zhang. Entropy stable high order discontinuous galerkin methods for ideal compressible mhd on structured meshes. *Journal of Computational Physics*, 354:163–178, 2018b.

Felix J Lockman, Robert A Benjamin, AJ Heroux, and Glen I Langston. The smith cloud: A high-velocity cloud colliding with the milky way. *The Astrophysical Journal Letters*, 679(1):L21, 2008.

Rainald Löhner. An adaptive finite element scheme for transient problems in cfd. *Computer methods in applied mechanics and engineering*, 61(3):323–338, 1987.

Maxime Lombart and Guillaume Laibe. Grain growth for astrophysics with discontinuous Galerkin schemes. *Monthly Notices of the Royal Astronomical Society*, 501(3):4298–4316, 11 2020. ISSN 0035-8711. doi: 10.1093/mnras/staa3682. URL https://doi.org/10.1093/mnras/staa3682.

Frieder Lörcher, Gregor Gassner, and Claus-Dieter Munz. An explicit discontinuous galerkin scheme with local time-stepping for general unsteady diffusion equations. *Journal of Computational Physics*, 227(11):5649–5670, 2008.

Yunguang Lu. *Hyperbolic conservation laws and the compensated compactness method*. CRC Press, 2002.

PC Ma, Y Lv, and M Ihme. Discontinuous galerkin scheme for turbulent flow simulations. *Annual Research Briefs, Center for Turbulence Research, Stanford University*, (1-12), 2015.

Mami Machida, M Hayashi, and R Matsumoto. Three-dimensional global mhd simulations of a torus threaded by toroidal magnetic fields. In *Star Formation 1999*, pages 245–246, 1999.

Gurvan Madec, Romain Bourdallé-Badie, Pierre-Antoine Bouttier, Clement Bricaud, Diego Bruciaferri, Daley Calvert, Jérôme Chanut, Emanuela Clementi, Andrew Coward, Damiano Delrosso, et al. Nemo ocean engine, 2017. URL https://www.nemo-ocean.eu/.

Nihar R Mahapatra and Balakrishna Venkatrao. The processor-memory bottleneck: problems and solutions. *Crossroads*, 5(3es):2, 1999.

Juan Manzanero, Esteban Ferrer, Gonzalo Rubio, and Eusebio Valero. On the role of numerical dissipation in stabilising under-resolved turbulent simulations using discontinuous Galerkin methods. *Journal of Computational Physics*, 2018a.

Juan Manzanero, Gonzalo Rubio, Esteban Ferrer, Eusebio Valero, and David A Kopriva. Insights on aliasing driven instabilities for advection equations with application to gauss–lobatto discontinuous galerkin methods. *Journal of Scientific Computing*, 75 (3):1262–1281, 2018b.

Juan Manzanero, Gonzalo Rubio, and Andrés M Rueda-Ramírez. Horses3d: is a 3d parallel code that uses the high–order discontinuous galerkin spectral element method (dgsem) and is written in modern object–oriented fortran (2008+). https://numath.dmae.upm.es/tools/horses3d, 2019.

Barry Marder. A method for incorporating gauss' law into electromagnetic pic codes. *Journal of Computational Physics*, 68(1):48–55, 1987.

Johannes Markert, Gregor Gassner, and Stefanie Walch. A sub-element adaptive shock capturing approach for discontinuous galerkin methods. *Communications on Applied Mathematics and Computation*, pages 1–43, 2021.

Johannes Markert, Stefanie Walch, and Gregor Gassner. A discontinuous galerkin solver in the flash multiphysics framework. *Monthly Notices of the Royal Astronomical Society*, 511(3):4179–4200, 2022.

PS Medvedev, S Yu Sazonov, and MR Gilfanov. Diffusion of elements in the interstellar medium in early-type galaxies. *Astronomy Letters*, 43(5):285–303, 2017.

Milica Micic, Simon CO Glover, Christoph Federrath, and Ralf S Klessen. Modelling h2 formation in the turbulent interstellar medium: solenoidal versus compressive turbulent forcing. *Monthly Notices of the Royal Astronomical Society*, 421(3):2531–2542, 2012.

A Mignone, M Flock, M Stute, SM Kolb, and G Muscianisi. A conservative orbital advection scheme for simulations of magnetized shear flows with the pluto code. *Astronomy & Astrophysics*, 545:A152, 2012.

Michael S Mock. Systems of conservation laws of mixed type. *Journal of Differential equations*, 37(1):70–88, 1980.

Philip Mocz, Mark Vogelsberger, Debora Sijacki, Rüdiger Pakmor, and Lars Hernquist. A discontinuous galerkin method for solving the fluid and magnetohydrodynamic equations in astrophysical simulations. *Monthly Notices of the Royal Astronomical Society*, 437(1):397–414, 2014.

Scott A Moe, James A Rossmanith, and David C Seal. A simple and effective high-order shock-capturing limiter for discontinuous galerkin methods. *arXiv preprint arXiv:1507.03024*, 2015.

A. Moitinho, A. F. Silva, M. Barros, C. Barata, and H. Savietto. Gaia's sky in colour - equirectangular projection, 2018. URL https://www.esa.int/ESA_Multimedia/Images/2018/04/Gaia_s_sky_in_colour2.

Guy M Morton. A computer oriented geodetic data base and a new technique in file sequencing. 1966.

R C Moura, G Mengaldo, J Peiró, and S J Sherwin. *Journal of Computational Physics*, 330:615–623, 2017.

Ewald Müller and Matthias Steinmetz. Simulating self-gravitating hydrodynamic flows. *Computer Physics Communications*, 89(1-3):45–58, 1995.

Claus-Dieter Munz, Michael Dumbser, and Sabine Roller. Linearized acoustic perturbation equations for low mach number flow with variable density and temperature. *Journal of Computational Physics*, 224(1):352–364, 2007.

François Murat. Compacité par compensation. *Annali della Scuola Normale Superiore di Pisa-Classe di Scienze*, 5(3):489–507, 1978.

Krzysztof Murawski. *Analytical and numerical methods for wave propagation in fluid media*, volume 7 of *A*. World Scientific, 2002.

Scott M Murman, Laslo Diosady, Anirban Garai, and Marco Ceze. A space-time discontinuous-Galerkin approach for separated flows. In *54th AIAA Aerospace Sciences Meeting*, page 1059, 2016.

Richard P Nelson and William D Langer. The dynamics of low-mass molecular clouds in external radiation fields. *The Astrophysical Journal*, 482(2):796, 1997.

Jonatan Núnez-De La Rosa and Claus-Dieter Munz. Xtroem-fv: a new code for computational astrophysics based on very high order finite-volume methods–i. magnetohydrodynamics. *Monthly Notices of the Royal Astronomical Society*, 455(4):3458–3479, 2016.

Jonatan Núñez-de la Rosa and Claus-Dieter Munz. Xtroem-fv: a new code for computational astrophysics based on very high order finite-volume methods–ii. relativistic hydro-and magnetohydrodynamics. *Monthly Notices of the Royal Astronomical Society*, 460(1):535–559, 2016.

Rika Okada, Jun Fukue, and Ryoji Matsumoto. A model of astrophysical tori with magnetic fields. *Publications of the Astronomical Society of Japan*, 41:133–140, 1989.

KM Olson, P MacNeice, B Fryxell, P Ricker, FX Timmes, and M Zingale. Paramesh: a parallel, adaptive mesh refinement toolkit and performance of the asci/flash code. In *American Astronomical Society Meeting Abstracts*, volume 195, pages 42–03, 1999.

Steven A Orszag. On the elimination of aliasing in finite-difference schemes by filtering high-wavenumber components. *Journal of Atmospheric Sciences*, 28(6):1074–1074, 1971.

Steven A Orszag and Cha-Mei Tang. Small-scale structure of two-dimensional magnetohydrodynamic turbulence. *Journal of Fluid Mechanics*, 90(1):129–143, 1979.

Rüdiger Pakmor, Volker Springel, Andreas Bauer, Philip Mocz, Diego J Munoz, Sebastian T Ohlmann, Kevin Schaal, and Chenchong Zhu. Improving the convergence properties of the moving-mesh code arepo. *Monthly Notices of the Royal Astronomical Society*, 455(1):1134–1143, 2016.

Maurizio Pandolfi and Domenic D'Ambrosio. Numerical instabilities in upwind methods: analysis and cures for the "carbuncle" phenomenon. *Journal of Computational Physics*, 166(2):271–301, 2001.

Konstantinos Panourgias and John A Ekaterinaris. A dissipative filter for the discontinuous galerkin method. In *53rd AIAA Aerospace Sciences Meeting*, page 0574, 2015.

M. Parsani, M.H. Carpenter, and E.J. Nielsen. Entropy stable discontinuous interfaces coupling for the three-dimensional compressible Navier-Stokes equations. *Journal of Computational Physics*, 290(C):132–138, 2015.

Matteo Parsani, Mark H. Carpenter, Travis C. Fisher, and Eric J. Nielsen. Entropy stable staggered grid discontinuous spectral collocation methods of any order for the compressible Navier–Stokes equations. *SIAM Journal on Scientific Computing*, 38(5):

A3129–A3162, January 2016. doi: 10.1137/15m1043510. URL https://doi.org/10.1137/15m1043510.

S. Parter. On the legendre-gauss-lobatto points and weights. *Journal of Scientific Computing*, 14(4):347–355, 1999.

Will Pazner. Sparse invariant domain preserving discontinuous galerkin methods with subcell convex limiting. *Computer Methods in Applied Mechanics and Engineering*, 382:113876, 2021.

Will Pazner and Per-Olof Persson. Analysis and entropy stability of the line-based discontinuous Galerkin method. *Journal of Scientific Computing*, 80(1):376–402, 2019.

Boris Peon. Is hinchliffe's rule true? *Submitted to: Annals of Gnosis*, 1988.

Per-Olof Persson and Jaime Peraire. Sub-cell shock capturing for discontinuous galerkin methods. In *44th AIAA Aerospace Sciences Meeting and Exhibit*. American Institute of Aeronautics and Astronautics, January 2006.

Ilya Peshkov and Evgeniy Romenski. A hyperbolic model for viscous newtonian flows. *Continuum Mechanics and Thermodynamics*, 28(1-2):85–104, 2016.

George F Pinder. *Numerical methods for solving partial differential equations: a comprehensive introduction for scientists and engineers*. John Wiley & Sons, 2018.

Kenneth G Powell, Philip L Roe, Timur J Linde, Tamas I Gombosi, and Darren L De Zeeuw. A solution-adaptive upwind scheme for ideal magnetohydrodynamics. *Journal of Computational Physics*, 154(2):284–309, 1999.

Jianxian Qiu and Jun Zhu. RKDG with WENO type limiters. In *Notes on Numerical Fluid Mechanics and Multidisciplinary Design*, pages 67–80. Springer Berlin Heidelberg, 2010.

Jianxian Qiu, Boo Cheong Khoo, and Chi-Wang Shu. A numerical study for the performance of the runge–kutta discontinuous galerkin method based on different numerical fluxes. *Journal of Computational Physics*, 212(2):540–565, 2006.

James J Quirk. A contribution to the great riemann solver debate. In *Upwind and High-Resolution Schemes*, pages 550–569. Springer, 1997.

Hendrik Ranocha. *Generalised summation-by-parts operators and entropy stability of numerical methods for hyperbolic balance laws*. Cuvillier Verlag, 2018.

Hendrik Ranocha, Michael Schlottke-Lakemper, Jesse Chan, Andrés M Rueda-Ramírez, Andrew R Winters, Florian Hindenlang, and Gregor J Gassner. Efficient implementation of modern entropy stable and kinetic energy preserving discontinuous galerkin methods for conservation laws. *arXiv preprint arXiv:2112.10517*, 2021.

Hendrik Ranocha, Michael Schlottke-Lakemper, Andrew Ross Winters, Erik Faulhaber, Jesse Chan, and Gregor Gassner. Adaptive numerical simulations with Trixi.jl: A case study of Julia for scientific computing. *Proceedings of the JuliaCon Conferences*, 1(1): 77, 2022. doi: 10.21105/jcon.00077.

L Rastätter, A Voge, and K Schindler. On current sheets in two-dimensional ideal magnetohydrodynamics caused by pressure perturbations. *Physics of plasmas*, 1(10):3414–3424, 1994.

JN Reddy. *An introduction to the finite element method*, volume 1221. McGraw-Hill New York, 2004.

Anne Reinarz, Dominic E Charrier, Michael Bader, Luke Bovard, Michael Dumbser, Kenneth Duru, Francesco Fambri, Alice-Agnes Gabriel, Jean-Matthieu Gallard, Sven Köppel, et al. Exahype: an engine for parallel dynamically adaptive simulations of wave problems. *Computer Physics Communications*, 254:107251, 2020.

Armin Rest, DL Welch, NB Suntzeff, L Oaster, H Lanning, K Olsen, RC Smith, AC Becker, M Bergmann, P Challis, et al. Scattered-light echoes from the historical galactic supernovae cassiopeia a and tycho (sn 1572). *The Astrophysical Journal*, 681(2):L81, 2008.

Robert D Richtmyer and Keith W Morton. Difference methods for initial-value problems. *Malabar*, 1994.

PM Ricker. A direct multigrid poisson solver for oct-tree adaptive meshes. *The Astrophysical Journal Supplement Series*, 176(1):293, 2008.

William J Rider and Robert B Lowrie. The use of classical lax–friedrichs riemann solvers with discontinuous galerkin methods. *International journal for numerical methods in fluids*, 40(3-4):479–486, 2002.

Philip L Roe. Approximate riemann solvers, parameter vectors, and difference schemes. *Journal of computational physics*, 43(2):357–372, 1981.

Philip L Roe. Characteristic-based schemes for the euler equations. *Annual review of fluid mechanics*, 18(1):337–365, 1986.

James A Rossmanith. High-order discontinuous galerkin finite element methods with globally divergence-free constrained transport for ideal mhd. *arXiv preprint arXiv:1310.4251*, 2013.

Andrés M Rueda-Ramírez, Sebastian Hennemann, Florian J Hindenlang, Andrew R Winters, and Gregor J Gassner. An entropy stable nodal discontinuous galerkin method for the resistive mhd equations. part ii: Subcell finite volume shock capturing. *Journal of Computational Physics*, 444:110580, 2021.

Andrés M Rueda-Ramírez, F. J Hindenlang, and Gregor J Gassner. Entropy-stable gauss collocation methods for ideal magnetohydrodynamics. *arXiv preprint arXiv:2203.06062*, 2022a.

Andrés M Rueda-Ramírez, Will Pazner, and Gregor J Gassner. Subcell limiting strategies for discontinuous galerkin spectral element methods. *arXiv preprint arXiv:2202.00576*, 2022b.

VV Rusanov. Calculation of intersection of non-steady shock waves with obstacles. *J. Comput. Math. Phys. USSR*, 1:267–279, 1961.

Dongsu Ryu, Francesco Miniati, TW Jones, and Adam Frank. A divergence-free upwind code for multidimensional magnetohydrodynamic flows. *The Astrophysical Journal*, 509 (1):244, 1998.

Ravi Samtaney, Dale I Pullin, and Branko Kosović. Direct numerical simulation of decaying compressible turbulence and shocklet statistics. *Physics of Fluids*, 13(5):1415–1430, 2001.

Shankar S Sastry. *Introductory methods of numerical analysis*. PHI Learning Pvt. Ltd., 2012.

Bill Saxton. False-color image of smith's cloud taken with the green bank telescope (nrao/aui/nsf). https://upload.wikimedia.org/wikipedia/commons/8/86/Smith%27s_Cloud_-_2008_-_Bill_Saxton%2C_NRAO%2C_AUI%2C_NSF.jpg, 2008.

John Scalo and Bruce G Elmegreen. Interstellar turbulence ii: implications and effects. *Annu. Rev. Astron. Astrophys.*, 42:275–316, 2004.

Kevin Schaal. *Shocks in the Illustris Universe and Discontinuous Galerkin Hydrodynamics*. PhD thesis, 2016.

Kevin Schaal, Andreas Bauer, Praveen Chandrashekar, Rüdiger Pakmor, Christian Klingenberg, and Volker Springel. Astrophysical hydrodynamics with a high-order discontinuous galerkin scheme and adaptive mesh refinement. *Monthly Notices of the Royal Astronomical Society*, 453(4):4278–4300, 2015.

Michael Schlottke-Lakemper, Gregor J Gassner, Hendrik Ranocha, Andrew R Winters, and Jesse Chan. Trixi.jl: Adaptive high-order numerical simulations of hyperbolic PDEs in Julia. https://github.com/trixi-framework/Trixi.jl, 09 2021.

L Sedov. 1.(1959).-" similarity and dimensional methods in mechanics.". *Translation from 4th Russian edition*, page 228, 1959.

August Segal. *Numerik partieller Differentialgleichungen für Ingenieure*. Springer-Verlag, 2013.

Mohsen Shadmehri and Bruce G Elmegreen. Mass functions in fractal clouds: the role of cloud structure in the stellar initial mass function. *Monthly Notices of the Royal Astronomical Society*, 410(2):788–804, 2011.

Jie Shen, Tao Tang, and Li-Lian Wang. *Spectral methods: algorithms, analysis and applications*, volume 41. Springer Science & Business Media, 2011.

Stuart Merrill Shieber. Is this article consistent with hinchliffe's rule? *Improbable Research*, 2015.

Chi-Wang Shu. Tvb uniformly high-order schemes for conservation laws. *Mathematics of Computation*, 49(179):105–121, 1987.

Chi-Wang Shu and Stanley Osher. Efficient implementation of essentially non-oscillatory shock-capturing schemes. *Journal of computational physics*, 77(2):439–471, 1988.

GailP Smith. A peculiar feature at lii= 40°. 5, bii=-15°. 0. *Bulletin of the Astronomical Institutes of the Netherlands*, 17:203, 1963.

Matthias Sonntag and Claus-Dieter Munz. Shock capturing for discontinuous galerkin methods using finite volume subcells. In *Finite Volumes for Complex Applications VII-Elliptic, Parabolic and Hyperbolic Problems*, pages 945–953. Springer International Publishing, 2014.

Seth C Spiegel, HT Huynh, and James R DeBonis. De-aliasing through over-integration applied to the flux reconstruction and discontinuous galerkin methods. In *22nd AIAA Computational Fluid Dynamics Conference*, page 2744, 2015.

Raymond J Spiteri and Steven J Ruuth. A new class of optimal high-order strong-stability-preserving time discretization methods. *SIAM Journal on Numerical Analysis*, 40(2): 469–491, 2002.

Lyman Spitzer. *Physical processes in the interstellar medium*. John Wiley & Sons, 1978.

Volker Springel. Moving-mesh hydrodynamics with the arepo code. *Proceedings of the International Astronomical Union*, 6(S270):203–206, 2010a.

Volker Springel. Smoothed particle hydrodynamics in astrophysics. *Annual Review of Astronomy and Astrophysics*, 48:391–430, 2010b.

Volker Springel, Naoki Yoshida, and Simon DM White. Gadget: a code for collisionless and gasdynamical cosmological simulations. *New Astronomy*, 6(2):79–117, 2001.

James M Stone and Michael L Norman. Zeus-2d: a radiation magnetohydrodynamics code for astrophysical flows in two space dimensions. i-the hydrodynamic algorithms and tests. *The Astrophysical Journal Supplement Series*, 80:753–790, 1992.

James M Stone, Thomas A Gardiner, Peter Teuben, John F Hawley, and Jacob B Simon. Athena: a new code for astrophysical mhd. *The Astrophysical Journal Supplement Series*, 178(1):137, 2008.

James M Stone, Kengo Tomida, Christopher J White, and Kyle G Felker. The athena++ adaptive mesh refinement framework: Design and magnetohydrodynamic solvers. *The Astrophysical Journal Supplement Series*, 249(1):4, 2020.

Gilbert Strang. On the construction and comparison of difference schemes. *SIAM journal on numerical analysis*, 5(3):506–517, 1968.

Gilbert Strang. Variational crimes in the finite element method. In *The mathematical foundations of the finite element method with applications to partial differential equations*, pages 689–710. Elsevier, 1972.

B Strömgren. unknown. *ApJ*, 89(526):unknown, 1939.

J Stutzki, F Bensch, A Heithausen, V Ossenkopf, and M Zielinsky. On the fractal structure of molecular clouds. *Astronomy and Astrophysics*, 336:697–720, 1998.

Andree Susanto. *High-Order Finite-Volume Schemes for Magnetohydrodynamics*. University of Waterloo, Waterloo, Ontario, Canada, 2014.

Magnus Svärd. A new eulerian model for viscous and heat conducting compressible flows. *Physica A: Statistical Mechanics and its Applications*, 506:350–375, 2018.

Eitan Tadmor. Numerical viscosity and the entropy condition for conservative difference schemes. *Mathematics of Computation*, 43(168):369–381, 1984.

Eitan Tadmor. The numerical viscosity of entropy stable schemes for systems of conservation laws. i. *Mathematics of Computation*, 49(179):91–103, 1987.

Luc Tartar. Compensated compactness and applications to partial differential equations. In *Nonlinear analysis and mechanics: Heriot-Watt symposium*, volume 4, pages 136–212, 1979.

Saul A Teukolsky. Formulation of discontinuous galerkin methods for relativistic astrophysics. *Journal of Computational Physics*, 312:333–356, 2016.

Eleuterio F Toro. *Riemann Solvers and Numerical Methods for Fluid Dynamics*. Springer Verlag, 1999.

Gábor Tóth. The ∇ b= 0 constraint in shock-capturing magnetohydrodynamics codes. *Journal of Computational Physics*, 161(2):605–652, 2000.

Thomas Toulorge and Wim Desmet. Cfl conditions for runge–kutta discontinuous galerkin methods on triangular grids. *Journal of Computational Physics*, 230(12):4657–4678, 2011.

Vladimir V Uchaikin and Renat T Sibatov. Anomalous diffusion in interstellar medium. In *Applications in Physics, Part B*, pages 151–182. Walter de Gruyter GmbH & Co KG, 2019.

Renato Vacondio, Corrado Altomare, Matthieu De Leffe, Xiangyu Hu, David Le Touzé, Steven Lind, Jean-Christophe Marongiu, Salvatore Marrone, Benedict D Rogers, and Antonio Souto-Iglesias. Grand challenges for smoothed particle hydrodynamics numerical schemes. *Computational Particle Mechanics*, 8(3):575–588, 2021.

EP Velikhov. Stability of an ideally conducting liquid flowing between cylinders rotating in a magnetic field. *Sov. Phys. JETP*, 36(9):995–998, 1959.

HK Versteeg and W Malalasekera. An introdution to computational fluid dynamics. *The finite volume method*, 1995.

François Vilar. A posteriori correction of high-order discontinuous galerkin scheme through subcell finite volume formulation and flux reconstruction. *Journal of Computational Physics*, 387:245–279, June 2019.

Damien Violeau. *Fluid mechanics and the SPH method: theory and applications*. Oxford University Press, 2012.

John VonNeumann and Robert D Richtmyer. A method for the numerical calculation of hydrodynamic shocks. *Journal of applied physics*, 21(3):232–237, 1950.

S Walch, P Girichidis, T Naab, A Gatto, SCO Glover, R Wünsch, RS Klessen, PC Clark, T Peters, D Derigs, et al. The silcc (simulating the lifecycle of molecular clouds) project–i. chemical evolution of the supernova-driven ism. *Monthly Notices of the Royal Astronomical Society*, 454(1):238–268, 2015.

SK Walch, AP Whitworth, T Bisbas, R Wünsch, and D Hubber. Dispersal of molecular clouds by ionizing radiation. *Monthly Notices of the Royal Astronomical Society*, 427 (1):625–636, 2012.

Junfeng Wang, Leisa K Townsley, Eric D Feigelson, Patrick S Broos, Konstantin V Getman, Carlos G Román-Zúniga, and Elizabeth Lada. A chandra study of the rosette star-forming complex. i. the stellar population and structure of the young open cluster ngc 2244. *The Astrophysical Journal*, 675(1):464, 2008.

Junfeng Wang, Eric D Feigelson, Leisa K Townsley, Carlos G Román-Zúñiga, Elizabeth Lada, and Gordon Garmire. A chandra study of the rosette star-forming complex. ii. clusters in the rosette molecular cloud. *The Astrophysical Journal*, 696(1):47, 2009.

Sheng-Hao Wang, Kun Fang, Xiao-Jun Bi, and Peng-Fei Yin. Test of the superdiffusion model in the interstellar medium around the geminga pulsar. *Physical Review D*, 103 (6):063035, 2021.

TC Warburton and George Em Karniadakis. A discontinuous galerkin method for the viscous mhd equations. *Journal of computational Physics*, 152(2):608–641, 1999.

Timothy Warburton and Thomas Hagstrom. Taming the cfl number for discontinuous galerkin methods on structured meshes. *SIAM Journal on Numerical Analysis*, 46(6): 3151–3180, 2008.

Maciej Waruszewski, Jeremy E Kozdon, Lucas C Wilcox, Thomas H Gibson, and Francis X Giraldo. Entropy stable discontinuous galerkin methods for balance laws in non-conservative form: Applications to euler with gravity. *arXiv preprint arXiv:2110.15920*, 2021.

Pieter Wesseling. *Principles of computational fluid dynamics*, volume 29. Springer Science & Business Media, 2009.

M Wetzstein, Andrew F Nelson, T Naab, and A Burkert. Vine—a numerical code for simulating astrophysical systems using particles. i. description of the physics and the numerical methods. *The Astrophysical Journal Supplement Series*, 184(2):298, 2009.

Vincent Wheatley, Harish Kumar, and Patrick Huguenot. On the role of riemann solvers in discontinuous galerkin methods for magnetohydrodynamics. *Journal of Computational Physics*, 229(3):660–680, 2010.

Ant Whitworth. The erosion and dispersal of massive molecular clouds by young stars. *Monthly Notices of the Royal Astronomical Society*, 186(1):59–67, 1979.

Samuel Williams, Andrew Waterman, and David Patterson. Roofline: an insightful visual performance model for multicore architectures. *Communications of the ACM*, 52(4): 65–76, 2009.

Niklas Wintermeyer, Andrew R Winters, Gregor J Gassner, and Timothy Warburton. An entropy stable discontinuous Galerkin method for the shallow water equations on curvilinear meshes with wet/dry fronts accelerated by GPUs. *Journal of Computational Physics*, 375:447–480, 2018.

Andrew R Winters and Gregor J Gassner. An entropy stable finite volume scheme for the equations of shallow water magnetohydrodynamics. *Journal of Scientific Computing*, 67(2):514–539, 2016.

Andrew R Winters and David A Kopriva. High-order local time stepping on moving dg spectral element meshes. *Journal of Scientific Computing*, 58(1):176–202, 2014.

Andrew R Winters, Rodrigo C Moura, Gianmarco Mengaldo, Gregor J Gassner, Stefanie Walch, Joaquim Peiro, and Spencer J Sherwin. A comparative study on polynomial dealiasing and split form discontinuous Galerkin schemes for under-resolved turbulence computations. *Journal of Computational Physics*, 372:1–21, 2018.

Richard Wünsch, Stefanie Walch, František Dinnbier, and A Whitworth. Tree-based solvers for adaptive mesh refinement code flash–i: gravity and optical depths. *Monthly Notices of the Royal Astronomical Society*, 475(3):3393–3418, 2018.

Richard Wünsch, Stefanie Walch, František Dinnbier, Daniel Seifried, Sebastian Haid, Andre Klepitko, Anthony P Whitworth, and Jan Palouš. Tree-based solvers for adaptive mesh refinement code flash–ii: radiation transport module treeray. *Monthly Notices of the Royal Astronomical Society*, 505(3):3730–3754, 2021.

Yulong Xing and Chi-Wang Shu. High order well-balanced weno scheme for the gas dynamics equations under gravitational fields. *Journal of Scientific Computing*, 54(2): 645–662, 2013.

HC Yee and Björn Sjögreen. Nonlinear filtering in compact high-order schemes. *Journal of plasma physics*, 72(6):833–836, 2006.

Helen C Yee and Björn Sjögreen. Development of low dissipative high order filter schemes for multiscale navier–stokes/mhd systems. *Journal of Computational Physics*, 225(1): 910–934, 2007.

Jian Yu and Chao Yan. An artificial diffusivity discontinuous galerkin scheme for discontinuous flows. *Computers & Fluids*, 75:56–71, 2013.

Vidhi Zala, Robert M Kirby, and Akil Narayan. Structure-preserving nonlinear filtering for continuous and discontinuous galerkin spectral/hp element methods. *SIAM Journal on Scientific Computing*, 43(6):A3713–A3732, 2021.

Olindo Zanotti, Michael Dumbser, Arturo Hidalgo, and Dinshaw Balsara. An ader-weno finite volume amr code for astrophysics. *arXiv preprint arXiv:1401.6448*, 2014.

Olindo Zanotti, Francesco Fambri, Michael Dumbser, and Arturo Hidalgo. Space–time adaptive ADER discontinuous galerkin finite element schemes with a posteriori sub-cell finite volume limiting. *Computers & Fluids*, 118:204–224, September 2015.

Xiangxiong Zhang and Chi-Wang Shu. On positivity-preserving high order discontinuous galerkin schemes for compressible euler equations on rectangular meshes. *Journal of Computational Physics*, 229(23):8918–8934, 2010.

Xiangxiong Zhang and Chi-Wang Shu. Maximum-principle-satisfying and positivity-preserving high-order schemes for conservation laws: survey and new developments. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 467(2134):2752–2776, 2011.

Xiangxiong Zhang and Chi-Wang Shu. Positivity-preserving high order finite difference weno schemes for compressible euler equations. *Journal of Computational Physics*, 231 (5):2245–2258, 2012.

Xu Zhang and Dan Stanescu. Large eddy simulations of round jet with spectral element method. *Computers & Fluids*, 39(2):251 – 259, 2010. ISSN 0045-7930. doi: http://dx.doi.org/10.1016/j.compfluid.2009.09.002. URL http://www.sciencedirect.com/science/article/pii/S0045793009001273.

Jian Zhao and Huazhong Tang. Runge–kutta discontinuous galerkin methods for the special relativistic magnetohydrodynamics. *Journal of Computational Physics*, 343: 33–72, 2017.

Jun Zhu and Jianxian Qiu. Hermite WENO schemes and their application as limiters for runge-kutta discontinuous galerkin method, III: Unstructured meshes. *Journal of Scientific Computing*, 39(2):293–321, January 2009.

Valentin Zingan, Jean-Luc Guermond, Jim Morel, and Bojan Popov. Implementation of the entropy viscosity method with the discontinuous galerkin method. *Computer Methods in Applied Mechanics and Engineering*, 253:479–490, January 2013.

B Zuckerman and NJ Evans. Models of massive molecular clouds. *The Astrophysical Journal*, 192:L149–L152, 1974.

List of Figures

1.1 Venn diagram with the three properties expected from any numerical scheme. Overlapping regions are associated to the three classes of schemes according to their characteristics in accuracy, robustness and speed. 5

2.1 Schematic of the Riemann problem (2.23) in 1D with left $\boldsymbol{u}^+$ and right states $\boldsymbol{u}^-$ touching at the interface with the jump $\boldsymbol{u}^- - \boldsymbol{u}^+$. 22

2.2 Riemann fan of the linearized version of the Cauchy problem (2.23) consisting of m characteristics spreading out into the domain at different speeds λ_i. For the nonlinear equations the characteristics generally show more complex patterns, which are difficult to visualize schematically. The characteristics can be curved, hence they can diverge away from or converge towards each other leading to the versatile flow dynamics discussed in the text. 23

3.1 Lateral view of our Galaxy, the Milky Way, taken by Gaia, a space observatory of the ESA. The two bright objects in the lower right of the image are the Large and Small Magellanic Clouds, two dwarf galaxies orbiting the Galaxy. Image source: Moitinho et al. (2018). 29

3.2 Ratio of stagnant density vs static density (3.13) plotted over turbulent Mach number $\mathcal{T}$ for different heat capacity ratios γ. A flow is considered compressible when the variation in density is above 5 % (gray dashed line). 36

3.3 Sketch of a shockwave in the co-moving reference frame moving at same speed as the shock front. On the right side is the untouched cool medium in pre-shock state $\boldsymbol{u}^-$ and on the left side is the heated medium in post-shock state $\boldsymbol{u}^+$. Due to cooling effects, the heated fluid quickly cools shortly after the shock has swept through as is indicated by the tilted state profile in the post-shock region. 47

3.4 Rankine-Hugoniot jump ratios subject to the Mach number for the sonic regime ($\beta \to \infty$, solid lines) and a magnetic pressure dominated regime ($\beta = 2/5$, dashed lines) for various physical quantities and $\gamma = 5/3$. For the latter regime the physically feasible domain begins at Mach numbers $\mathcal{M} > 2$ (demarked by the dotted, gray vertical line). 51

3.5 Free fall time t_{ff} of a molecular cloud at density ρ according to (3.58). We consider a range from 1 particle per ccm (ISM) to extremely dense molecular cores of over 10^9 particles per ccm. The free fall time ranges from tenths of millions of years (galactic timescales) down to extremely short period of a several thousand years. Structures with densities above 10^7 particles per ccm (dashed, vertical line) are considered not resolvable by our codes. 58

3.6 Jeans' length L_{J} of a molecular cloud at density ρ and constant temperature $T = 10^4$ K according to (3.59). We consider a range from 1 particle per ccm (ISM) to very dense molecular cores of over 10^9 particles per ccm. The Jeans' length is plotted in $\log_2$-scale and ranges from $2^{10}\,\text{pc} = 1024$ pc down to $2^{-4}\,\text{pc} = 0.0625$ pc. Structures with densities above 10^7 particles per ccm (dashed, vertical line) are considered not resolvable by our codes. 60

4.1 Initial density of the 2D MHD KHI simulation with a weak constant magnetic field $B_1 = 10^{-2}$ in x-direction. 81

4.2 Contour plot of the density of the weakly magnetized 2D MHD KHI setup run by EC FV and SSP-RK(4,3) close before the crash. The result with LS-RK(5,4) looks the same. In the lower half, the streamlines (white) of the velocity field are overlayed. 82

4.3 Contour plots the inverse plasma-beta β^{-1} of the weakly magnetized MHD KHI setup run by the entropy conservative FV and SSP-RK(4,3) close before the crash. The result with LS-RK(5,4) looks the same. The streamlines (white) of the magnetic field are overlayed. 83

4.4 Contour plots the Mach number of the weakly magnetized MHD-KHI setup run by the EC FV and SSP-RK(4,3) close before the crash. The upper half depicts the sonic Mach number $\mathcal{M}$ according to (3.11) and the lower half depicts the turbulent Mach number $\mathcal{T}_q$ per block Ω_q according (3.12). The turbulent Mach number $\mathcal{T}_q$ is amplified by a factor of 3 in order to match the color range of the upper half. The bright red blocks in the lower half correspond to $\mathcal{T}_q \approx 0.3$ and the blocks in shades of cyan correspond to $\mathcal{T}_q \approx 0.15$. 84
4.5 Evolution of the global entropy drift (4.29) and mass drift of the weakly magnetized MHD-KHI setup run by the EC FV method with the third order SSP RK scheme and the fourth order Low-Storage RK scheme. Both simulations crash at $t \approx 3.7$ visible by a rapid surge of entropy drift caused by a breakdown of the numerics. Apparently, the SSP RK scheme has a noticeably linear mass drift losing mass at a rate of $\Delta\text{mass}/\Delta t \approx -1.3\times10^{-10}$. 85
4.6 Cell entropy production rates (4.36) of the weakly magnetized KHI setup (4.28) at $t = 2.5$ computed with the EC FV scheme and timestepping method SSP-RK(4,3). 87
4.7 Evolution of the maximum and minimum cell entropy production rate (4.36) in the whole computational domain computed with the EC FV scheme and timestepping method SSP-RK(4,3). The simulation crashes (negative densities and pressures) after 3180 timesteps respectively at simulation time $t = 3.727675$. 88
4.8 Contour plots of the density (left) and of cell entropy production rate (4.36) (right) of the weakly magnetized MHD KHI setup (4.28) run by the Minmod FV. 91
4.9 Evolution of the measured minimum and maximum entropy production rates at each timestep and in the whole domain of weakly magnetized MHD KHI setup (4.28) run by the Minmod FV till final simulation time $T = 10$. 92
4.10 Evolution of the entropy production rates for the weakly magnetized MHD-KHI setup (4.28) and with the Standard DG schemes. 106
4.11 Evolution of the entropy production rates for the weakly magnetized MHD-KHI setup (4.28) and with the Entropy Corrected DG schemes. 110

4.12 Evolution of the entropy production rates for the weakly magnetized MHD-KHI setup (4.28) and with the Flux Differencing DG schemes. 114
4.13 1D schematic of four ($N = 4$) mean values $\overline{\boldsymbol{u}}_i$ and their reconstructed nodal values (polynomial coefficients) $\widetilde{\boldsymbol{u}}_i$ constituting a polynomial of degree 3 spanning over the whole DG element. 115
4.14 1D schematic of two neighboring elements q and $q + 1$ each with four ($N = 4$) mean values of the indicator variable $\overline{\kappa}_i$ and their reconstructed polynomial coefficients $\widetilde{\kappa}_i$. The relative difference of the jumps at an element interface is considered to be a measure of the smoothness of the solution at hand. 120
4.15 Evolution of the entropy production rates for the weakly magnetized MHD-KHI setup (4.28) and with the convex blended Gauss DG schemes. 123
4.16 Evolution of the minimum entropy production rates for different DG variants. A noisy course with lots of high spikes indicates a troubled numerical scheme as is visible by the blue curve. The scheme eventually crashes at $t \approx 5.3$. 124
4.17 Density contours of four Gauss DG variants at time t = 7.5. Blend (Standard) Gauss DG and Blend Gauss DG with entropy correction (top row), Blend Gauss DG with entropy correction and EBP and the Flux Differencing Gauss DG with EBP (bottom row). The solution in the bottom left plot is a bit more dissipative than the rest. An explanation for this observation is given in the text. 126
4.18 Blending factors of two DG schemes at later time t = 7.5. Blend Gauss DG with entropy correction (left) and Blend Gauss DG with entropy correction and EBP (right). Clearly, EBP increases the blending activity causing more smearing of the solution. 127
4.19 Example of the density profile by the standard projection and the entropy projection. The vertical, gray, dashed lines depict the four Gauss quadrature points ($N = 4$) within the reference element. 127

5.1 Morton ordering, also known as Z-curve, of a 2D mesh with elements on different refinement levels. 136

5.2 Example of a small cluster with four compute nodes each equipped with four cores. In computing, it is customary to count instances starting with 0. On each node one MPI process respective NEMO instance (ne_i) manages four OpenMP threads (th_i) each alloted to one core. Threads within a common thread pool can exchange data via shared memory, while data exchange between nodes is managed by the NEMO instances via the MPI message bus (dashed rectangle in the center). 138
5.3 2D simulation with NEMO of two supersonic jets of gas (left plot) moving in opposite directions and getting diverted by a fictional black hole at the center. The right plot visualizes the mesh with different resolution levels managed by the quadtree library P4EST with six color-encoded parallel zones. In MPI jargon, a rank is the number or identifier (id) of a MPI process. The dark blue grid lines highlight the element boundaries. Not explicitly shown are the ghost layers clinging along the fault lines of adjacent zones. 139
5.4 Sketch of the processor - memory performance gap over the years since 1980. Dashed lines are rough extrapolation of performance gains for upcoming next generation hardware. 143
5.5 Speedup of NEMO on a single compute node on CHEOPS equipped with two CPUs á 12 cores. The parallelization method is pure OpenMP. Shown are the results (solid lines) for Standard DG and Flux Diff. DG (Gauss nodes) in 2D and 3D. The fits (dashed lines) with Ahmdal's law (5.4) were calculated only with data points from the first CPU. The course of the fits from 12 to 24 cores should be interpreted as extrapolations. The deviation of the measured results from the model in the second half of the plot are discussed in the text. 145
5.6 Speedup of NEMO on multiple compute nodes on the clusters CHEOPS and ODIN. Shown are the results (solid lines) with Standard DG (Gauss nodes) in 3D on a uniform grid of 256^3 DOF. The dashed lines are fits with Ahmdal's law (5.4). The fits were calculated from data points on CHEOPS for compute nodes 1 to 4 and on ODIN for nodes 1 to 10. The course of the fits beyond the fitting ranges should be interpreted as extrapolations. The dip in scaling on ODIN for nodes 11, 13 and 14 is discussed in the text. . . 146

6.1 Cutout of a PARAMESH grid with one block of 8^3 cells (shaded in yellow) and a neighboring block (shaded in white) on top at a finer resolution level. One 4^{th} order DG element is highlighted in blue and the blue dots indicate the position of the Gauss quadrature nodes (4.50) at element interfaces. The red square with its face-centered red boundary points exemplifies one FV cell respectively mean value within the block. 151

6.2 Cross section of a PARAMESH grid with blocks of 8^3 cells at different refinement levels (i.e. mesh resolution). The block of interest is shaded in yellow and its guard cell layer is shaded in gray. One 4^{th} order DG element with its adjacent elements is highlighted in blue and the blue dots indicate the position of the Gauss quadrature nodes (4.50). The red square with its red midpoint exemplifies one FV cell in the block. 152

6.3 Shown is the z-component of the current density $\vec{j} = \nabla \times \vec{B}$ at final time $T = 1.0$ as calculated with the DGFV4 scheme. 156

6.4 Magnetic pressure (left) with overlayed magnetic field lines (black) and blending factors (right) with overlayed grid lines (white) for the MHD current sheet test at simulation time $t = 2.5$. 158

6.5 Shown is the z-component of the current density $\vec{j} = \nabla \times \vec{B}$ with overlayed magnetic field lines for the MHD current sheet test at final time $T = 10$. . 158

6.6 Evolution of the domain-integrated absolute magnetic field divergence $\int_\Omega |\nabla \cdot \vec{B}(t)| d\Omega$ for the MHD current sheet test. Three runs with different flavors of divergence cleaning are shown. The simulation without any divergence correction method ("none"; blue line) crashed at around $t = 2.5$. The run where only the Powell source terms are used (orange line) is stable but has a substantially larger divergence error than the full scheme with Powell source terms and hyperbolic divergence cleaning (green line) which is the default setting in our code. 159

6.7 The percentage of total and individual energies as a function of time for the MHD current sheet test. We compare the results with (solid lines) and without (dashed lines) our hyperbolic divergence cleaning method. The results for the run without any divergence treatment are not shown, since it crashed early on. 160

6.8 Density, pressure, magnetic field, and blending profiles of the Briu-Wu shock tube at final time $T = 0.1$ run by the DGFV4 scheme on an AMR grid with a minimum resolution of 32 cells and a maximum resolution of 512 cells. We compute the reference solution with the ATHENA code using a resolution of 2048 cells. 162
6.9 MHD Shu-Osher shock tube test problem: The density, pressure, magnetic field, and blending profiles are shown at final time $T = 0.7$. The reference solution has been computed using ATHENA on 2048 grid cells. 164
6.10 Shown are the profiles of the numerical solution of the two-component Sod shock tube problem computed with the DGFV4 scheme (on an AMR grid with a minimum resolution of 32 cells and a maximum resolution of 256 cells) together with the exact solution at final time $T = 0.2$. Apparently, the blending is triggered at the pressure jump as designed. The contact discontinuity (density jump between $x = 0.6$ and $x = 0.8$) is sufficiently resolved by DG without any need in stabilization via blending. 166
6.11 Density (left) and pressure (right) contours of the 2D Sedov Blast setup computed with the DGFV4 scheme at final simulation time $T = 0.05$. . . . 168
6.12 Zoom-in of Figure 6.12 together with blended element entropy production rates (top right) given by (4.108) and blending factor (bottom right). The grid lines are overlayed in black (left) and white (right). 169
6.13 The profiles of shell-average profiles of normalized density (ρ/ρ_0), normalized pressure (p/p_0), scaled entropy production rate ($\dot{\Delta\mathcal{S}}/100$), and blending factor ($\alpha$) of the 2D Sedov blast setup computed with the DGFV4. The exact solution for density and pressure is shown as dashed lines. 170
6.14 Density profile of the Orszag-Tang vortex at the final time $T = 0.5$ as calculated with our DGFV4 scheme on an AMR mesh (shown as black lines in the left half of the plot) with a maximum refinement level of $l = 8$, which is equivalent to a maximum resolution of 1024^2 cells. 171
6.15 Numerical solution of the pressure profiles at $y = 0.3125$ and time $T = 0.5$ are shown for the DGFV4 solver with AMR and a maximum resolution of 1024^2 DOF. The reference solution was computed with the Athena code at a uniform resolution of 2048^2 DOF. 172

6.16 Domain-integrated divergence error $\int_\Omega |\nabla \cdot \vec{B}| d\Omega$ as a function of the simulation time for the Orszag-Tang vortex setup. Three runs with the DGFV4 are shown, where the Powell scheme and hyperbolic divergence cleaning are either on or off. The simulation without any divergence correction method activated is labeled as "none". 173
6.17 Turbulent Mach number (left) given by (3.12) and the blending factor (right) of the Orszag-Tang vortex setup computed with DGFV4 at final simulation time $t = 0.5$. The white lines in the right plot denote the grid lines. 174
6.18 Speedup of Bouchut5 and DGFV4 in FLASH on multiple compute nodes on ODIN. The 3D Orszag-Tang Vortex was computed till $T = 0.05$ with a fixed problem size of 256^3 DOF. The fits (dashed lines) are determined with Ahmdal's law (5.4). The volatility of the runtime measurements on ODIN is discussed in Section 5.4. 176
6.19 Density contours in logarithmic scale of the magnetic rotor at final time $T = 0.15$ calculated with DGFV4 on an dynamic AMR mesh (shown as black lines in the left half of the plane) with a maximum refinement level of $l = 8$ which is equivalent to a maximum resolution of 1024^2 cells. 177
6.20 Contours of the local divergence error $|\nabla \cdot \vec{B}|$ in logarithmic scale of the magnetic rotor run by the DGFV4 scheme at final time $T = 0.15$. 178
6.21 Turbulent Mach number (left) given by (3.12) and the blending factor (right) of the magnetic rotor setup computed with DGFV4 at final simulation time $T = 0.15$. The white lines in the right plot denote the grid lines. 179
6.22 Slice of the magnetic pressure at $z = 0$ for the MHD blast wave test solved with the DGFV4 scheme. The black lines on the left side highlight part of the AMR mesh and the gray stream lines indicate the field spanned by the x and y component of the magnetic field. 180
6.23 Turbulent Mach number (left) given by (3.12) and the blending factor (right) of the 3D MHD blast setup computed with DGFV4 at final simulation time $T = 0.01$. The white lines denote the grid lines. 181
6.24 Domain-integrated energies over simulation time for the Evrard test run by our DGFV4 solver. The dashed lines show the reference solution computed with PPM on a uniform grid. 183

6.25 Density slice of the Starbench test setup at simulation time $t = 0.8$ Myr computed with our DGFV4 solver and a maximum resolution of 256^3 cells. The dashed circle highlights the Starbench solution given by (6.10). The black lines depict the AMR grid. 186
6.26 Ionization fronts over time computed with the DGFV4 solver. For reference the Spitzer solution (6.8) and the Starbench solution (6.10) are given. . . . 187

7.1 Computational domain (cubic box) covering one octant of the supernova model. The faces at the coordinate axes are set to reflecting walls while the opposite sides are set to outflow. 191
7.2 Slices of density (top) and mass scalars (bottom) at z = 1 pc and final simulation time $T = 500$ yr of the young supernova remnant simulation carried out by the DGFV4 scheme in FLASH. The noise in the density near the plot axes are caused by the carbuncle phenomenon, a well-known issue for many numerical schemes at grid-aligned shocks. 195
7.3 Slices of blending factor (top) and turbulent Mach number (bottom) at z = 1 pc and final simulation time $T = 500$ yr of the young supernova remnant simulation carried out by the DGFV4 scheme in FLASH. The visible numerical artifacts in the inner region are discussed in the text. . . 196
7.4 Comparison of a false-color photo (left) of SN 1572 and a density schlieren (column sum) plot (right) computed from our simulation data at time $t = 450$ yr. The photo shown here is only segment of a much larger photo. Image source: Eriksen (2011). 197
7.5 MHD torus simulation at the initial time (top row) and after 8 periods carried out by our DGFV4 solver in FLASH. Shown are slices of the density (middle row) and magnetic field (bottom row) in x-y and x-z plane. Image source: Markert et al. (2022). 200
7.6 Slices of turbulent Mach number and blending factor in the x-y plane for the MHD torus simulation after 8 periods carried out by our DGFV4 solver in FLASH. 201
7.7 Slices of velocity field in x-y and x-z planes for the MHD torus simulation after 8 periods carried out by our DGFV4 solver in FLASH. 201

7.8 Observed turbulent D-type expansion within the Rosette nebula (left, Image source: Corban) and simulation within a fractal medium run by our DGFV4 solver. The false-color image on the left shows only a section of the much larger nebula with a radius of 20 pc taken by the Hubble Space Telescope (ESA/ESO/NASA). The right side shows the column density along the z-axis: $\int_{z_{\min}}^{z_{\max}} \rho(x, y, z)\,\mathrm{d}z$. Image source: Markert et al. (2022). . 204
7.9 Turbulent D-type expansion within a fractal medium simulated by our DGFV4 solver in FLASH. Shown are slices of density, temperature, turbulent Mach number and blending factor along the x-y plane at $t = 0.7$ Myr. 205
7.10 Molecular cloud within a jet of hot galactic outflow run by the DGFV4 solver. Shown is the column density along the z-axis for the total mass and different species. 208
7.11 Molecular cloud within a jet of hot galactic outflow run by the DGFV4 solver. Shown are slices along the x-y plane of the turbulent Mach number and the blending factor at $t = 20$ Myr. 209
7.12 False-color image of Smith's Cloud as an example of a high-velocity cloud hitting gas in our Galaxy's outskirts, which gives it the shape resembling a comet with a tail. The image shown here was rotated and cut to size in order to fit the page formatting. Image source: Saxton (2008). 209

A.1 Evolution of the entropy production rates of MHD-KHI setup and various DG variants stabilized via blending according to approximated entropy errors in each DG element. 218
A.2 Slice of the magnetic pressure at constant $z = 0$ solved with the Bouchut5 scheme. The black lines on the left side highlight part of the AMR mesh and the gray stream lines indicate the field spanned by the x- and y-component of the magnetic field. The maximum resolution level is set to 256^3 DOF. . 219
A.3 *left column:* 2D density slice (see figure 7.1) showing the instability region respectively remnant of the supernova at $T = 500\,\mathrm{yr}$ simulated with the (from top to bottom) DGFV2, DGFV4 and DGFV8 scheme. *right column:* 2D slice of the weighted blending factors of the respective blending schemes at $T = 500\,\mathrm{yr}$. The black lines correspond to the element boundaries of the Cartesian non-conforming mesh. Image source: Markert et al. (2021). . . . 220

A.4 MHD torus after 8 periods run for max. resolutions (from left to right) 64^3, 128^3, 256^3 and two fluid solvers Bouchut5 (rows 1,3) and DGFV4 (rows 2,4). Shown are log-scale density slices in the x-y plane (rows 1,2) and in the x-z plane (rows 3,4). Image source: Markert et al. (2022). . . . 221
A.5 MHD torus after 8 periods run for max. resolutions (from left to right) 64^3, 128^3, 256^3 and two fluid solvers Bouchut5 (rows 1,3) and DGFV4 (rows 2,4). Shown are log-scale magnetic pressure slices in the x-y plane (rows 1,2) and in the x-z plane (rows 3,4). The streamlines in white denote the magnetic field lines. Image source: Markert et al. (2022). 222
A.6 Turbulent D-type expansion into a fractal medium run by Bouchut5 solver at final simulation time $T = 1$ Myr. Shown is the column density along the z-axis. 223
A.7 Stellar feedback simulation run by the DGFV4 solver. Shown is the density slice along the x-y plane with annotations of the various flow features observable in the plot. 224
A.8 Stellar feedback simulation run by the Bouchut5 solver. Shown is the density slice along the x-y plane with annotations of the various flow features observable in the plot. 224
A.9 Molecular cloud within a jet of hot galactic outflow run by the Bouchut5 solver. Shown is the column density along the z-axis for the total mass and different species. 225

List of Tables

4.1 EOC of total pressure P of the MHD vortex problem (4.27) run by the second order EC FV scheme. 80
4.2 EOC of total pressure P of the MHD vortex problem (4.27) run by the second order Minmod FV scheme (4.40). 90
4.3 EOC of total pressure P of the MHD vortex problem (4.27) run by the fourth order Standard DG with Gauss quadrature. 102
4.4 EOC of total pressure P of the MHD vortex problem (4.27) run by the fourth order Standard DG with Lobatto quadrature. The simulations crash for resolutions 16^2 and 32^2 DOF. 103
4.5 EOC of total pressure P of the MHD vortex problem (4.27) run by the fourth order Standard DG with Gauss quadrature and EBP. 105
4.6 EOC of total pressure P of the MHD vortex problem (4.27) run by the fourth order Entropy Corrected Gauss DG without EBP. 109
4.7 EOC of total pressure P of the MHD vortex problem (4.27) run by the fourth order Entropy Corrected Gauss DG with EBP. 109
4.8 EOC of total pressure P of the MHD vortex problem (4.27) run by the fourth order Entropy Corrected Lobatto DG. 109
4.9 EOC of total pressure P of the MHD vortex problem (4.27) run by the fourth order Flux Diff. Gauss DG without EBP. 112
4.10 EOC of total pressure P of the MHD vortex problem (4.27) run by the fourth order Flux Diff. Gauss DG with EBP. 113
4.11 EOC of total pressure P of the MHD vortex problem (4.27) run by the fourth order Flux Diff. Lobatto DG. 113

5.1 Runtimes of the six DG variants, detailed in Section 4.6, for the 2D MHD-KHI setup till $T = 2.5$ and on a uniform grid of 256^2 DOF run by NEMO on a single core (#core = 1) of the workstation. All simulations had a total number of iteration steps of #steps = 1263×5 (LS-RK(5,4)). 142

6.1 EOC of total pressure P of the smooth Alfvén wave problem in 2D run by the fourth order DGFV4 scheme. 155
6.2 EOC of total pressure P of the manufactured solution problem in 3D run by the fourth order DGFV4 scheme. 155
6.3 Runtimes of Bouchut5 and DGFV4 in FLASH for the 3D Orszag-Tang Vortex setup run with a single core on the workstation. The total amount of DOF is $64^3 \approx 2.6 \times 10^5$. 175
6.4 Runtimes of Bouchut5 and DGFV4 in FLASH for the 3D Orszag-Tang Vortex setup run with 4×24 cores on CHEOPS. The total amount of DOF is $128^3 \approx 2.1 \times 10^6$. 175
6.5 Runtimes of Bouchut5 and DGFV4 in FLASH for the 3D MHD blast setup run with 4×24 cores on CHEOPS. The computations are conducted with AMR and the effective $\mathrm{DOF}_{\mathrm{eff.}}$ is $256^3 \approx 16.8 \times 10^6$. Details can be found in the text. 181

7.1 Conversion from cgs units to simulation units. 190
7.2 Hydrodynamical parameters in cgs units and simulation units. 191
7.3 Runtimes of the MHD torus setup for the Bouchut5 and our DGFV4 solver implemented in FLASH. All simulations ran on the cluster ODIN on 16 nodes á 16 cores. The definition for the computational throughput (TP) is given by eqn. (5.3). 202

Curriculum Vitae

Personalien:

Name	Johannes Markert (geb. König)
Geburtstag	19.04.1990
Geburtsort	Stollberg/Sachsen
Nationalität	Deutsch

Bildungsweg:

2017 - 2022	Promotionsstudium in angewandter Mathematik und wissenschaftlichem Rechnen *Universität zu Köln*
2013 - 2017	Masterstudium in Physik *Universität zu Köln*
2009 - 2013	Bachelorstudium in Physik *Technische Universität Chemnitz*
2001 - 2009	Hochschulreife auf dem ersten Bildungsweg *Carl-von-Bach-Gymnasium Stollberg/Sachsen*
1997 - 2001	Grundschulbildung *Albrecht-Dürer-Schule Stollberg/Sachsen*

Erklärung

Hiermit versichere ich an Eides statt, dass ich die vorliegende Dissertation selbstständig und ohne die Benutzung anderer als der angegebenen Hilfsmittel und Literatur angefertigt habe. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten und nicht veröffentlichten Werken dem Wortlaut oder dem Sinn nach entnommen wurden, sind als solche kenntlich gemacht. Ich versichere an Eides statt, dass diese Dissertation noch keiner anderen Fakultät oder Universität zur Prüfung vorgelegen hat; dass sie - abgesehen von unten angegebenen Teilpublikationen und eingebundenen Artikeln und Manuskripten - noch nicht veröffentlicht worden ist sowie, dass ich eine Veröffentlichung der Dissertation vor Abschluss der Promotion nicht ohne Genehmigung des Promotionsausschusses vornehmen werde. Die Bestimmungen dieser Ordnung sind mir bekannt. Darüber hinaus erkläre ich hiermit, dass ich die Ordnung zur Sicherung guter wissenschaftlicher Praxis und zum Umgang mit wissenschaftlichem Fehlverhalten der Universität zu Köln gelesen und sie bei der Durchführung der Dissertation zugrundeliegenden Arbeiten und der schriftlich verfassten Dissertation beachtet habe und verpflichte mich hiermit, die dort genannten Vorgaben bei allen wissenschaftlichen Tätigkeiten zu beachten und umzusetzen. Ich versichere, dass die eingereichte elektronische Fassung der eingereichten Druckfassung vollständig entspricht.

Teilpublikationen:

Johannes Markert, Gregor Gassner, and Stefanie Walch. (2020).
A Sub-Element Adaptive Shock Capturing Approach for Discontinuous Galerkin Methods. *Communications on Applied Mathematics and Computation*

Johannes Markert, Stefanie Walch, and Gregor Gassner. (2021).
A Discontinuous Galerkin Solver in the FLASH Multi-Physics Framework. *Monthly Notices of the Royal Astronomical Society, vol. 511, issue 3, pp. 4179-4200*

Köln, den 20.03.2022

Digital signiert von Johannes Markert
Datum: 2022.05.31 11:29:08+02'00'