Friedrich-Schiller-Universität Jena
Physikalisch-Astronomische Fakultät
Theoretisch-Physikalisches Institut

Masterarbeit

A New ADER-DG Scheme based on a Local Continuous Runge-Kutta Method

von

Matthias Pilz
geboren am 24.03.1990 in Frankfurt (Oder)

2015

Submitted on: October 30, 2015

Thesis supervisors: Prof. Dr. Bernd Brügmann
Dr. David Hilditch

Abbreviations and Notation

Throughout the thesis we apply Einsteins summation convention, implying summation over all indices appearing twice, except for equations which make use of the Gaussian quadrature formula.

There are a few variables that are used in the same way consistently, namely

M	polynomial degree of data representation
N	number of cells on specified domain
C	CFL number
$\mathbf{U}$	vector of conserved quantities
$\mathbf{F}$	vector of fluxes
$\mathbf{S}$	vector of sources
λ_l	Gauss-Legendre nodes in the unit interval
$\varphi_l(\xi)$	Lagrange interpolating polynomials
$\mathbf{u}_h$	polynomial approximation of state vector
$\mathbf{v}_h$	alternative constant cell average approximation of state vector
$\mathbf{q}_h$	space-time predictor function
$\mathbf{w}_h$	polynomial reconstruction of cell average data

A list of frequently used abbreviations is given below.

ADER	Arbitrary Derivative Riemann problem
CFL	Courant-Friedrichs-Lewy
DG	Discontinuous Galerkin
ERK	Explicit Runge-Kutta method
FV	Finite Volume
HCL	Hyperbolic Conservation Law
IVP	Initial Value Problem
MOOD	Multi-Dimensional Optimal Order Detection
NAD	Numerical Admissibility Detection
ODE	Ordinary Differential Equation
PAD	Physical Admissibility Detection
PDE	Partial Differential Equation
RK	Runge-Kutta
TOV	Tolman-Oppenheimer-Volkoff
WENO	Weighted Essentially Non-Oscillatory

Abstract

We study a discontinuous Galerkin (DG) finite element method for systems of nonlinear hyperbolic conservation laws applying an a posteriori finite volume subcell limiter technique. A key part of the algorithm is the calculation of a space-time predictor solution, which enables us to obtain a high order approximation of the numerical fluxes in the one-step DG scheme. A novel alternative to the existing iterative procedure using an element-local continuous extension of Runge-Kutta methods is proposed. Test cases with linear and nonlinear equations display a reduced run time, while achieving the same accuracy and order of convergence.

Kurzdarstellung

Wir untersuchen ein diskontinuierliches finite Elemente Galerkin Verfahren für Systeme von nichtlinearen hyperbolischen Erhaltungsgleichungen, welches einen a posteriori finite Volumen Limiter beinhaltet. Ein wichtiger Bestandteil des Algorithmus ist die Berechnung einer Raum-Zeit Prädiktor Lösung, die eine Approximation des numerischen Flusses in hoher Ordnung ermöglicht. Eine neuartige Alternative zu dem bereits existierenden iterativen Verfahren wird vorgeschlagen. Diese basiert auf einer elementweisen stetigen Erweiterung von Runge-Kutta Methoden. Tests mit linearen und nichtlinearen Gleichungen zeigen eine Verringerung der Laufzeit bei gleichbleibender Genauigkeit und Konvergenzordnung.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Systems of hyperbolic conservation laws (HCL) describe a wide range of physical setups, e.g. the wave equation, Eulers equation of compressible fluids and the equations of general relativistic hydrodynamics. This makes them very interesting to study and a lot of time has already been spent on developing numerical methods that are able to preserve high accuracy in smooth regions and can deal with discontinuities and shocks, which might arise due to the nonlinearities of the equations. In 1973, Reed and Hill were the first to introduce a discontinuous Galerkin (DG) finite element method [Reed and Hill, 1973]. The name results from the piecewise polynomial data representation, allowing for discontinuities at the cell boundaries. In the nineties, Cockburn and Shu established an extensive theoretical framework concerning the application of the DG method on HCL. Nowadays, DG methods manifest themselves as particularly successful (see e.g. [Bugner et al., 2015; Zanotti et al., 2015a]), because of their flexibility in handling complicated geometries, their natural subcell resolution property and stability [Dumbser et al., 2014]. Furthermore, [Zanotti et al., 2015b] showed how well space-time adaptive mesh refinement is compatible in the DG-framework.

To avoid the Gibbs phenomenon nonlinear limiting is needed at shock waves or other discontinuities. Many limiters have been investigated in the past, sharing the basic idea of: first, computing a solution with the DG scheme, then identifying those cells that seem to be troubled, i.e. need limiting. After that the unlimited solution is modified with some sort of nonlinear reconstruction technique, e.g. weighted essentially non-oscillatory (WENO) reconstruction, based on the troubled cell and its neighbours.

The algorithm we want to investigate in this thesis proposes a different approach. Instead of trying to correct the unlimited solution, we recompute the solution using a different numerical scheme, namely an ADER-WENO finite volume method [Dumb-

ser et al., 2013]. ADER is short for arbitrary derivative Riemann problem which hints towards the special treatment of the flux integrals. In order to not destroy the subcell resolution of the DG method, the recomputation is done on a finer subgrid inside each cell. For an appropriate number of subcells per topcell we can define suitable projection and reconstruction operators to transfer data between the grids without loss of information [Dumbser et al., 2014]. In fact, the choice of the subgrid size is also of great importance with respect to the CFL number C. Preferably, we would like to take C as large as possible and ensure that the respective time step size is identical to the one of the DG scheme on the topgrid. Applying polynomial approximation of degree M in the DG scheme we will end up with $2M + 1$ subcells per topcell.

In this thesis we want to characterise the efficiency of the proposed ADER-DG method. As mentioned ealier, a key part of the ADER algorithms is the evaluation of the numerical flux function in both the DG method, as well as in the finite volume method. It serves as an approximate solution of the generalised Riemann problem at the interfaces of the cells and allows to construct schemes with arbitrary high-order accuracy. It turns out that the local space-time predictor needed to evaluate the numerical flux terms makes up a significant amount of the overall computational cost. This leads to the development of a new alternative predictor method based on the continuous extension of Runge-Kutta (RK) schemes. The dense output formalism produces full polynomial output in temporal direction using only the already calculated right hand sides. As a result the novel method is more efficient, in the sense of needing less flux function evaluations while preserving the same accuracy and order of convergence, for all orders tested between three and six.

The rest of this thesis is organised as follows. There are three major parts. In the first part called "Mathematical Framework" we review some basics of hyperbolic conservation laws (Chapter 2), introduce the key building blocks of the algorithm (Chapter 3) and eventually combine these to give a detailed description of the full method (Chapter 4). Here we also discuss details of the actual implementation. The second part concentrates solely on the two different predictor methods. Firstly, we explain the one used in [Dumbser et al., 2013; Dumbser et al., 2014] and prove the convergence of this iterative approach (Chapter 5). Secondly, we derive an alternative local RK predictor method (Chapter 6). In the final part we carry out several numerical tests to show the correctness and efficiency of our code (Chapter 7). Chapter 8 summarises the thesis and points out perspectives.

Part I

Mathematical Framework

Chapter 2

Basics of Hyperbolic Conservation Laws

In this chapter we define the term hyperbolic conservation law and introduce the method of characteristics which allows us to obtain an analytic solution for systems with constant coefficients. The advection equation and the Burgers equation serve as examples of this procedure. For the most parts we follow the notation of the book [Toro, 2009], which gives a very comprehensive overview on this topic.

2.1 Definitions

So that we can define a system of hyperbolic conservation laws, we need to introduce two basic definitions (see [Toro, 2009]):

Definition 2.1 (Conservation Laws). *Conservation laws are systems of partial differential equations (PDEs) that can be written in the form*

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}(\mathbf{U})}{\partial x} = \mathbf{0} \ , \tag{2.1}$$

where

$$\mathbf{U} = \begin{pmatrix} U_1 \\ U_2 \\ \vdots \\ U_\nu \end{pmatrix} \ , \qquad \textit{and} \qquad \mathbf{F}(\mathbf{U}) = \begin{pmatrix} F_1 \\ F_2 \\ \vdots \\ F_\nu \end{pmatrix} \ . \tag{2.2}$$

We call $\mathbf{U}$ *the vector of conserved quantities and* $\mathbf{F}(\mathbf{U})$ *the vector of fluxes of length* ν *where each of the components* F_i *is a function of the components* U_j *of* $\mathbf{U}$.

Definition 2.2 (Jacobian Matrix). *The Jacobian of the flux function* $\mathbf{F}(\mathbf{U})$ *in* (2.1) *is the matrix*

$$\mathbf{A}(\mathbf{U}) = \frac{\partial \mathbf{F}(\mathbf{U})}{\partial \mathbf{U}} = \begin{pmatrix} \partial F_1/\partial U_1 & \dots & \partial F_\nu/\partial U_1 \\ \vdots & \ddots & \vdots \\ \partial F_1/\partial U_\nu & \dots & \partial F_\nu/\partial U_\nu \end{pmatrix} . \tag{2.3}$$

With the Jacobian we can rewrite the second term in (2.1) and obtain

$$\frac{\partial \mathbf{U}}{\partial t} + \mathbf{A}(\mathbf{U})\frac{\partial \mathbf{U}}{\partial x} = \mathbf{0} . \tag{2.4}$$

Depending on the algebraic properties of the Jacobian we can define a hyperbolic system as follows:

Definition 2.3 (Hyperbolic System). *A system* (2.4) *of* ν *equations is said to be hyperbolic at a point* (x,t) *if* $\mathbf{A}$ *has* ν *real eigenvalues* $\lambda_1, \dots, \lambda_\nu$ *and a corresponding set of* ν *linearly independent right eigenvectors* $\mathbf{K}^{(1)}, \dots, \mathbf{K}^{(\nu)}$*. The system is said to be strictly hyperbolic if the eigenvalues* λ_i *are all distinct.*

A powerful tool for solving PDEs is the method of characteristics, which we want to apply on the initial-value problem (IVP) for the special case of scalar hyperbolic conservation laws, namely

$$\frac{\partial U}{\partial t} + \lambda(U)\frac{\partial U}{\partial x} = 0 , \qquad U(x,0) = U_0(x) . \tag{2.5}$$

Definition 2.4 (Characteristic Curves). *Characteristic curves are curves* $x = x(t)$ *in the* $t-x$ *plane along which the PDE becomes an ordinary differential equation (ODE).*

In the following we look at characteristic curves $x = x(t)$ that satisfy the IVP

$$\frac{dx}{dt} = \lambda(U) , \qquad x(0) = x_0 . \tag{2.6}$$

Considering both U and x to be functions of t, that is $U = U(x(t), t)$, we can express the rate of change of U along $x(t)$ as

$$\begin{aligned} \frac{dU}{dt} &= \frac{\partial U}{\partial t} + \frac{dx}{dt}\frac{\partial U}{\partial x} \\ &= \frac{\partial U}{\partial t} + \lambda(U)\frac{\partial U}{\partial x} = 0 . \end{aligned} \tag{2.7}$$

From this we see that U is constant along the characteristic curves, which are straight lines with slope $\lambda(U)$. The value of U along each curve is determined by

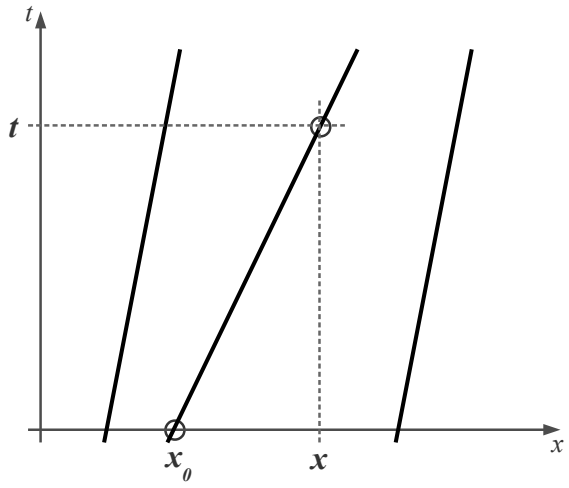


Figure 2.1: "Typical characteristic curves for a nonlinear hyperbolic conservation law." [Toro, 2009]

the initial condition and we write $U(x,t) = U_0(x_0)$. The solutions of the IVP (2.6) are

$$x = x_0 + \lambda(U_0(x_0))t \ . \tag{2.8}$$

which shows that x_0 in a sense depends on the point (x,t). This is also depicted in Figure 2.1. The exact and implicit solution of the IVP (2.5) is

$$U(x,t) = U_0(x - \lambda(U_0(x_0))t) \ . \tag{2.9}$$

2.2 Examples

To get a better understanding of (2.9) we look at two important examples of HCLs, namely the advection equation and the Burgers equation.

Example 2.5 (Advection Equation).

$$\frac{\partial U}{\partial t} + a\frac{\partial U}{\partial x} = 0 \ , \qquad U(x,0) = U_0(x) \ . \tag{2.10}$$

The characteristic speed a is constant over the whole domain, which means that all characteristic curves have the same slope and $x_0 = x_0(x,t)$ can be calculated explicitly. Putting this into (2.9) we obtain

$$U(x,t) = U_0(x - at) \ , \tag{2.11}$$

which describes a translation of the initial data with constant speed a.

Example 2.6 (Burgers Equation).

$$\frac{\partial U}{\partial t} + \frac{1}{2}\frac{\partial U^2}{\partial x} = 0 \ , \qquad U(x,0) = U_0(x) \ . \tag{2.12}$$

Rewriting (2.12) in the form of (2.4) reveals that the characteristic speed $\lambda = U$ for this problem depends directly on the solution itself. That means we can only find an implicit solution (see (2.9))

$$U(x,t) = U_0(x - Ut) \ . \tag{2.13}$$

In order to obtain explicit values, we will apply a rather simple but efficient Newton-Raphson method as described in section 9.4 of [Press et al., 1988]. This is possible as long as the solution did not develop any discontinuities, because the method is not able to deal with two-valued functions U. In this case we compute explicit values with the finite volume WENO scheme (see Section 3.3.1) using a very high resolution.

Chapter 3

Building Blocks of the Algorithm

Consider a more general version of the hyperbolic conservation law (2.1), where we introduce a non-zero source term $\mathbf{S} = \mathbf{S}(\mathbf{U}, x, t)$. The resulting equation is usually referred to as hyperbolic balance law:

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}(\mathbf{U})}{\partial x} = \mathbf{S} \ . \tag{3.1}$$

In order to solve this PDE we introduce a grid setup consisting of two grids in Section 3.1. On the topgrid we approximate the data by polynomials of degree M and perform a space-time ADER-DG method to step forward in time (see Section 3.2). If the solution is troubled, in a sense to be defined, we project it locally onto a finer subgrid, where data is stored in form of cell averages. The corresponding projection and reconstruction operators to switch between the two data representations are shown in Section 3.1. Updating the subcell information will be done by a finite volume WENO scheme which we derive in Section 3.3.

3.1 Grid Setup & Data Representation

Our spatial discretisation is based on a grid

$$a = x_{\frac{1}{2}} < x_{\frac{3}{2}} < \cdots < x_{N-\frac{1}{2}} < x_{N+\frac{1}{2}} = b \ , \tag{3.2}$$

which allows us to define N conforming cells I_i, cell centers x_i and cell sizes Δx_i by

$$\begin{aligned} I_i \equiv \left[x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}} \right] \ , \qquad & x_i \equiv \frac{1}{2} \left(x_{i-\frac{1}{2}} + x_{i+\frac{1}{2}} \right) \ , \\ \Delta x_i \equiv x_{i+\frac{1}{2}} - x_{i-\frac{1}{2}} \ , \qquad & i = 1, 2, \ldots, N \ . \end{aligned} \tag{3.3}$$

This holds for both the topgrid and the subgrid where N is equal to N_{top} and N_{sub}, respectively, with $N_{sub} = \eta \cdot N_{top}$. This means for every topcell T_i we have several

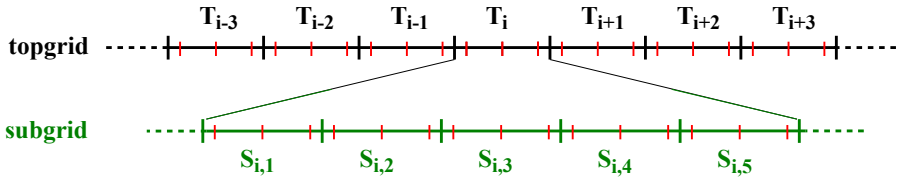


Figure 3.1: Exemplary representation of the topgrid-subgrid relation for $M = 2$. Each topcell (black) contains $M+1 = 3$ Gauss-Legendre nodes (red). For one topcell T_i we have $\eta = 2M + 1 = 5$ subcells $S_{i,j}$ (green), where each of them also contains three Gauss-Legendre nodes.

subcells denoted by $S_{i,j},\ j = 1, \ldots, \eta$. The choice of η will be discussed in Chapter 4. We use this double index-notation when we want to explicitly point out the respective topcell, otherwise we simply write S_i with $i = 1, \ldots, N_{sub}$. Throughout this thesis, equidistant cells are chosen for both grids. A graphical summary of these definitions is given in Figure 3.1.

We define the finite dimensional approximation space of piecewise polynomials up to degree M by

$$\mathbb{H}^M := \{\varphi : \varphi(x)|_{I_i} \in \mathbb{P}^M(I_i)\} \ , \tag{3.4}$$

with $\mathbb{P}^M(I_i)$ denoting the space of polynomials on I_i of degree at most M. Again we take I_i as the general expression representing both T_i and $S_{i,j}$. To be more explicit we consider a nodal basis of polynomials of degree M rescaled on the unit interval $I = [0, 1]$, namely $M + 1$ Lagrange interpolating polynomials

$$\varphi_l(\xi) \equiv \prod_{\substack{k=1 \\ k \neq l}}^{M+1} \frac{\xi - \lambda_k}{\lambda_l - \lambda_k}, \qquad l = 1, 2, \ldots, M+1 \ , \tag{3.5}$$

where the λ_k are the $M + 1$ associated Gauss-Legendre nodes in the unit interval I. They are closely related to the roots of the Legendre polynomial P_{M+1} of order $M + 1$. More details on these non-uniformly distributed collocation points will be discussed in Chapter 4. From the definition of the Lagrange polynomials we see that the standard property

$$\varphi_l(\lambda_k) = \delta_{lk} \ , \qquad k, l = 1, \ldots, M+1 \ , \tag{3.6}$$

where δ_{lk} is the Kronecker symbol, is valid.

In each cell of the topgrid the state vector $\mathbf{U}$ at time t^n is represented by $\mathbf{u}_h(x, t^n)$,

$$\mathbf{u}_h(x, t^n) = \sum_l \varphi_l(x) \hat{\mathbf{u}}_l^n \equiv \varphi_l(x) \hat{\mathbf{u}}_l^n \ . \tag{3.7}$$

Because of (3.6) the data is directly given at the Gaussian quadrature points by the discrete coefficients $\hat{\mathbf{u}}_l^n$, which will be very useful when calculating integrals over the

reference interval.

On the subcells we represent the data by simple cell averages which are referred to as $\mathbf{v}_h(x,t^n)$. This alternative data representation is defined by a set of piecewise constant subcell averages which can directly be computed from $\mathbf{u}_h(x,t^n)$

$$\mathbf{v}_{i,j}^n = \frac{1}{|S_{i,j}|}\int\limits_{S_{i,j}} \mathbf{u}_h(x,t^n)\ dx = \frac{1}{|S_{i,j}|}\int\limits_{S_{i,j}} \varphi_l(x)\ dx\ \hat{\mathbf{u}}_l^n \qquad \forall S_{i,j} \in T_i\ . \tag{3.8}$$

The additional $M+1$ Gauss-Legendre nodes in each of the subcells (see Figure 3.1) will become relevant when we discuss the numerical method applied on the subcell data. Equation (3.8) defines a projection operator $\mathcal{P}$ for which we will use the short notation $\mathbf{v}_h(x,t^n) = \mathcal{P}(\mathbf{u}_h(x,t^n))$ in the following. In order to guarantee that the topcell polynomial of degree M can be recovered identically from the subcell averages by means of a reconstruction operator $\mathbf{u}_h(x,t^n) = \mathcal{R}(\mathbf{v}_h(x,t^n))$ we set $\eta = 2M+1$ and require $\mathcal{RP}$ to give the identity operator (but not necessarily $\mathcal{PR}$). In other words we want to find a polynomial of degree M that satisfies the $2M+1$ equations (3.8). This system is overdetermined but can be solved by a constrained least-square approach. In fact, $\mathcal{R}$ turns out to be the pseudo-inverse of the matrix associated with the projection operator (3.8). Further details on the data representations and the operators are given in Section 4.2 of [Dumbser et al., 2014].

3.2 Topcell Methods

Firstly, we will introduce the numerical method applied on the topcell polynomial data in Section 3.2.1. As we will see, the method is unlimited in the sense that it will produce unphysical oscillations when it has to deal with steep gradients or discontinuities.

Secondly, these problematic cells need to be detected which motivates the introduction of a troubled cell indicator. Instead of using the full multi-dimensional optimal order detection (MOOD) approach proposed in [Dumbser et al., 2014] we make use of a modified version where we adopt a different numerical admissibility criterion which is similar to what is used in [Qiu and Shu, 2005]. It is based on the standard minmod function and will be discussed in Section 3.2.2.

3.2.1 ADER Discontinuous Galerkin Scheme

We multiply the governing PDE (3.1) by a test function $\varphi_k \in \mathbb{H}$ and integrate over the space-time control volume $T_i \times [t^n, t^{n+1}]$, hence

$$\int_{t^n}^{t^{n+1}} \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \varphi_k(x) \left(\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}(\mathbf{U})}{\partial x} - \mathbf{S} \right) \, dx \, dt = 0 \, . \tag{3.9}$$

In the next step we integrate the second term by parts and insert our choice of representation of the state vector $\mathbf{U}$ (see (3.7))

$$\begin{aligned} &\int_{t^n}^{t^{n+1}} \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \varphi_k \frac{\partial \mathbf{u}_h}{\partial t} dx \, dt + \int_{t^n}^{t^{n+1}} \left[\varphi_k \mathbf{F}(\mathbf{u}_h) \right]_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \, dt \\ &- \int_{t^n}^{t^{n+1}} \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \frac{\partial \varphi_k}{\partial x} \mathbf{F}(\mathbf{u}_h) \, dx \, dt - \int_{t^n}^{t^{n+1}} \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \varphi_k \mathbf{S}(\mathbf{u}_h) \, dx \, dt = 0 \, . \end{aligned} \tag{3.10}$$

We replace the physical flux $\mathbf{F}$ in the boundary term by a numerical flux function $\tilde{\mathbf{f}}$, which is a function of the left and right boundary-extrapolated data. To compute the temporal integrals for the flux and source terms we need to introduce the space-time function $\mathbf{q}_h$, which is an approximate solution to (3.1) neglecting the influence of neighbouring cells. As mentioned in [Montecinos et al., 2012] the numerical flux function can be interpreted as an solution to the generalised Riemann problem at the interface of the cells and allows the construction of an arbitrary high-oder accurate scheme in space and time. In Part II we will derive two different approaches how to obtain this predictor. We end up with an one-step discontinuous Galerkin scheme:

$$\begin{aligned} &\left(\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \varphi_k \varphi_l \, dx \right) (\hat{\mathbf{u}}_l^{n+1} - \hat{\mathbf{u}}_l^n) + \int_{t^n}^{t^{n+1}} \left[\varphi_k \tilde{\mathbf{f}} \left(\mathbf{q}_h^-(x,t), \mathbf{q}_h^+(x,t) \right) \right]_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \, dt \\ &- \int_{t^n}^{t^{n+1}} \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \frac{\partial \varphi_k}{\partial x} \mathbf{F}(\mathbf{q}_h) \, dx \, dt - \int_{t^n}^{t^{n+1}} \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \varphi_k \mathbf{S}(\mathbf{q}_h) \, dx \, dt = 0 \, . \end{aligned} \tag{3.11}$$

Note that we integrated the first term by part as well. The superscript assigned to the predictor $\mathbf{q}_h$ denotes from which side we approach the respective boundary

value (see Figure 3.2), hence

$$\begin{aligned} \mathbf{q}_h^-(x_{i-\frac{1}{2}}) &= \varphi_k(1)\ \hat{\mathbf{q}}_k^{i-1}\ , & \mathbf{q}_h^-(x_{i+\frac{1}{2}}) &= \varphi_k(1)\ \hat{\mathbf{q}}_k^{i}\ , \\ \mathbf{q}_h^+(x_{i-\frac{1}{2}}) &= \varphi_k(0)\ \hat{\mathbf{q}}_k^{i}\ \ , & \mathbf{q}_h^+(x_{i+\frac{1}{2}}) &= \varphi_k(0)\ \hat{\mathbf{q}}_k^{i+1}\ . \end{aligned} \tag{3.12}$$

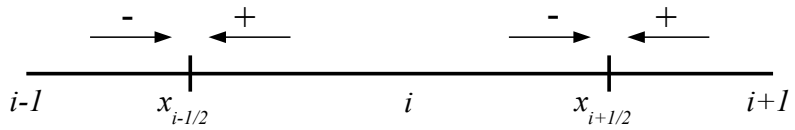


Figure 3.2: Graphical explanation of the superscripts $+/-$ in the numerical flux function in (3.11) at the cell interfaces.

Assuming we have given a polynomial representation $\mathbf{u}_h^n$ and a space-time predictor $\mathbf{q}_h$, (3.11) tells us how to compute the discrete coefficients $\hat{\mathbf{u}}_h^{n+1}$ of the solution on the next time slice. Details on the implementation are given in Chapter 4.

From (3.11) we see that the numerical flux function $\tilde{\mathbf{f}}$ takes two arguments, which represent their value at a certain interface boundary approached from either the positive or negative direction. We require it to be identical to the analytic flux function if both values are equal. This is called the consistency condition [Toro, 2009]:

$$\tilde{\mathbf{f}}\left(\mathbf{a},\mathbf{a}\right) = \mathbf{F}(\mathbf{a})\ . \tag{3.13}$$

Throughout the whole thesis the local Lax-Friedrichs flux, also called Rusanov flux,

$$\tilde{\mathbf{f}}\left(\mathbf{a},\mathbf{b}\right) = \frac{1}{2}\left(\mathbf{F}(\mathbf{a}) + \mathbf{F}(\mathbf{b})\right) - \frac{1}{2}\left|\lambda_{\max}\right|\left(\mathbf{b}-\mathbf{a}\right) \tag{3.14}$$

is used, where $|\lambda_{\max}|$ is the maximum absolute value of the eigenvalues of the Jacobian matrix (2.3) in the current cell. This choice obviously fulfils condition (3.13).

As noted in [Dumbser et al., 2014] this family of schemes is unlimited, meaning that it cannot prevent numerical oscillations from occuring in the presence of steep gradients. For smooth solutions the scheme has been shown empirically to converge with order $M+1$. [Dumbser et al., 2008a; Dumbser et al., 2014]

3.2.2 Troubled Cell Indication

In order to identify those cells that show numerical oscillations we adopt the troubled cell indicator proposed in [Qiu and Shu, 2005]. Additionally, we implement an idea of [Dumbser et al., 2014], who does not only check for numerical admissibility but also physically motivated criteria. The reason why we do not apply their full MOOD detection is twofold: (i) Our tests have shown that in certain setups small

local oscillations were not detected (see Section 7.2.1) and (ii) instead of projecting everything onto the subcells for the detection, we work directly on the topcell level and only need to project those cells that are problematic. We apply the following detection criteria on a candidate solution $\mathbf{u}_h^{*,n+1}$ obtained from (3.11).

Physical admissibility detection (PAD)

This criterion is based on the specific form of the HCL. One example is that there might be physical quantities which have to satisfy the positivity constraint, namely

$$\pi\left(\mathbf{u}_h^{*,n+1}(x)\right) > 0 \ , \qquad \forall x \in T_i \ . \tag{3.15}$$

This is the case for the Euler equations of compressible gas dynamics, where we assume the mass density ρ and the pressure p to be positive, hence $\pi_1(\mathbf{U}) = \rho$ and $\pi_2(\mathbf{U}) = p$. (see [Dumbser et al., 2014])

Numerical admissibility detection (NAD)

Given the polynomials $\mathbf{u}_h^{*,n+1}$ we calculate the average over each topcell T_i

$$\bar{\mathbf{u}}_i = \frac{1}{\Delta x_{top}} \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \mathbf{u}_h^{*,n+1} \, dx \ . \tag{3.16}$$

Furthermore, we denote the boundary values of $\mathbf{u}_h^{*,n+1}$ by

$$\mathbf{u}_{i-\frac{1}{2}}^{+} = \mathbf{u}_h^{*,n+1}(x_{i-\frac{1}{2}}) \ , \qquad \mathbf{u}_{i+\frac{1}{2}}^{-} = \mathbf{u}_h^{*,n+1}(x_{i+\frac{1}{2}}) \ , \tag{3.17}$$

using the usual convention for the superscripts similar to (3.12) and define the differences

$$^{l}\mathbf{u}_i := \bar{\mathbf{u}}_i - \mathbf{u}_{i-\frac{1}{2}}^{+} \ , \qquad ^{r}\mathbf{u}_i := \mathbf{u}_{i+\frac{1}{2}}^{-} - \bar{\mathbf{u}}_i \ , \tag{3.18}$$

$$\Delta_+ \bar{\mathbf{u}}_i := \bar{\mathbf{u}}_{i+1} - \bar{\mathbf{u}}_i \ , \qquad \Delta_- \bar{\mathbf{u}}_i := \bar{\mathbf{u}}_i - \bar{\mathbf{u}}_{i-1} \ . \tag{3.19}$$

For all of our tests we use the TVB modified minmod function with the TVB constant κ (see [Qiu and Shu, 2005])

$$\tilde{m}(a_1, a_2, \ldots, a_n) = \begin{cases} a_1 \ , & \text{if } |a_1| \leq \kappa \ (\Delta x_{top})^2 \\ m(a_1, \ldots, a_n) \ , & \text{otherwise} \end{cases} \ , \tag{3.20}$$

where m is the standard minmod function given by

$$m(a_1, \ldots, a_n) = \begin{cases} s \cdot \min_{1 \leq j \leq n} |a_j| \ , & \text{if } \ \operatorname{sign}(a_1) = \cdots = \operatorname{sign}(a_n) = s \\ 0 \ , & \text{otherwise} \end{cases} \ . \tag{3.21}$$

We calculate

$$\tilde{m}\left({}^{r}\mathbf{u}_i,\ \Delta_+\bar{\mathbf{u}}_i,\ \Delta_-\bar{\mathbf{u}}_i\right)\ , \qquad \tilde{m}\left({}^{l}\mathbf{u}_i,\ \Delta_+\bar{\mathbf{u}}_i,\ \Delta_-\bar{\mathbf{u}}_i\right)\ , \tag{3.22}$$

and identify those cells as troubled where the minmod function returns any other argument than the first one. Note that we perform this detection componentwise and declare a cell as troubled if at least one component shows problematic behaviour. Troubled cells are denoted by $\tilde{T}_i$. From (3.20) we see that choosing very large values for κ leads to a suppression of the indicator, while too small κ's will mark even good cells as troubled. The choice of the TVB constant depends highly on the solution of the problem.

If either one of the criteria, PAD or NAD, is not fulfilled, the solution of this particular cell and both adjacent ones is recomputed. Taking these additional cells into account makes the program slightly slower but provides greater security.

3.3 Subcell Methods

The solution in the troubled cells $\tilde{T}_i$ has to be recomputed. This is done on the respective subcells applying a finite-volume scheme, summarised in the following subsection.

We will also introduce the WENO reconstruction method from [Dumbser et al., 2013], which is necessary for obtaining a polynomial predictor solution from the subcell average data.

3.3.1 Finite-Volume Scheme

We integrate (3.1) over the space-time control volume $S_i \times [t^n, t^{n+1}]$ and obtain the standard finite volume (FV) discretisation

$$\mathbf{v}_i^{n+1} = \mathbf{v}_i^n - \frac{\Delta t}{\Delta x_{sub}}\left(\mathbf{f}_{i+\frac{1}{2}} - \mathbf{f}_{i-\frac{1}{2}}\right) + \Delta t\ \bar{\mathbf{s}}_i\ , \tag{3.23}$$

where

$$\mathbf{v}_i^n = \frac{1}{\Delta x_{sub}} \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \mathbf{u}_h^n\ dx\ , \tag{3.24a}$$

$$\bar{\mathbf{s}}_i = \frac{1}{\Delta t}\frac{1}{\Delta x_{sub}} \int_{t^n}^{t^{n+1}} \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \mathbf{S}\left(\mathbf{q}_h(x,t)\right)\ dx\ dt\ , \tag{3.24b}$$

$$\mathbf{f}_{i+\frac{1}{2}} = \frac{1}{\Delta t} \int_{t^n}^{t^{n+1}} \tilde{\mathbf{f}} \left(\mathbf{q}_h^-(x_{i+\frac{1}{2}}, t), \mathbf{q}_h^+(x_{i+\frac{1}{2}}, t) \right) \, dt \, . \tag{3.24c}$$

As defined earlier $\mathbf{v}_i^n$ is the spatial average of the polynomial $\mathbf{u}_h^n$ in cell S_i at time t^n, while $\bar{\mathbf{s}}_i$ are the space-time averaged sources. The time averages at the intercell boundary $x_{i+\frac{1}{2}}$ of the numerical fluxes $\tilde{\mathbf{f}}$ are denoted by $\mathbf{f}_{i+\frac{1}{2}}$. Again we use the local Lax-Friedrichs flux (3.14) and introduce the space-time predictor solution $\mathbf{q}_h$ (see (3.11)). Note that we use the same time step size Δt as in the topcell method, whereas the spatial cell size is smaller by a factor of η. Further details on the underlying Courant-Friedrichs-Lewy (CFL) condition are discussed in Chapter 4.

3.3.2 WENO Reconstruction Method

As we will see in Part II we need a polynomial representation of the data in cell S_i to obtain the space-time predictor solution required in (3.24c). Given the cell average values on the subgrid our task reads: "construct a polynomial of degree M in cell S_i from the cell averages of this cell and its neighbours". In principle such a reconstruction of order M is always possible starting from $M+1$ average values. Since the solution of (3.1) can develop discontinuities, a straightforward reconstruction using a fixed stencil would lead to oscillations in the respective cells. This is called the Gibbs phenomenon and does not represent the true physical solution. The general idea of a WENO reconstruction is thus to use multiple stencils and build a weighted combination of the corresponding polynomials, leading to a non-oscillatory interpolation.

Throughout the thesis we apply the WENO reconstruction method as proposed in [Dumbser et al., 2013]. Depending on the degree of the polynomial, which we want to reconstruct, we choose either three or four stencils for an even or odd degree,

Table 3.1: Summary of the values for the parameters $L = L(M, s)$ and $R = R(M, s)$ for even and odd M. Additionally we introduce the weights λ_s which differ for the centered and one-sided stencils. (see [Dumbser et al., 2013])

	M even		M odd		λ_s
$s=0$	$-$	$-$	$L = \text{floor}(M/2)+1$	$R = \text{floor}(M/2)$	10^5
$s=1$	$L = M/2$	$R = M/2$	$L = \text{floor}(M/2)$	$R = \text{floor}(M/2)+1$	10^5
$s=2$	$L = M$	$R = 0$	$L = M$	$R = 0$	1
$s=3$	$L = 0$	$R = M$	$L = 0$	$R = M$	1

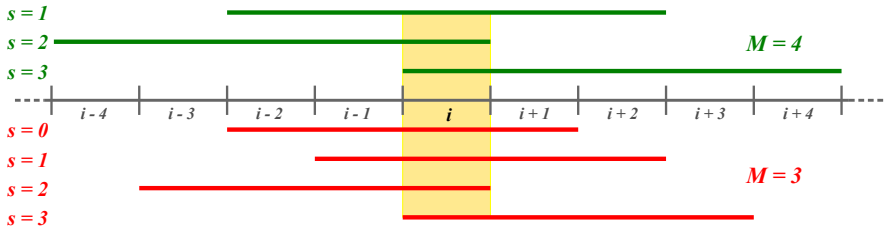


Figure 3.3: Exemplary representation of the stencils Γ_i^s for reconstruction polynomials in cell S_i (yellow) of even degree $M = 4$ (green) and odd degree $M = 3$ (red).

respectively. Each stencil Γ_i^s consists of $M + 1$ cells

$$\Gamma_i^s = \bigcup_{e=i-L}^{i+R} S_e \ , \tag{3.25}$$

where the superscript s denotes the stencil and $L = L(M, s)$ and $R = R(M, s)$ express the spatial extension of the stencil to the left and to the right. A summary of the values for L and R is shown in Table 3.1.

In Figure 3.3 we present two examples of which stencils we use for $M = 3$ and $M = 4$. We denote the reconstruction polynomial of stencil s with $\mathbf{w}_h^s$ and can express it in terms of our basis functions $\varphi_l(x)$

$$\mathbf{w}_h^s(x, t^n) = \varphi_l(x)\hat{\mathbf{w}}_{i,l}^{n,s} \ . \tag{3.26}$$

Next we require the polynomial $\mathbf{w}_h^s$ to be consistent with the cell average values $\mathbf{v}^n$ in the current stencil:

$$\frac{1}{\Delta x_e} \int_{x_{e-\frac{1}{2}}}^{x_{e+\frac{1}{2}}} \varphi_l(x)\hat{\mathbf{w}}_{i,l}^{n,s} \, dx = \mathbf{v}_e^n \ , \qquad \forall I_e \in \Gamma_i^s \ . \tag{3.27}$$

This is a system of $M+1$ equations for the $M+1$ unknown coefficients $\hat{\mathbf{w}}_{i,l}^s$ and can be solved by inverting the coefficient matrix, which depends only on the choice of our basis. In order to obtain the final reconstruction polynomial $\mathbf{w}_h$ for cell S_i, we have to construct a data-dependent nonlinear combination of these coefficients, i.e.

$$\mathbf{w}_h(x, t^n) = \varphi_l \hat{\mathbf{w}}_{i,l}^n \ , \qquad \text{with} \qquad \hat{\mathbf{w}}_{i,l}^n = \sum_s^{N_s} \omega_s \hat{\mathbf{w}}_{i,l}^{n,s} \ , \tag{3.28}$$

where N_s is the number of stencils as discussed before. We require

$$\omega_s \geq 0 \ , \qquad \sum_s^{N_s} \omega_s = 1 \tag{3.29}$$

for stability and consistency [Shu, 1997]. This leads to

$$\omega_s = \frac{\tilde{\omega}_s}{\sum_k \tilde{\omega}_k} , \qquad \text{with} \qquad \tilde{\omega}_s = \frac{\lambda_s}{(\sigma_s + \epsilon)^r} , \tag{3.30}$$

where we indroduce $\epsilon > 0$ to avoid the denominator to become 0. In all our numerical tests we set $\epsilon = 10^{-10}$ and $r = 4$. The oscillation indicator σ_s is given by

$$\sigma_s = \Sigma_{pm} \hat{\mathbf{w}}_{i,p}^{n,s} \hat{\mathbf{w}}_{i,m}^{n,s} , \tag{3.31}$$

with the precomputable oscillation indicator matrix

$$\Sigma_{pm} = \sum_{\alpha=1}^{M} \int_0^1 \frac{\partial^\alpha \varphi_p(\xi)}{\partial \xi^\alpha} \cdot \frac{\partial^\alpha \varphi_m(\xi)}{\partial \xi^\alpha} \, d\xi . \tag{3.32}$$

Note that this WENO reconstruction method produces entire polynomials, whereas the original WENO scheme of Jiang & Shu (see [Shu, 1997]) results in point values at the cell boundaries. As mentioned in [Dumbser et al., 2013] this is not optimal in the sense that we have to use larger stencils to obtain a given order of accuracy. Nevertheless, it is essential for the computation of the predictor that we have a polynomial representation of the data (see Part II).

Chapter 4

The Full Algorithm

After introducing the numerical methods employed on the top- and subgrid in the previous chapter, we now want to bring it all together and formulate the full algorithm, which is referred to as:

> [...] *a posteriori* finite volume subcell limiter for the Discontiuous Galerkin finite element method [Dumbser et al., 2014]

This is done in Section 4.1 where we will also discuss the proper CFL condition and give further insight on the relation between both grids.

In Section 4.2 we expand on the actual implementation of the algorithm. We rewrite (3.11) and (3.23) to obtain a matrix-vector form and show that the matrices can be precomputed analytically with the help of the Gauss-Legendre quadrature formula.

4.1 Description

A graphical summary of how one time step is performed by the algorithm is depicted in Figure 4.1. We start off with a polynomial representation $\mathbf{u}_h^n$ of degree M at time t^n on the topgrid with N_{top} cells. This means we have $M+1$ coefficients $\hat{\mathbf{u}}_l^n$ given at the Gauss-Legendre nodes in each cell T_i. From these we calculate locally (for each cell seperately) a weak solution $\mathbf{q}_h$ of our governing equation neglecting the influence of neighbouring cells completely. In Part II we will discuss two different ways of obtaining this predictor solution, which will be given as a space-time polynomial of degree M. The space-time predictor is the key ingredient to evaluate the numerical flux function $\tilde{\mathbf{f}}$ in (3.11), allowing us to compute a preliminary solution $\hat{\mathbf{u}}_h^{*,n+1}$.

Due to the properties of the DG scheme this candidate solution might contain

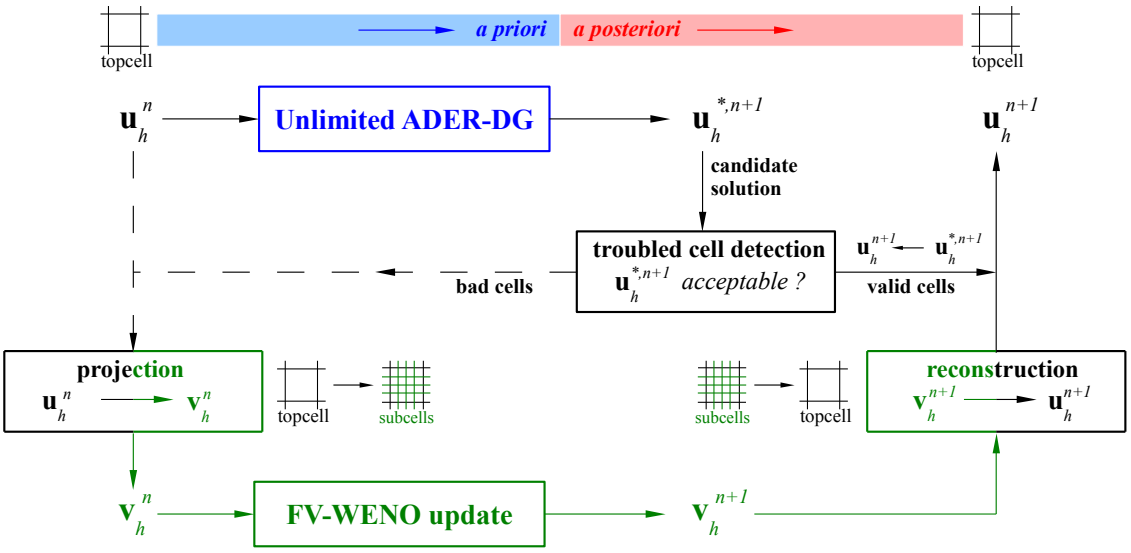


Figure 4.1: Schematic representation of the "*a posteriori* finite volume subcell limiter for the DG finite element method"[Dumbser et al., 2014]. Starting with the polynomial representation $\mathbf{u}_h^n$ we perform an unlimited ADER-DG step (3.11) to obtain a candidate solution $\mathbf{u}_h^{*,n+1}$. A troubled cell indicator determines whether this solution is valid or needs to be recomputed. Solutions in cells that pass the test are accepted without further ado. For all bad cells we scatter the old data $\mathbf{u}_h^n$ onto the subgrid and perform a FV-WENO method on the subcell averages $\mathbf{v}_h^n$. Its solution is gathered back onto the topgrid by means of the reconstruction operator (see Section 3.1), completing one time step (similar to [Dumbser et al., 2014]).

unphysical oscillations, which is why we test it on its physical and numerical admissibility applying the detection criteria (3.15) and (3.22). For all the cells, where the solution passes these tests, we declare the candidate solution to be the solution at time t^{n+1}. The solution of the cells which were marked troubled is recomputed on a subgrid with $N_{sub} = \eta \cdot N_{top}$ cells ($\eta = 2M + 1$).

To do so, the polynomial data $\mathbf{u}_h^n$ is projected on the subgrid where it is represented by N_{sub} cell averages $\mathbf{v}_h^n$. From these cell averages we reconstruct a polynomial $\mathbf{w}_h^n$ of degree M using the WENO reconstruction method. Given $\mathbf{w}_h^n$ we can again obtain a completely local solution by means of either of the two predictor methods to be shown. As in the DG scheme this predictor solution occurs in the numerical flux $\tilde{\mathbf{f}}$ but this time we apply the more robust FV scheme (3.23) and obtain new cell averages $\mathbf{v}_h^{n+1}$. The cell averages are gathered back onto the topgrid via the reconstruction operator (see (3.8)) and give us the polynomial result $\mathbf{u}_h^{n+1}$. In case of detecting a cell as troubled in two successive time steps, we do not use the reconstruced polynomial, but instead directly take the previous subcell average data to compute the time evolution.

CFL condition and optimal choice of η

In Equation (3.23) we have seen that the finite volume scheme, which updates the subcell averages, depends on the previous cell average and a flux term scaled with the grid parameters Δt and Δx. Closely related to the choice of these is the CFL condition. It is a necessary condition for stability and convergence to the solution of the differential equation (3.1). We define the CFL number C as

$$C = \frac{\Delta t \ |\lambda_{\max}|}{\Delta x} \ . \tag{4.1}$$

It can be seen as the ratio of the maximum characteristic speed $\lambda_{\max}$ to the grid speed $\frac{\Delta x}{\Delta t}$ defined by our discretisation. The free parameter is the time step size Δt, which ought to be chosen such that the true domain of dependency lies completely inside the numerical domain of dependency (see Figure 4.2), hence $C_{FV} \leq 1$. The word "true" refers to the fact that only information from this part of the cell ($C \cdot \Delta x$) can reach the boundary due to the characteristic speed $\lambda_{\max}$.

For the DG method the restriction on the CFL number for stability is more severe. It depends on the degree M of the polynomials used to represent the data. In one spatial dimension the CFL number has to satisfy (see [Dumbser et al., 2014])

$$C_{DG} = \frac{\Gamma}{2M+1} \qquad \text{with} \qquad 0 \leq \Gamma \leq 1 \ . \tag{4.2}$$

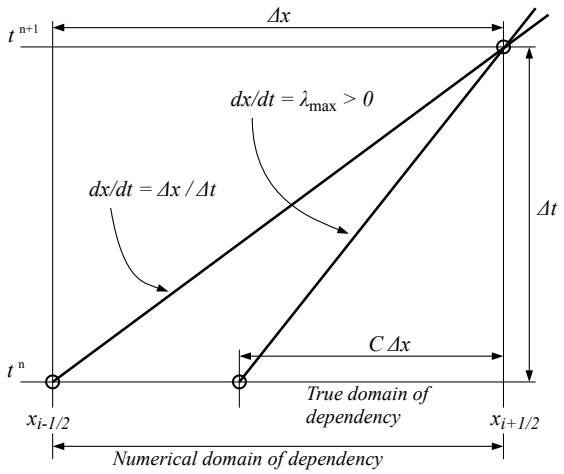


Figure 4.2: In our simulations we fix the numerical domain of dependency by choosing a certain grid spacing Δx. Given a characteristic speed $\lambda_{\max}$ we are left with choice of the time step size Δt. In order to guarantee stability we have to assure that the true domain of dependency lies completely inside the numerical domain of dependency by choosing an appropriate CFL number C (similar to [Toro, 2009]).

Provided that we want to use the same time step size on both grids, (4.2) gives an expression for the CFL number on the subgrid

$$C_{FV} = \frac{\Gamma}{(2M+1)} \frac{\Delta x_{top}}{\Delta x_{sub}} \ . \tag{4.3}$$

From [Dumbser et al., 2014] we know that the error terms of the finite volume scheme (3.23) contain the factor $(1 - C_{FV})$, which means it is preferable to take a larger CFL number. This is the reason why we take the number of subcells for one topcell as $\eta = 2M + 1$, i.e. $\Delta x_{top} = (2M + 1)\Delta x_{sub}$ (see (4.3)). In principle it would be sufficient to represent the full polynomial information by $M + 1$ subcell averages, but this has the drawback of a CFL number C_{FV} being only about half of the maximal admissible value. Taking $\eta > 2M + 1$ is also suboptimal, because it would further restrict the (already small) time step size on the topgrid.

4.2 Implementation Details

After deriving and discussing the full algorithm this section will give further insight on the practical details of the actual implementation. To be more specific, all the calculations were rewritten in matrix-vector notation with the advantage of being able to compute many matrices in advance of the simulation with analytic precision. In order to do so we apply the Gaussian quadrature rule (see Section 4.2.1). In Section 4.2.2 we show how to rewrite the DG scheme (3.11) as well as the finite volume scheme (3.23) and give analytic expressions for the precomputable matrices.

4.2.1 Gauss-Legendre Quadrature

The n-point Gauss-Legendre quadrature rule is an approximative method to compute the definite integral of an arbitrary function. Conventionally the domain of integration is taken to be $[-1, 1]$, hence

$$\int_{-1}^{1} f(x)\ dx \approx \sum_{i=1}^{n} w_i f(x_i) \ . \tag{4.4}$$

Note that for all the applications of the quadrature formula we explicitly do not use the Einstein summation convention. The quadrature formula is constructed such that one obtains the exact result for polynomials up to degree $2n - 1$ for certain weights w_i and points x_i. In particular the x_i are the roots of the associated Legendre polynomials $P_n(x)$. Since we mainly work on the unit interval $I = [0, 1]$ we have to

transform (4.4)

$$\int_0^1 f(x)\ dx = \frac{1}{2}\sum_{i=1}^{n} w_i\ f\left(\frac{1}{2}x_i + \frac{1}{2}\right) = \frac{1}{2}\sum_{i=1}^{n} w_i\ f(\lambda_i)\ , \tag{4.5}$$

where we returned to the Gauss-Legendre nodes λ_i, which were used to define the basis functions (see (3.5)). The weights w_i are given by

$$w_i = \frac{2}{(1-x_i^2)[P_n'(x_i)]^2}\ . \tag{4.6}$$

In the following we will see that most of the integrations we perform include our polynomial basis functions. Since we use $M+1$ Gauss-Legendre nodes λ_i in each cell, it is in principle possible to obtain exact integration results for polynomials of degree $2M+1$ or less with (4.5).

4.2.2 Matrices for the DG & FV Schemes

Before we can rewrite (3.11) and (3.23) into a matrix-vector form, we anticipate the representation of the space-time predictor $\mathbf{q}_h$, which will be introduced in more detail later (see (5.7)). Similar to (3.7) we write

$$\mathbf{q}_h(\xi,\tau) = \varphi_p(\xi)\varphi_s(\tau)\hat{\mathbf{q}}_{ps}\ . \tag{4.7}$$

DG-scheme

Inserting (4.7) into the explicit form of our numerical flux function (3.14), i.e. the local Lax-Friedrichs flux, we obtain

$$\begin{aligned}\mathbf{M}_{kl}\left[\hat{\mathbf{u}}_l^{n+1} - \hat{\mathbf{u}}_l^{n}\right] &+ \mathbf{G}^{11}_{kps}\left[\mathbf{F}(\hat{\mathbf{q}}^{(i)}_{ps}) + |\lambda_{\max}|\hat{\mathbf{q}}^{(i)}_{ps}\right] + \mathbf{G}^{10}_{kps}\left[\mathbf{F}(\hat{\mathbf{q}}^{(i+1)}_{ps}) - |\lambda_{\max}|\hat{\mathbf{q}}^{(i+1)}_{ps}\right]\\ &- \mathbf{G}^{00}_{kps}\left[\mathbf{F}(\hat{\mathbf{q}}^{(i)}_{ps}) - |\lambda_{\max}|\hat{\mathbf{q}}^{(i)}_{ps}\right] - \mathbf{G}^{01}_{kps}\left[\mathbf{F}(\hat{\mathbf{q}}^{(i-1)}_{ps}) + |\lambda_{\max}|\hat{\mathbf{q}}^{(i-1)}_{ps}\right]\\ &- \mathbf{L}_{kps}\mathbf{F}(\hat{\mathbf{q}}_{ps}) - \mathbf{H}_{kps}\mathbf{S}(\hat{\mathbf{q}}_{ps}) = \mathbf{0}\ ,\end{aligned} \tag{4.8}$$

with

$$\mathbf{M}_{kl} = \int_0^1 \varphi_k(\xi)\varphi_l(\xi)d\xi\ , \tag{4.9a}$$

$$\mathbf{G}^{ab}_{kps} = \frac{1}{2}\frac{\Delta t}{\Delta x_{top}}\ \varphi_k(a)\varphi_p(b)\int_0^1 \varphi_s(\tau)d\tau = \frac{1}{2}\frac{\Delta t}{\Delta x_{top}}\ \varphi_k(a)\varphi_p(b)\mathbf{A}_s\ , \tag{4.9b}$$

$$\mathbf{L}_{kps} = \frac{\Delta t}{\Delta x_{top}}\int_0^1 \frac{\partial\varphi_k(\xi)}{\partial\xi}\varphi_p(\xi)d\xi\ \int_0^1 \varphi_s(\tau)d\tau = \frac{\Delta t}{\Delta x_{top}}\mathbf{S}_{kp}\mathbf{A}_s\ , \tag{4.9c}$$

$$\mathbf{H}_{kps} = \Delta t \int_0^1 \varphi_k(\xi)\varphi_p(\xi)d\xi \int_0^1 \varphi_s(\tau)d\tau \ = \Delta t \mathbf{M}_{kp}\mathbf{A}_s \ . \tag{4.9d}$$

All matrices are thus proportional to combinations of the mass matrix $\mathbf{M}_{ab}$, the stiffness matrix $\mathbf{S}_{ab}$ and the average matrix $\mathbf{A}_s$. The grid parameter Δt and Δx arise from the transformation onto the space-time reference interval $[0,1]^2$. Note that the term which included the numerical flux function results in four terms which take the neighbouring cells into account (see Figure 3.2).

All the integrals in (4.9) contain polynomials of degree $2M$ or less and can thus be computed exactly with the Gauss-Legendre quadrature formula (4.5). Here we want to show how this is done explicitly for the components of matrix $\mathbf{H}_{kps}$ and refer to the Appendix A.1 for the calculations of the remaining ones:

$$\begin{aligned}
\mathbf{H}_{kps} &= \Delta t \int_0^1 \varphi_k(\xi)\varphi_p(\xi)d\xi \int_0^1 \varphi_s(\tau)d\tau \\
&= \frac{\Delta t}{4} \sum_{i=0}^{M} w_i \varphi_k(\lambda_i)\varphi_p(\lambda_i) \ \sum_{j=0}^{M} w_j \varphi_s(\lambda_j) \\
&= \frac{\Delta t}{4} \sum_{i=0}^{M} w_i \delta_{ki}\delta_{pi} \ \sum_{j=0}^{M} w_j \delta_{sj} \\
&= \frac{\Delta t}{4} \ w_k \delta_{kp} \ w_s \ .
\end{aligned} \tag{4.10}$$

In the actual implementation we also precompute the multiplication with the inverse mass matrix $\mathbf{M}_{kl}^{-1}$ in (4.8) and only store the resulting matrices.

FV-scheme

Analogous to the DG-scheme we rewrite (3.23) which yields

$$\begin{aligned}
\mathbf{v}_i^{n+1} = \mathbf{v}_i^n &- \tilde{\mathbf{G}}_{ps}^0 \left[\mathbf{F}(\hat{\mathbf{q}}_{ps}^{(i+1)}) - \mathbf{F}(\hat{\mathbf{q}}_{ps}^{(i)}) + |\lambda_{\max}| \left(\hat{\mathbf{q}}_{ps}^{(i)} - \hat{\mathbf{q}}_{ps}^{(i+1)}\right)\right] \\
&- \tilde{\mathbf{G}}_{ps}^1 \left[\mathbf{F}(\hat{\mathbf{q}}_{ps}^{(i)}) - \mathbf{F}(\hat{\mathbf{q}}_{ps}^{(i-1)}) + |\lambda_{\max}| \left(\hat{\mathbf{q}}_{ps}^{(i)} - \hat{\mathbf{q}}_{ps}^{(i-1)}\right)\right] + \tilde{\mathbf{H}}_{ps}\mathbf{S}(\hat{\mathbf{q}}_{ps}^{(i)}),
\end{aligned} \tag{4.11}$$

with

$$\tilde{\mathbf{G}}_{ps}^a = \frac{1}{2}\frac{\Delta t}{\Delta x_{sub}} \ \varphi_p(a) \int_0^1 \varphi_s(\tau)d\tau = \frac{1}{2}\frac{\Delta t}{\Delta x_{sub}} \ \varphi_p(a)\mathbf{A}_s \ , \tag{4.12a}$$

$$\tilde{\mathbf{H}}_{ps} = \Delta t \int_0^1 \varphi_p(\xi)d\xi \ \int_0^1 \varphi_s(\tau)d\tau \ = \Delta t \mathbf{A}_p \mathbf{A}_s \ . \tag{4.12b}$$

For further details on the respective calculations see Appendix A.2.

Part II

Space-Time Predictor Methods

Chapter 5

Iteration Scheme by Dumbser et al.

One way to obtain the local space-time predictor, needed to evaluate the numerical flux function in either the ADER-DG scheme (3.11) on the topgrid or the FV scheme (3.23) on the subgrid, is given in [Dumbser et al., 2008a; Dumbser et al., 2013; Dumbser et al., 2014]. They derive a weak formulation of the governing PDE and solve the problem with an iterative procedure obtaining a space-time polynomial of degree M in both space and time. A summary of the derivation is given in Section 5.1. In Section 5.2 we show that the procedure converges within $M+1$ iterations for the linear case regardless of the initial condition.

5.1 Derivation of Weak Solution

For computational convenience we introduce spatial and temporal reference coordinates (ξ, τ), namely

$$x = x_{i-\frac{1}{2}} + \xi \ \Delta x \ , \qquad t = t^n + \tau \ \Delta t \ , \tag{5.1}$$

which transform our regular space-time control volume $[x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}}] \times [t^n, t^{n+1}]$ onto the space-time reference control volume $[0,1] \times [0,1]$. As a result we can rewrite the governing PDE (3.1) as

$$\frac{\partial \mathbf{U}}{\partial \tau} + \frac{\partial \mathbf{F}^*(\mathbf{U})}{\partial \xi} = \mathbf{S}^* \ , \tag{5.2}$$

with

$$\mathbf{F}^* = \frac{\Delta t}{\Delta x}\mathbf{F} \ , \qquad \mathbf{S}^* = \Delta t \ \mathbf{S} \ . \tag{5.3}$$

Note that this is applicable on both grids with $\Delta x = \Delta x_{top}$ and $\Delta x = \Delta x_{sub}$ for the topgrid and the subgrid, respectively. We introduce the space-time basis functions $\theta_{ps} = \theta_{ps}(\xi, \tau)$, which are piecewise space-time polynomials of degree M, as the tensor-product of the Lagrange basis functions φ_l (3.5)

$$\theta_{ps}(\xi, \tau) = \varphi_p(\xi) \ \varphi_s(\tau) \ . \tag{5.4}$$

Multiplying (5.2) with a space-time test function θ_{qr} and integrating over the reference control volume $[0,1]^2$ yields

$$\int_0^1 \int_0^1 \theta_{qr} \left(\frac{\partial \mathbf{U}}{\partial \tau} + \frac{\partial \mathbf{F}^*}{\partial \xi} - \mathbf{S}^* \right) \, d\xi \, d\tau = 0 \, . \tag{5.5}$$

The next step is to integrate the first term which contains the time derivative by parts:

$$\begin{aligned} &\int_0^1 \theta_{qr}(\xi,1)\mathbf{U}(\xi,1) \, d\xi - \int_0^1 \int_0^1 \left(\frac{\partial}{\partial \tau} \theta_{qr} \right) \mathbf{U} \, d\xi \, d\tau \\ &+ \int_0^1 \int_0^1 \theta_{qr} \left[\frac{\partial \mathbf{F}^*}{\partial \xi} - \mathbf{S}^* \right] d\xi \, d\tau = \int_0^1 \theta_{qr}(\xi,0)\mathbf{U}(\xi,0) \, d\xi \, . \end{aligned} \tag{5.6}$$

Depending on the grid, $\mathbf{U}(\xi,0)$ is either given by the initial data $\mathbf{u}_h^n$ (topgrid) or by the WENO reconstruction $\mathbf{w}_h$ of the cell average data (subgrid). For simplicity we will only use $\mathbf{u}_h^n$ in the upcoming equations, keeping the second meaning in mind.

In the following, we indicate the discrete numerical space-time solution for $\mathbf{U}$ by $\mathbf{q}_h$, for which we make the ansatz

$$\mathbf{q}_h = \mathbf{q}_h(\xi,\tau) = \theta_{ps}(\xi,\tau)\hat{\mathbf{q}}_{ps} \, , \tag{5.7}$$

which also applies to the fluxes and source terms, hence

$$\mathbf{F}^* = \theta_{ps}\hat{\mathbf{F}}^*_{ps} \, , \qquad \mathbf{S}^* = \theta_{ps}\hat{\mathbf{S}}^*_{ps} \, . \tag{5.8}$$

Due to our choice of the Lagrange polynomials defined on the associated Gauss-Legendre nodes, the coefficients of the fluxes and source terms are explicitly given by the pointwise evaluation of the physical fluxes and source terms, namely

$$\hat{\mathbf{F}}^*_{ps} = \mathbf{F}^*(\hat{\mathbf{q}}_{ps}) \, , \qquad \hat{\mathbf{S}}^*_{ps} = \mathbf{S}^*(\hat{\mathbf{q}}_{ps}) \, . \tag{5.9}$$

By using the ansatz (5.7) and (5.8) the weak formulation of the governing PDE can be rewritten,

$$\begin{aligned} &\int_0^1 \theta_{qr}(\xi,1)\theta_{ps}(\xi,1)\hat{\mathbf{q}}_{ps} \, d\xi - \int_0^1 \int_0^1 \left(\frac{\partial}{\partial \tau} \theta_{qr} \right) \theta_{ps}\hat{\mathbf{q}}_{ps} \, d\xi \, d\tau \\ &+ \int_0^1 \int_0^1 \theta_{qr} \left[\frac{\partial}{\partial \xi} \theta_{ps}\mathbf{F}^*(\hat{\mathbf{q}}_{ps}) - \theta_{ps}\mathbf{S}^*(\hat{\mathbf{q}}_{ps}) \right] d\xi \, d\tau = \int_0^1 \theta_{qr}(\xi,0)\varphi_l \hat{\mathbf{u}}_l^n \, d\xi \, . \end{aligned} \tag{5.10}$$

This equation is an element-local nonlinear algebraic equation system for the unknown coefficients of the space-time predictor $\mathbf{q}_h$. For convenience we introduce a

suitable matrix notation, which allows us to write (5.10) in a compact matrix-vector form

$$\mathbf{K}^1_{qrps}\hat{\mathbf{q}}_{ps} + \mathbf{K}^{\xi}_{qrps}\mathbf{F}^*(\hat{\mathbf{q}}_{ps}) - \mathbf{M}_{qrps}\mathbf{S}^*(\hat{\mathbf{q}}_{ps}) = \mathbf{I}^0_{qrl}\hat{\mathbf{u}}^n_l \ , \tag{5.11}$$

with

$$\mathbf{K}^1_{qrps} = \int_0^1 \theta_{qr}(\xi,1)\ \theta_{ps}(\xi,1)\ d\xi - \int_0^1\int_0^1 \left(\frac{\partial}{\partial\tau}\theta_{qr}\right)\theta_{ps}\ d\xi\ d\tau \ , \tag{5.12a}$$

$$\mathbf{K}^{\xi}_{qrps} = \int_0^1\int_0^1 \theta_{qr}\left(\frac{\partial}{\partial\xi}\theta_{ps}\right) d\xi\ d\tau \ , \tag{5.12b}$$

$$\mathbf{M}_{qrps} = \int_0^1\int_0^1 \theta_{qr}\theta_{ps}\ d\xi\ d\tau \ , \tag{5.12c}$$

$$\mathbf{I}^0_{qrl} = \int_0^1 \theta_{qr}(\xi,0)\ \varphi_l\ d\xi \ . \tag{5.12d}$$

All integrals in (5.12) include polynomials of order at most $2M$ and can thus be integrated analytically by Gaussian quadrature using the $M+1$ collocation points in spatial and temporal direction. For details, see section 4.2.

As proposed in [Dumbser et al., 2008a] and later works the system (5.11) is solved iteratively

$$\mathbf{K}^1_{qrps}\hat{\mathbf{q}}^{(k+1)}_{ps} = \mathbf{I}^0_{qrl}\hat{\mathbf{u}}^n_l - \mathbf{K}^{\xi}_{qrps}\mathbf{F}^*(\hat{\mathbf{q}}^{(k)}_{ps}) + \mathbf{M}_{qrps}\mathbf{S}^*(\hat{\mathbf{q}}^{(k)}_{ps}) \ . \tag{5.13}$$

For stiff equations the source term should be taken implicitely [Hidalgo and Dumbser, 2011]. As initial data we set the coefficients $\hat{\mathbf{q}}^{(0)}_{ps} = \hat{\mathbf{u}}^n_p \ , \ \forall s$. This is equivalent to the assumption that the solution is constant over time. In [Hidalgo and Dumbser, 2011] another strategy for the initial guess is discussed including information of the solution of the PDE. This did not lead to a significant improvement in our tests and was thus not further investigated.

In the following section we will show that the iteration (5.13) for linear HCL converges to the analytic solution in a certain number of steps for any initial data. All matrices involved in the iteration scheme can be computed analytically beforehand. Further details are shown in the Appendix A.3.

5.2 Proof of Convergence

We want to show that the iteration scheme (5.13) to obtain the local space-time predictor converges in $M+1$ iterations independent of the initial data. For simplicity

we want to restrict ourselves to the linear case ($\mathbf{F}(\mathbf{q}) = a\mathbf{q}$) with $\mathbf{S} \equiv \mathbf{0}$ to strictly prove this statement. Nevertheless, in Chapter 7 we will see that it seems to hold for nonlinear fluxes and non-vanishing source terms as well.

Multiplying (5.13) with the inverse of $\mathbf{K}^1$ and using (5.3) yields

$$\hat{\mathbf{q}}_{ps}^{(k+1)} = \left(\mathbf{K}^1_{qrps}\right)^{-1} \mathbf{I}^0_{qrl} \hat{\mathbf{u}}^n_l - a\frac{\Delta t}{\Delta x} \left(\mathbf{K}^1_{qrps}\right)^{-1} \mathbf{K}^{\xi}_{qrp's'} \hat{\mathbf{q}}^{(k)}_{p's'} \ . \tag{5.14}$$

The first term does not depend on the current iteration stage k and is thus constant in that sense. It can be computed once for each iteration and will further be denoted by $\hat{\mathbf{c}}_{ps}$. For the second term we will introduce the abbreviation $\mathbf{B} = (\mathbf{K}^1)^{-1}\mathbf{K}^{\xi}$ and end up with,

$$\hat{\mathbf{q}}_{ps}^{(k+1)} = \hat{\mathbf{c}}_{ps} - a\frac{\Delta t}{\Delta x} \mathbf{B}_{psp's'} \hat{\mathbf{q}}^{(k)}_{p's'} \ . \tag{5.15}$$

This is a fixed point iteration, usually written as $x_{k+1} = g(x_k)$, where we want to find the fixed point of g, i.e. $g(x) = x$. In order to investigate the convergence we will look at the difference of the coefficients $\hat{\mathbf{q}}_{ps}$ in two successive iteration steps

$$\begin{aligned}
\hat{\mathbf{q}}_{ps}^{(k+1)} - \hat{\mathbf{q}}_{ps}^{(k)} &= \hat{\mathbf{c}}_{ps} - a\frac{\Delta t}{\Delta x} \mathbf{B}_{psp's'} \hat{\mathbf{q}}^{(k)}_{p's'} - \hat{\mathbf{c}}_{ps} - a\frac{\Delta t}{\Delta x} \mathbf{B}_{psp's'} \hat{\mathbf{q}}^{(k-1)}_{p's'} \\
&= -a\frac{\Delta t}{\Delta x} \mathbf{B}_{psp's'} \left(\hat{\mathbf{q}}^{(k)}_{p's'} - \hat{\mathbf{q}}^{(k-1)}_{p's'} \right) \\
&\vdots \\
&= \left(-a\frac{\Delta t}{\Delta x} \right)^k \left(\mathbf{B}_{psp's'}\right)^k \left(\hat{\mathbf{q}}^{(1)}_{p's'} - \hat{\mathbf{q}}^{(0)}_{p's'} \right) \ ,
\end{aligned} \tag{5.16}$$

and find that it depends on a non-zero coefficient containing the grid spacings, the initial condition $\hat{\mathbf{q}}^{(0)}$ and the kth (and k+1th) power of the matrix $\mathbf{B}$. In the following we will show that for a certain $k = k'$ all entries of the matrix are equal to zero and thus the difference $\hat{\mathbf{q}}^{(k'+1)} - \hat{\mathbf{q}}^{(k')}$. This means the fixed point of g is $\hat{\mathbf{q}}^{(k')}$.

Combining one spatial and one temporal index (see [Dumbser et al., 2013]) we can identify $\mathbf{B}_{psp's'}$ as a $(M+1)^2 \times (M+1)^2$ matrix. In order to calculate its powers we will introduce the similarity transformation $\mathbf{B} = \mathbf{P}\mathbf{J}\mathbf{P}^{-1}$, where $\mathbf{J}$ is the Jordan normal form of $\mathbf{B}$. This allows us to reduce the powers of $\mathbf{B}$ to the powers of $\mathbf{J}$. In general the Jordan matrix is a block diagonal matrix $\mathbf{J} = \mathrm{diag}(\mathbf{J}_1, \ldots, \mathbf{J}_m)$, where the blocks are square matrices of the form

$$\mathbf{J}_i = \begin{pmatrix} \lambda_i & 1 & & \\ & \lambda_i & \ddots & \\ & & \ddots & 1 \\ & & & \lambda_i \end{pmatrix} \ , \tag{5.17}$$

with the eigenvalues λ_i of $\mathbf{B}$. For a given λ_i the number of Jordan blocks $\mathbf{J}_i$ corresponds to the geometric multiplicity of said eigenvalue, which is given by the

dimension of the associated eigenspace [Horn and Johnson, 1985]. From [Dumbser et al., 2008a; Dumbser et al., 2013; Dumbser et al., 2014] we know that $\mathbf{B}$ has the remarkable property that all eigenvalues are zero. The geometric multiplicity is given by

$$\dim\ker(\mathbf{B}) = \dim(\mathbf{B}) - \operatorname{rank}(\mathbf{B}) = M + 1 \ , \tag{5.18}$$

meaning $\mathbf{J}$ consists of $M+1$ blocks of the form

$$\mathbf{J}_i = \begin{pmatrix} 0 & 1 & & \\ & 0 & \ddots & \\ & & \ddots & 1 \\ & & & 0 \end{pmatrix} \ , \tag{5.19}$$

of yet unknown size. With the expression

$$2 \cdot \dim\ker(\mathbf{B}^j) - \dim\ker(\mathbf{B}^{j-1}) - \dim\ker(\mathbf{B}^{j+1}) \ , \tag{5.20}$$

we can calculate the number of Jordan blocks of size $j \times j$. It turns out that all blocks have the same size ($j = M+1$). In other words, $\mathbf{J}$ consists of $M+1$ nilpotent block matrices (5.19) of degree $M+1$. We can thus conclude

$$\mathbf{J}_i^{M+1} = \mathbf{0} \ \Rightarrow \mathbf{J}^{M+1} = \mathbf{0} \ \Rightarrow \mathbf{B}^{M+1} = \mathbf{0} \ , \tag{5.21}$$

proving the statement that the iteration scheme (5.15) converges in $M+1$ iteration steps.

All computations regarding the Jordan matrix and its properties were done using *Mathematica*. The corresponding notebook can be found on the CD attached to the thesis.

Chapter 6

Runge-Kutta Predictor Method

In this chapter we derive a new alternative to the iteration procedure (see Chapter 5) to obtain a local space-time predictor solution of (3.1). It is based on the Runge-Kutta method. In Section 6.1 we give a short summary of the basics of Runge-Kutta methods (see [Hairer et al., 1993]). The idea to produce a continuous output based on the (already computed) right hand sides is called dense output and will be discussed in Section 6.2 (see [Owren and Zennaro, 1991; Hairer et al., 1993]). With this knowledge we are able to derive the local Runge-Kutta predictor in Section 6.3.

6.1 Runge-Kutta Basics

More than one hundred years ago the two mathematicians C. Runge and M.W. Kutta developed a technique to solve the IVP for ordinary differential equations (ODEs)

$$\mathbf{y}'(x) = \mathbf{g}(x, \mathbf{y}(x)) \ , \qquad \mathbf{y}(x_0) = \mathbf{y}_0 \ , \tag{6.1}$$

where $\mathbf{y}_0, \mathbf{y}(x) \in \mathbb{R}^n$ and $x \in \mathbb{R}$.

Definition 6.1 (Explicit Runge-Kutta method). *Let s be an integer and a_{21}, a_{31}, a_{32}, ..., a_{s1}, a_{s2}, ..., $a_{s\ s-1}$, b_1, ..., b_s, c_2, ..., c_s be real coefficients. Then the method*

$$\begin{aligned}
\mathbf{k}_1 &= \mathbf{g}(x_0, \mathbf{y}_0) \\
\mathbf{k}_2 &= \mathbf{g}(x_0 + c_2 h, \mathbf{y}_0 + h a_{21} \mathbf{k}_1) \\
\mathbf{k}_3 &= \mathbf{g}(x_0 + c_3 h, \mathbf{y}_0 + h(a_{31} \mathbf{k}_1 + a_{32} \mathbf{k}_2)) \\
&\dots \\
\mathbf{k}_s &= \mathbf{g}(x_0 + c_s h, \mathbf{y}_0 + h(a_{s1} \mathbf{k}_1 + \dots + a_{s\ s-1} \mathbf{k}_{s-1})) \\
\mathbf{y}_1 &= \mathbf{y}_0 + h(b_1 \mathbf{k}_1 + \dots + b_s \mathbf{k}_s)
\end{aligned} \tag{6.2}$$

is called an s-stage explicit Runge-Kutta method (ERK) for (6.1)*. h is the step size of the scheme.*

The coefficients are chosen such that the Taylor series for the exact solution $\mathbf{y}(x_0+h)$ and for $\mathbf{y}_1$ cancel each other up to a term h^p, where p is the order of the method. The specific choice is not necessarily unique, meaning that there might be several sets of coefficients leading to a scheme of order p. A lot of research has been undertaken in order to find the "optimal" formulas. Firstly, to reduce the need for computer storage. Secondly, to reduce the error term (terms in h^{p+1}) as much as possible. In the 1960s it became customary to depict the coefficients in a Butcher tableau

$$\begin{array}{c|ccccc} 0 & & & & & \\ c_2 & a_{21} & & & & \\ c_3 & a_{31} & a_{32} & & & \\ \vdots & \vdots & \vdots & \ddots & & \\ c_s & a_{s1} & a_{s2} & \dots & a_{s\ s-1} & \\ \hline & b_1 & b_2 & \dots & b_{s-1} & b_s \end{array}$$

The Butcher tableaus we use for our tests can be found in the Appendix B.1.

In general one needs $s \geq p$ stages to construct an ERK of order p. Butcher proved that there is no ERK of order $p \geq 5$ with $s = p$ stages. Furthermore, he showed similar relations for even higher orders. Finding the minimal number of stages s for a scheme of order p is still an unsolved problem. In Table 6.1 you can see a summary for schemes up to order eight.

Table 6.1: Minimal number of stages s needed to construct an explicit Runge-Kutta method of order p.

p	2	3	4	5	6	7	8
$\min s$	2	3	4	6	7	9	11

6.2 Dense Output

The explicit Runge-Kutta methods defined in the previous section become rather inefficient if one wants to obtain a large number of output points (small h). A solution to this problem is the dense output method, which is a continuous extension providing a numerical approximation to $\mathbf{y}(x_0+\vartheta h)$ for the whole integration interval $0 \leq \vartheta \leq 1$. In the best case we can do this without any additional function evaluation, but similar to the Butcher barriers (see Table 6.1) there are certain limits as

Table 6.2: Minimal number of stages s^* needed to construct a continuous explicit Runge-Kutta method with polynomial approximation of degree p^*.

p^*	1	2	3	4	5	6	7
$\min s^*$	1	2	4	6	8	13	16

we will see in the following [Owren and Zennaro, 1991].

Similar to (6.2), we consider methods of the form

$$\mathbf{k}_i = \mathbf{g}\left(x_0 + c_i h, \mathbf{y}_0 + h\sum_{j=1}^{i-1} a_{ij}\mathbf{k}_j\right) , \qquad i = 1, \dots, s^* \tag{6.3a}$$

$$\mathbf{y}(x_0 + \vartheta h) = \mathbf{y}_0 + h\sum_{i=1}^{s^*} b_i(\vartheta)\mathbf{k}_i , \qquad \vartheta \in [0,1] \tag{6.3b}$$

with s^* stages, where the $b_i(\vartheta)$ are polynomials of degree $\leq p^*$. We require $b_i(0) = 0,\ i = 1, \dots, s^*$ for consistency. Furthermore, it should be clear that for $s = s^*$ we can recover a discrete ERK method with constant coefficients $b_i = b_i(1),\ i = 1, \dots, s$.

From the discussions concerning the DG method we know that a representation of the data by polynomials of order M leads to a method of order $M + 1$. It is thus sufficient to require $p^* = p - 1$ to preserve the order of the respective discrete ERK method.

[Owren and Zennaro, 1991] provide theoretical background on how many additional stages $(s^* - s)$ are needed to obtain said methods. Their results up to $p^* = 5$ are shown in Table 6.2 together with non-proved estimates for higher orders from [Hairer et al., 1993].

Let us point out a few important examples, namely the "classic" Runge-Kutta method ($p = 4,\ s = 4$) and the fifth order method by Dormand and Prince ($p = 5,\ s = 6$). Comparing the order and stage parameters with Table 6.2 it becomes clear that for both of them it should be possible to construct a continuous extension of order $p^* = p - 1$ without any additional stages. The respective polynomials $b_i(\vartheta)$ are summarised in the appendix B.2 together with the formulas we use for other orders. For $p^* > 4$ the number of additional stages grows monotonously, e.g. the scheme with $p^* = 7$ needs $16 - 11 = 5$ additional stages.

6.3 Local Runge-Kutta Predictor

Before we can apply a Runge-Kutta method (6.3) on our PDE (5.2) (in reference coordinates) we have to bring it in the form of an ODE (6.1). This will be done by discretising the spatial derivative but leaving the temporal derivative as is, which

is usually called the method of lines. In order to do so we use our ansatz for the conserved variables $\mathbf{U}$ (see (3.7)), namely the representation as spatial polynomials of degree M with time dependent coefficients $\hat{\mathbf{u}}_l(\tau)$:

$$\frac{\partial \mathbf{u}_h}{\partial \tau} = -\frac{\Delta t}{\Delta x} \frac{\partial \mathbf{F}(\mathbf{u}_h)}{\partial \xi} + \Delta t \ \mathbf{S}(\mathbf{u}_h) \tag{6.4a}$$

$$\varphi_l(\xi)\frac{\partial \hat{\mathbf{u}}_l(\tau)}{\partial \tau} = -\frac{\Delta t}{\Delta x} \mathbf{F}(\hat{\mathbf{u}}_k) \frac{\partial \varphi_k(\xi)}{\partial \xi} + \Delta t \ \mathbf{S}(\hat{\mathbf{u}}_k) \ \varphi_k(\xi) \ , \tag{6.4b}$$

where we used the properties (5.9) of the nodal basis. In the end, we want to obtain a local space-time predictor $\mathbf{q}_h(\xi,\tau) = \varphi_q(\xi)\varphi_r(\tau)\hat{\mathbf{q}}_{qr}$, which is uniquely defined by its coefficients $\hat{\mathbf{q}}_{qr}$. Since these coincide with the values at the Gauss-Legendre nodes ($\varphi_a(\xi_b) = \delta_{ab}$), it is sufficient to evaluate (6.4b) at $\xi = \xi_q$. This gives us $M+1$ ODEs for the $M+1$ unknowns $\hat{\mathbf{u}}_q$

$$\frac{\partial \hat{\mathbf{u}}_q(\tau)}{\partial \tau} = -\frac{\Delta t}{\Delta x} \mathbf{F}(\hat{\mathbf{u}}_k) \ \mathbf{D}_{kq} + \Delta t \ \mathbf{S}(\hat{\mathbf{u}}_q) \ . \tag{6.5}$$

The matrix $\mathbf{D}_{kq} = \frac{\partial \varphi_k}{\partial \xi}(\xi_p)$ is called the derivative matrix and can be computed beforehand. Note that there is still a summation over k on the right hand side, meaning that we have a coupled system of $M+1$ equations.

To summarise, we have to calculate the $\mathbf{k}_q^{(i)}$

$$\mathbf{k}_q^{(i)} = \mathbf{g}\left(\hat{\mathbf{u}}_q(\tau=0) + \sum_{j=1}^{i-1} a_{ij} \ \mathbf{k}_q^{(j)}\right) \ , \qquad i = 1, \ldots, s^* \tag{6.6}$$

with

$$\mathbf{g}(\hat{\mathbf{u}}_q) = -\frac{\Delta t}{\Delta x} \mathbf{F}(\hat{\mathbf{u}}_k) \ \mathbf{D}_{kq} + \Delta t \ \mathbf{S}(\hat{\mathbf{u}}_q) \ , \tag{6.7}$$

in order to calculate the polynomial approximation in time for each Gauss-Legendre node in space

$$\hat{\mathbf{u}}_q(\vartheta) = \hat{\mathbf{u}}_q(0) + \sum_{i=1}^{s^*} b_i(\vartheta) \ \mathbf{k}_q^{(i)} \ , \qquad \vartheta \in [0,1] \ . \tag{6.8}$$

Due to the matrix-vector notation of the right hand side (6.7) we compute all $\mathbf{k}_q^{(i)}$ for a certain stage i simultaneously. As a last step we read off the coefficients of the space-time predictor by evaluating the polynomials at the temporal Gauss-Legendre nodes

$$\hat{\mathbf{q}}_{qr} = \hat{\mathbf{u}}_q(\tau_r) \ . \tag{6.9}$$

Since we do not need the full polynomial information at all points in the interval, we can do this already in (6.8) by precomputing the constant $s^* \times (M+1)$ matrix $\mathbf{B}_{ir}^{\vartheta} = b_i(\tau_r)$ and perform the matrix multiplication

$$\hat{\mathbf{q}}_{qr} = \hat{\mathbf{u}}_q(0) + \mathbf{B}_{ir}^{\vartheta} \ \mathbf{k}_q^{(i)} \ . \tag{6.10}$$

Finally, we want to give some remarks on the stability of this method for the linear case ($\mathbf{F}(\mathbf{U}) = a\mathbf{U}$) without a source term. Following the general stability analysis for Runge-Kutta schemes in [Gustafsson et al., 1995] we find that due to the properties of the derivative matrix this procedure is unconditionally (for any CFL number) unstable. This was expected since it is completely local without taking any influence of neighbouring cells into account. Stability will be obtained in the corrector step (3.11) or (3.23) on the topgrid or the subgrid, respectively.

Part III

Results

Chapter 7

Numerical Tests

After introducing the algorithm in Part I and in particular the important local space-time predictor in Part II we will now perform several numerical tests to show three things: Firstly, we prove the correct implementation of the methods by carrying out standard convergence tests (see [Harten et al., 1987; Qiu and Shu, 2005; Bugner et al., 2015]) in Section 7.1. This includes the scalar nonlinear Burgers equation and the Euler equations for an ideal gas as an example for a nonlinear system of hyperbolic conservation laws.

Secondly, we test the performance of the troubled cell indicator in Section 7.2 by evolving the same initial data as used in the Burgers convergence test but let it run until a discontinuity arises. We contrast the results with the numerical admissibility detection criterion given in [Dumbser et al., 2014]. Additionally, we reproduce all the tests of [Montecinos, 2015], showing that the algorithm can deal with (soft) nonlinear source terms. This will also provide evidence of a well functioning troubled cell indication.

Thirdly, the efficiencies of the two predictor methods are compared. We will discuss the dependency on the polynomial degree M and the number of cells N on the topgrid for the linear and nonlinear case.

The complete source code for all the tests can be found on the CD attached to the thesis.

7.1 Convergence Tests

In this section we will perform two standard convergence tests as done for example in [Bugner et al., 2015; Harten et al., 1987; Qiu and Shu, 2005]. For both of them we know the analytic solution from the methods of characteristics shown in Chapter 2 and are thus able to compare our numerical results to the exact values. Throughout

the tests we will use two norms to quantify the error componentwise, namely the discrete l_1- and the discrete l_∞-norm

$$||e_j||_{l_1} = ||U_j - u_{h,j}||_{l_1} = \frac{1}{N}\sum_{i=1}^{N} \left|U_j(x_i) - u_{h,j}(x_i)\right| \ , \qquad j = 1, \dots, \nu \tag{7.1a}$$

$$||e_j||_{l_\infty} = ||U_j - u_{h,j}||_{l_\infty} = \max_{i\in[1,\dots,N]} \left(\left|U_j(x_i) - u_{h,j}(x_i)\right|\right) \ , \qquad j = 1, \dots, \nu \ . \tag{7.1b}$$

By comparing the error norms for different numbers of grid cells (say N_1 and N_2) we can calculate the order of convergence O for this resolution:

$$O = \ln\left(\frac{||e_j^{(N_1)}||}{||e_j^{(N_2)}||}\right) \Big/ \ln\left(\frac{N_2}{N_1}\right) \ , \qquad j = 1, \dots, \nu \ . \tag{7.2}$$

Later, we will show convergence tables where the number of cells in consecutive rows is increased by a factor of two. The order O_{l_1} (O_{l_∞}) in this case means that the l_1-(l_∞)-error was reduced by a factor of

$$\frac{||e_j^{(N_1)}||}{||e_j^{(N_2)}||} = 2^O \ , \qquad j = 1, \dots, \nu \ . \tag{7.3}$$

According to [Dumbser et al., 2013; Dumbser et al., 2014] we expect the order of convergence to depend on the degree M of the polynomial representation of the data, namely $O = M + 1$.

7.1.1 Burgers Equation

We solve the scalar nonlinear Burgers equation:

$$\frac{\partial U}{\partial t} + \frac{1}{2}\frac{\partial U^2}{\partial x} = 0 \ , \tag{7.4}$$

with the initial condition $U(x, 0) = 0.5 + \sin(\pi x)$. As seen in Example 2.6 we can obtain an analytic solution at $t = 0.5/\pi$ by iteratively solving the characteristic relation (see (2.13))

$$U = 0.5 + \sin\left(\pi[x - Ut]\right) \ , \tag{7.5}$$

with the initial guess $U = U(x, 0)$. This exact test was also done in [Qiu and Shu, 2005] for polynomials of degree $M = 1, 2, 3$ and in [Bugner et al., 2015] for $M = 1, 3, 5$, which gives us the chance to compare with our results shown in Table 7.1. We present the error in the two norms (7.1) and the respective convergence orders for the full algorithm (see Figure 4.1) applying the two different predictor methods discussed in Part II denoted by DG-ADER$_M$ (see Chapter 5) and DG-RK$_M$ (see Chapter 6). Comparing the results for $M = 2, 3, 5$ we see that the error

Table 7.1: Convergence tests for the Burgers equation $U_t + 0.5(U^2)_x = 0$ using different DG-ADER$_M$ and DG-RK$_M$ schemes. Initial condition set to $U(x,0) = 0.5 + \sin(\pi x)$. $C_{FV} = 0.5$, $t = 0.5/\pi$ and $\kappa = 10$. Topgrid consists of N_{top} uniform cells on the interval $I = [0, 2]$ with periodic boundary conditions.

		DG-ADER$_M$				DG-RK$_M$				
		error		order		error		order		
	N_{top}	l_1	l_∞	l_1	l_∞	l_1	l_∞	l_1	l_∞	Theory
M = 2	20	3.74e-04	3.82e-03			3.74e-04	3.82e-03			3
	40	4.69e-05	4.38e-04	3.00	3.12	4.69e-05	4.38e-04	3.00	3.12	
	80	5.60e-06	5.84e-05	3.07	2.91	5.60e-06	5.84e-05	3.07	2.91	
	160	6.13e-07	7.44e-06	3.19	2.97	6.13e-07	7.44e-06	3.19	2.97	
	320	7.25e-08	1.78e-06	3.08	2.06	7.24e-08	1.78e-06	3.08	2.06	
M = 3	20	3.67e-05	1.07e-03			3.71e-05	1.07e-03			4
	40	2.97e-06	5.40e-05	3.63	4.31	2.99e-06	5.40e-05	3.63	4.31	
	80	2.09e-07	6.27e-06	3.83	3.11	2.09e-07	6.27e-06	3.84	3.11	
	160	1.43e-08	3.10e-07	3.87	4.34	1.43e-08	3.10e-07	3.87	4.34	
	320	9.76e-10	4.11e-08	3.87	2.92	9.77e-10	4.11e-08	3.87	2.92	
M = 4	20	1.19e-05	1.82e-04			1.19e-05	1.82e-04			5
	40	3.22e-07	2.21e-06	5.21	6.36	3.22e-07	2.21e-06	5.21	6.36	
	80	9.81e-09	8.41e-08	5.04	4.72	9.81e-09	8.41e-08	5.04	4.72	
	160	2.73e-10	2.67e-09	5.17	4.98	2.73e-10	2.67e-09	5.17	4.98	
	320	7.05e-12	9.43e-11	5.28	4.82	7.05e-12	9.43e-11	5.28	4.82	
M = 5	20	5.41e-07	8.32e-06			5.39e-07	8.32e-06			6
	40	9.03e-09	2.50e-07	5.90	5.06	9.03e-09	2.50e-07	5.90	5.06	
	80	1.67e-10	1.18e-08	5.76	4.41	1.67e-10	1.18e-08	5.76	4.41	
	160	3.21e-12	3.82e-10	5.70	4.95	3.21e-12	3.82e-10	5.70	4.95	
	320	6.36e-14	1.24e-11	5.66	4.95	6.37e-14	1.24e-11	5.66	4.95	
M = 6	20	3.01e-07	5.07e-06							7
	40	2.33e-09	5.45e-08	7.01	6.54					
	80	1.87e-11	2.85e-10	6.96	7.58					
	160	1.52e-13	3.06e-12	6.94	6.54					
	320	1.56e-15	2.00e-14	6.61	7.26					
M = 7	20	1.11e-07	1.98e-06							8
	40	2.04e-10	2.88e-09	9.09	9.43					
	80	7.44e-13	1.91e-11	8.10	7.24					
	160	4.95e-15	2.29e-13	7.23	6.38					
	320	1.07e-15	3.51e-14	2.21	2.71					

in both norms is compatible with the one shown in [Bugner et al., 2015; Qiu and Shu, 2005]. To be more specific, we obtain slightly smaller error norms for $M = 2$ compared to [Qiu and Shu, 2005], whereas the error is marginally larger for $M = 3$. In comparison with [Bugner et al., 2015] for $M = 3, 5$ our results differ by less than a factor of two to our advantage. In Figure 7.1 we present the pointwise (rescaled) errors. For each of the subfigures the maximum error is obtained in the vicinity of the region where the discontinuity is going to arise. The deviation from the analytic solution is especially small in the interval $[0, 0.3]$, which might be explained by the fact that due to the structure of the equation the solution does not change much

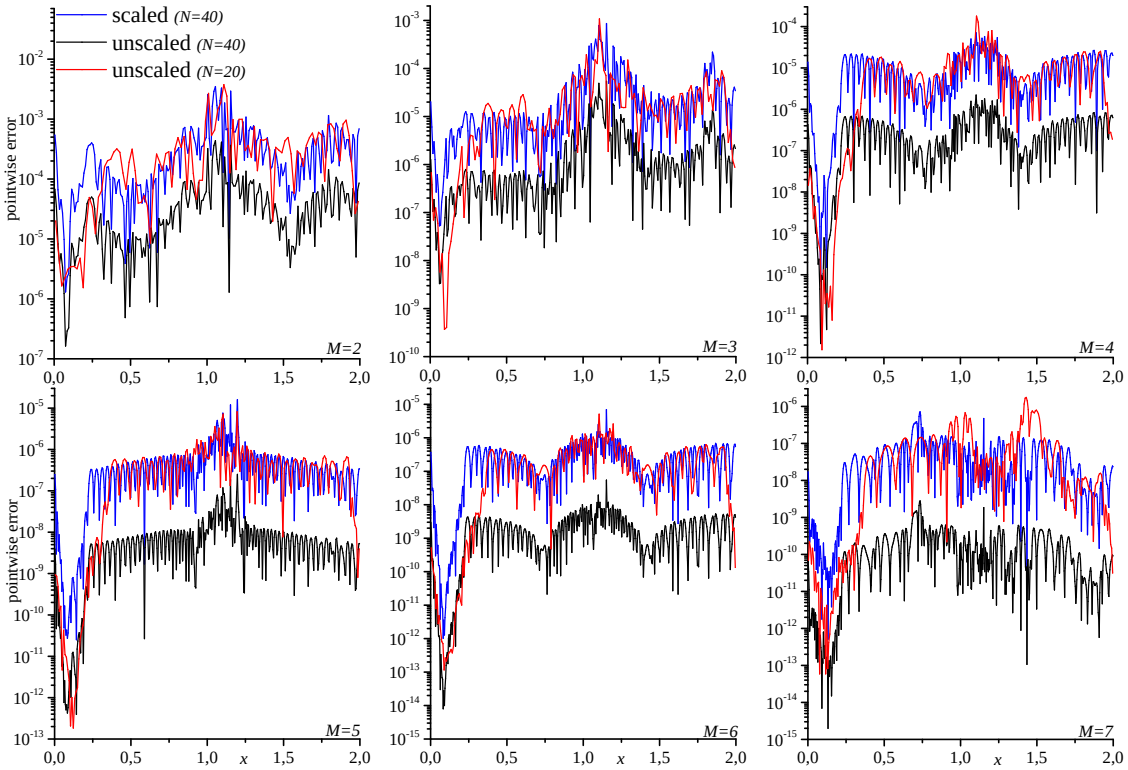


Figure 7.1: Pointwise convergence test for the Burgers equation using the DG-ADER$_M$ method (see Table 7.1). Each subfigure shows the absolut pointwise error of the subcell average solution for $N_{top} = 20$ topcells (red) and $N_{top} = 40$ topcells (black) together with the pointwise rescaled (factor 2^{M+1}) error for $N_{top} = 40$ topcells (blue) over the whole domain $[0, 2]$. Note the different scales on the vertical axes.

compared to the initial data in this region.

The troubled cell parameter $\kappa = 10$ leads to an average number of troubled cells detected of less than one percent. This did not introduce any oscillations because the solution is smooth during the whole run. We set the CFL number to the highest value applicable for $M = 7$, guaranteeing a stable simulation. Tests have shown that for smaller values of M the CFL number can be chosen larger, i.e. up to $C_{FV} = 0.93$ for $M = 2$.

The comparison between the two predictor methods reveals that there is no significant difference between the schemes. The error values are even equal at the given precision for most of the configurations. The theoretically expected order of convergence for all setups is verified by the global error (see Table 7.1) as well as the pointwise absolute error (see Figure 7.1). We note that for even degrees of the reconstructing polynomial the convergence seems to be slightly higher than expected, whereas it is the other way round for odd degrees.

7.1.2 Euler Equations

Table 7.2: Convergence tests for the Euler equations using different DG-ADER$_M$ and DG-RK$_M$ schemes. Initial condition set to $\rho(x,0) = 1 + 0.2\sin(\pi x)$, $u(x,0) = 0.7$, $p(x,0) = 1.0$. $C_{FV} = 0.3$, $t = 2/0.7$ and $\kappa = 500$. Topgrid consists of N_{top} uniform cells on the interval $I = [0,2]$ with periodic boundary conditions.

		DG-ADER$_M$				DG-RK$_M$				
		error		order		error		order		
	N_{top}	l_1	l_∞	l_1	l_∞	l_1	l_∞	l_1	l_∞	Theory
M = 2	20	5.42e-05	1.47e-04			5.42e-05	1.47e-04			3
	40	6.17e-06	1.99e-05	3.13	2.88	6.17e-06	1.99e-05	3.13	2.88	
	80	7.43e-07	2.61e-06	3.05	2.93	7.43e-07	2.61e-06	3.05	2.93	
	160	9.31e-08	3.29e-07	3.00	2.99	9.31e-08	3.29e-07	3.00	2.99	
	320	1.15e-08	4.21e-08	3.02	2.97	1.15e-08	4.21e-08	3.02	2.97	
M = 3	20	9.49e-07	3.28e-06			9.49e-07	3.28e-06			4
	40	3.37e-08	9.27e-08	4.82	5.14	3.37e-08	9.27e-08	4.82	5.14	
	80	1.80e-09	5.01e-09	4.23	4.21	1.80e-09	5.01e-09	4.23	4.21	
	160	1.08e-10	3.12e-10	4.06	4.01	1.08e-10	3.12e-10	4.06	4.01	
	320	6.68e-12	1.94e-11	4.02	4.01	6.68e-12	1.94e-11	4.02	4.01	
M = 4	20	6.93e-07	3.18e-06			6.93e-07	3.18e-06			5
	40	1.01e-08	1.02e-07	6.10	4.96	1.01e-08	1.02e-07	6.10	4.96	
	80	1.64e-10	2.84e-09	5.94	5.17	1.64e-10	2.84e-09	5.94	5.17	
	160	2.64e-12	7.97e-11	5.96	5.16	2.64e-12	7.71e-11	5.96	5.20	
	320	4.75e-14	2.53e-12	5.80	4.98	4.73e-14	2.47e-12	5.80	4.96	

Consider the nonlinear classical Euler equations

$$\frac{\partial}{\partial t}\begin{pmatrix}\rho\\ \rho u\\ E\end{pmatrix} + \frac{\partial}{\partial x}\begin{pmatrix}\rho u\\ \rho u^2 + p\\ u(E+p)\end{pmatrix} = \mathbf{0}\ , \tag{7.6}$$

with the density ρ, the pressure p, the particle velocity u and the energy density E

$$E = \rho\left(\frac{1}{2}u^2 + e\right)\ , \tag{7.7}$$

where e is the specific internal energy. Assuming an ideal gas we use the equation of state

$$e = e(\rho, p) = \frac{p}{(\gamma-1)\rho}\ . \tag{7.8}$$

The ratio of the specific heats $\gamma = c_p/c_v$ is arbitrarily set to $\gamma = 1.4$ for all tests. The initial condition on the interval $I = [0,2]$ is set to

$$\rho_0(x) = 1 + 0.2\sin(\pi x)\ , \tag{7.9a}$$

$$u_0(x) = 0.7\ , \tag{7.9b}$$

$$p_0(x) = 1.0\ . \tag{7.9c}$$

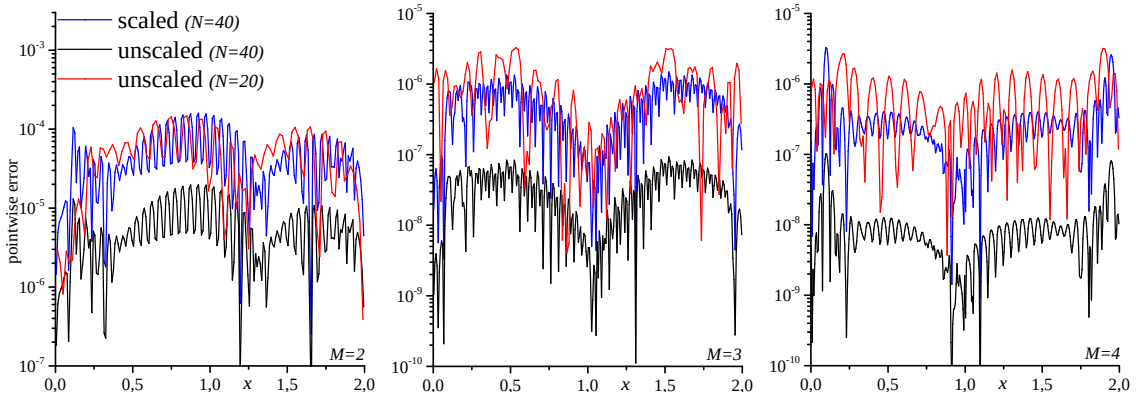


Figure 7.2: Pointwise convergence test for the Euler equations using the DG-ADER$_M$ method (see Table 7.2). Each subfigure shows the absolut pointwise error of the subcell average solution for $N_{top} = 20$ topcells (red) and $N_{top} = 40$ topcells (black) together with the pointwise rescaled (factor 2^{M+1}) error for $N_{top} = 40$ topcells (blue) over the whole domain $[0, 2]$. Note the different scales on the vertical axes.

This test is referred to as the simple wave test and has the analytic solution $\rho(x,t) = 1 + 0.2\sin(\pi[x - ut])$, $u = 0.7$, $p = 1$. We implement periodic boundary conditions computing the solution up to $t = 2/0.7$ where it is supposed to coincide exactly with its initial values. The results for the density are shown in Table 7.2.

Again, the different predictor methods produce almost the same outcome. The order of convergence in l_∞-norm matches the theoretical value very well. The same holds for the l_1-norm, except for $M = 4$ where a slightly higher order is achieved. This is also visible in Figure 7.2 (right panel) where the blue line (scaled by a factor of $2^{4+1} = 32$) is below the red line.

7.2 Tests with Shocks and Source Terms

Whereas we presented quantitative evidence in form of explicit convergence tests in the previous section, here we want to perform several qualitative test runs hinting towards the correct implementation of the algorithm (see Figure 4.1). Note that every test scenario applies the novel Runge-Kutta predictor scheme. In all of the simulations we either know the analytic solution from the method of characteristics (see Chapter 2) or compute a reference solution on a very fine grid.

7.2.1 Troubled Cell Indicator

In order to examine the performance of the troubled cell indicator (see Section 3.2.2), we are going to perfom a test similar to that shown in Section 7.1.1, knowing that

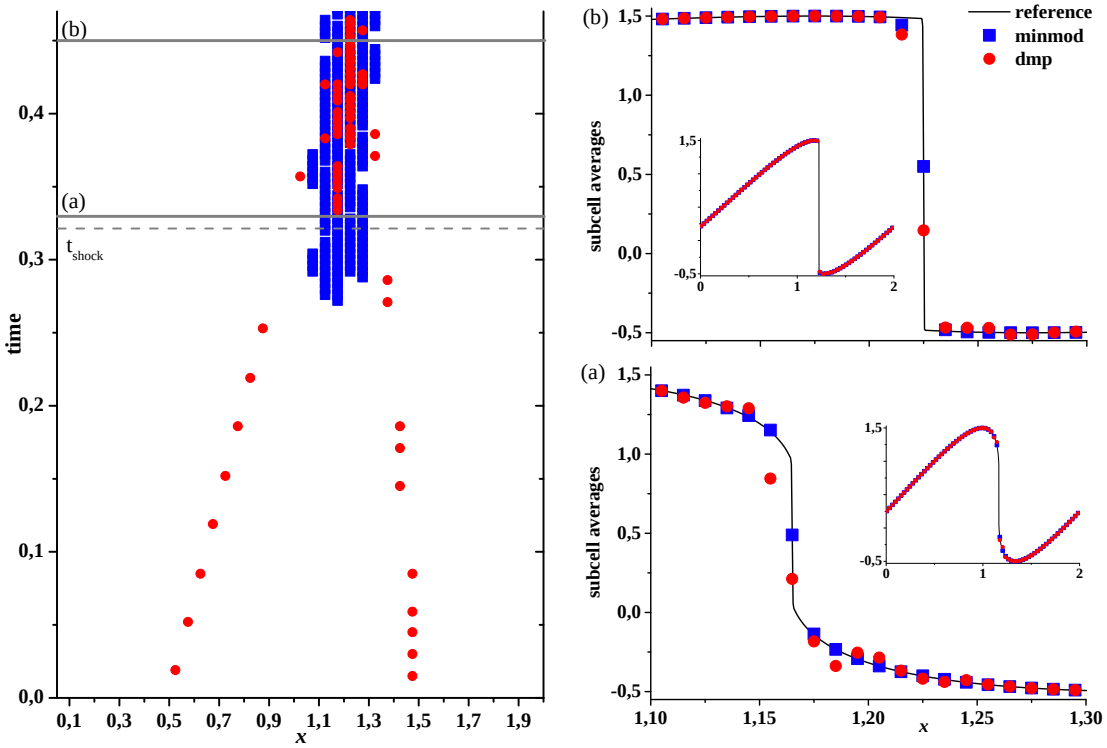


Figure 7.3: Comparing the results of two runs with different troubled cell indicators, based on the minmod-function as in [Qiu and Shu, 2005] (blue) and on the discrete maximum principle (dmp) as in [Dumbser et al., 2014] (red). Dots in the left plot show which cells were marked problematic during the evolution of the Burgers equation with initial data $U_0 = 0.5 + \sin(\pi x)$ on the interval $[0, 2]$. On the right hand side subcell average data at two time levels (a) $t = 0.33$ and (b) $t = 0.45$ are plotted, focussing on the domain where the shock occurs and giving a smaller overview of the whole solution respectively ($N_{top} = 40$, $M = 3$, $C_{FV} = 0.75$, $\kappa = 10$). Reference solution stems from a DG-RK$_2$ run on one thousand topcells.

the Burgers equation (7.4) can develop a discontinuity from smooth initial data U_0 at

$$t_{shock} = -\left(\min \frac{\partial U_0}{\partial x}\right)^{-1} . \tag{7.10}$$

Applying this formula to the initial data $U_0 = U(x, 0) = 0.5 + \sin(\pi x)$, we used for the convergence test in the previous section, we find that a shock occurs at $t_{shock} = 1/\pi \approx 0.32$. In order to show that the code can deal with this discontinuity and to compare the troubled cell indicators proposed by [Dumbser et al., 2014] with [Qiu and Shu, 2005] we evolve the same initial data up to $t = 0.45$. The results are shown in Figure 7.3, where we plot the average cell values of the numerical solution $v(x, t)$ on the subgrid at times $t = 0.33$ and $t = 0.45$ next to the cells $\tilde{T}_i$ which were identified as troubled. The reference solution is obtained from a run

with one thousand topcells because the simple Newton-Raphson iteration for the characteristic relation (7.5) fails at the presence of a discontinuity.

It becomes clear that the indicator based on the discrete maximum principle (dmp) as proposed by [Dumbser et al., 2014] marks the smooth extrema as troubled, whereas the one taken from [Qiu and Shu, 2005] is only active in the vicinity of the shock as it should be. In principle, this might not be problematic because it could still lead to a non-oscillatory valid solution. What is troublesome is that the dmp fails to detect the formation of the shock early enough. We see this in the bottom right panel of Figure 7.3. At time (a) $t = 0.33 > t_{shock}$ the subcell average data plot oscillates noticeably because the cells around $x_{shock} \approx 1.16$ were not indicated as troubled. Both solutions at (b) $t = 0.45$ look rather smooth due to the smoothing effect of the WENO reconstruction applied on the subcells.

A possible way to estimate κ is given in [Qiu and Shu, 2005] where they state that it is proportional to the second derivative of the initial condition at smooth extrema. In our particular example that is $\kappa' = \pi^2$. This should be regarded as an order of magnitude estimation where further fine tuning is needed. Especially for very coarse grids a smaller value is a safer choice.

As a result of this test we use the numerical admissibility criterion of [Qiu and Shu, 2005] in all runs with the freedom to choose κ. The additional pilot runs to find a suitable value for κ seem to be an acceptable price to pay to guarantee a stable simulation.

7.2.2 Burgers Equation with Source

Next, we assess the performance of the algorithm for the Burgers equation with different source terms, running the setups given in [Montecinos, 2015] of the form

$$\frac{\partial U}{\partial t} + \frac{1}{2}\frac{\partial U^2}{\partial x} = S(U(x,t)) \ . \tag{7.11}$$

There are three cases, namely linear, quadratic and exponential source functions. Since all of them will include steep gradients or discontinuities, this also provides more examples of a well functioning troubled cell indicator. Plots of the time evolutions of all cases are shown in Figure 7.7.

Firstly, we consider a linear source term $S = -2U$. The solutions at $t = 0.2$ for the initial condition

$$U_0(x) = \begin{cases} 2 \ , & \text{if} \quad x \leq 0.1 \\ 2 + \frac{x-0.1}{0.1} \ , & \text{if} \quad 0.1 \leq x \leq 0.2 \\ 3 \ , & \text{if} \quad 0.2 \leq x \leq 0.4 \\ 3 - \frac{x-0.4}{0.2} \ , & \text{if} \quad 0.4 \leq x \leq 0.6 \\ 2 \ , & \text{if} \quad x \geq 0.6 \end{cases} . \tag{7.12}$$

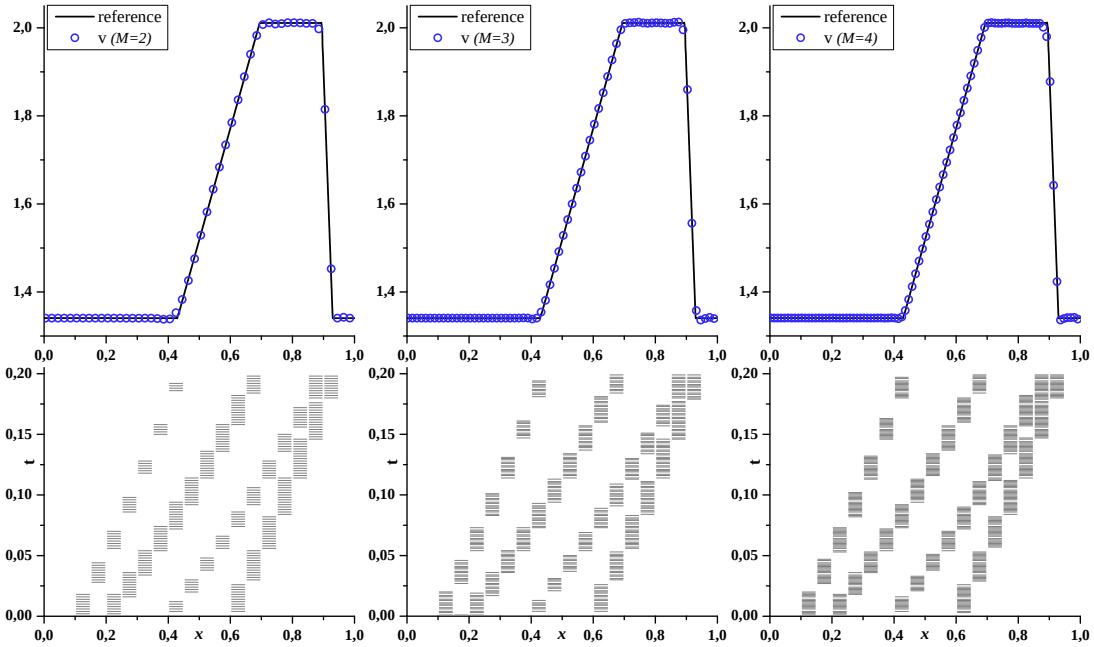


Figure 7.4: Burgers equation with linear source term. Initial condition is given by (7.12). The subcell average results at $t = 0.2$ are shown for different degrees M of the polynomial approximation together with a reference solution. Below each plot a time history of which cells were denoted as troubled is depicted using the troubled cell indicator (3.22) with $\kappa = 15$ ($N_{top} = 20$, $C_{FV} = 0.6$).

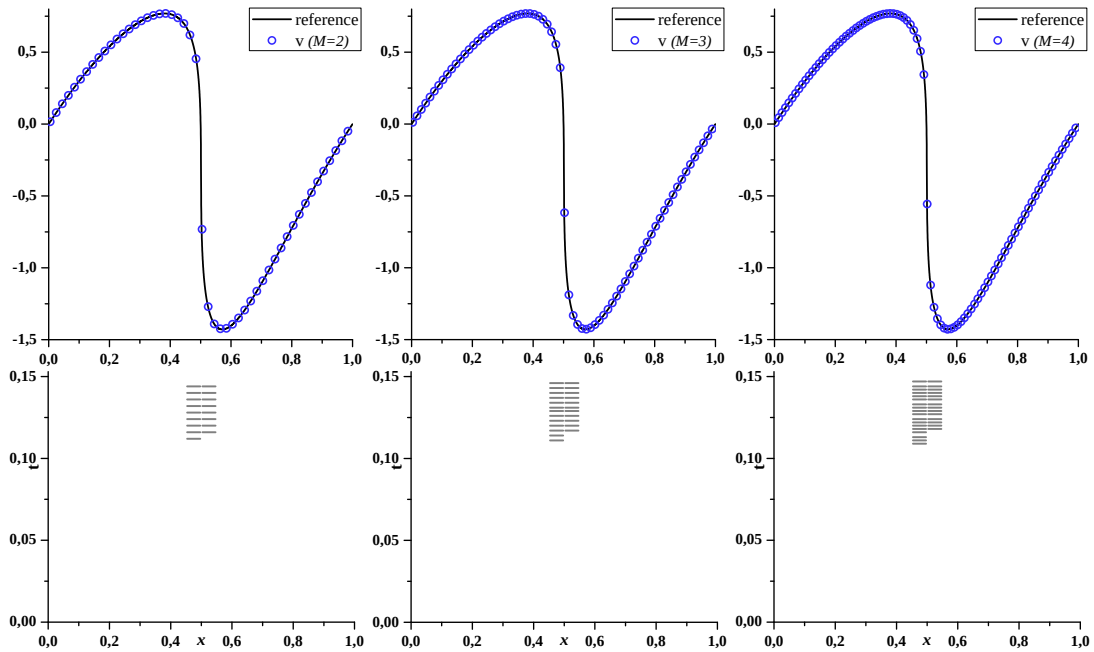


Figure 7.5: Burgers equation with quadratic source term. Initial condition is given by (7.13). The subcell average results at $t = 0.15$ are shown for different degrees M of the polynomial approximation together with a reference solution. Below each plot a time history of which cells were denoted as troubled is depicted using the troubled cell indicator (3.22) with $\kappa = 100$ ($N_{top} = 20$, $C_{FV} = 0.6$).

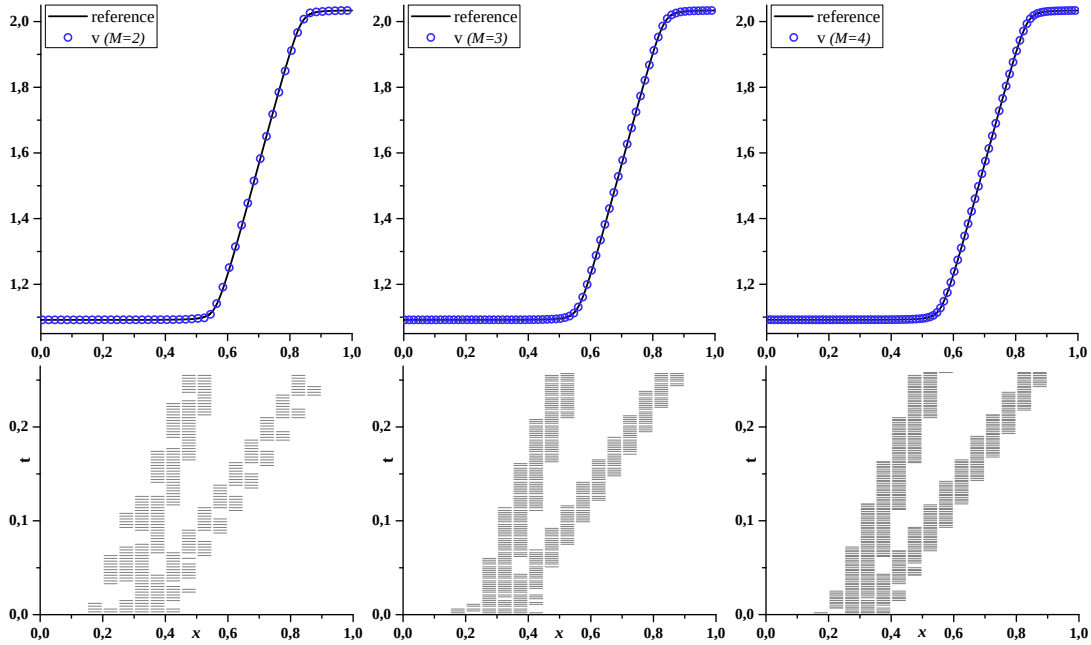


Figure 7.6: Burgers equation with exponential source term. Initial condition is given by (7.14). The subcell average results at $t = 0.26$ are shown for different degrees M of the polynomial approximation together with a reference solution. Below each plot a time history of which cells were denoted as troubled is depicted using the troubled cell indicator (3.22) with $\kappa = 1$ ($N_{top} = 20$, $C_{FV} = 0.6$).

are shown in Figure 7.4. In these three examples we changed the polynomial degree M, but fixed the other parameters ($N_{top} = 20$, $C_{FV} = 0.6$ and $\kappa = 15$). Note that since we plot the $(2M + 1)N_{top}$ subcell average data points, the number of points increases from left to right, although the number of topcells is fixed. All the results are very close to the reference solution, using the DG-RK$_2$ method on one thousand topcells. We observe that the troubled cell indicator tracks the transitions between the different domains, which was expected since the derivatives exhibit jumps at these points.

Secondly, the source term is set to be $S = -2U^2$. Similar to the linear case we keep the number of cells N_{top}, the CFL number C_{FV} and κ fixed while changing M. The initial condition is determined as in Example 4 of [Montecinos, 2015]

$$U_0(x) = \sin(2\pi x) \ . \tag{7.13}$$

This test is similar to the one shown in the previous section. We plot the subcell average solution in Figure 7.5 at $t = 0.2$, right before the development of the discontinuity. The choice $\kappa = 100$ leads to an early detection of the arising discontinuity but does not mark the extrema in any case. Although only twenty topcells were

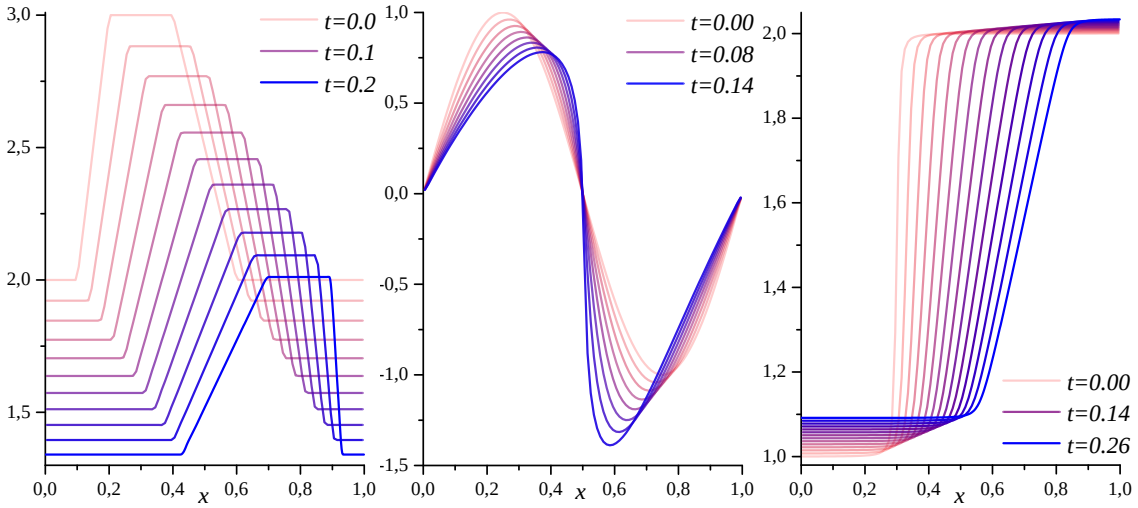


Figure 7.7: Time evolution of the Burgers equation (7.11) with different source terms. From left to right: linear, quadratic and exponential with initial data and parameters as in Figure 7.4, Figure 7.5 and Figure 7.6, respectively. Time difference between two lines is $t = 0.02$ in all panels ($M = 2$).

used, all the results seem to match the reference solution perfectly, probably due to the smoothness of the solution.

Lastly, we reproduce Example 5 of [Montecinos, 2015], applying an exponential source term $S = \exp(-U)$. The initial condition is given by

$$U_0(x) = 2\left(\frac{1 - w(x)}{2}\right) + \left(\frac{1 + w(x)}{2}\right) , \tag{7.14}$$

with

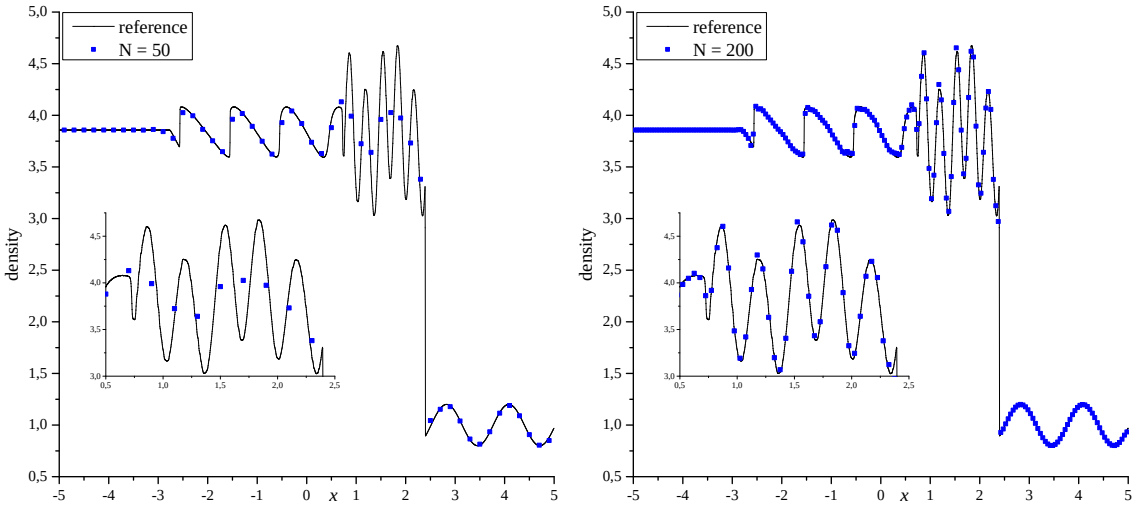
$$w(x) = \frac{0.3 - x}{\sqrt{(0.3 - x)^2 + 10^{-4}}} . \tag{7.15}$$

Here we proceed in the same way with the parameters as before. The solution at $t = 0.26$ is shown in Figure 7.6. As in the test with the linear source term (see Figure 7.4) cells in the vicinity of the kinks are detected to be troubled. Again, the smoothness of the solution results in very good agreement of our subcell data with the reference solution.

All in all, these three test cases show that the algorithm is able to deal with (soft) nonlinear source terms. Additionally, we were able to approve the troubled cell indicator once more.

7.2.3 Euler Equations

The last test contains both shocks and complex smooth region structures. In [Dumbser et al., 2014] they refer to it as Shu-Osher oscillatory shock tube. Initially we



(a) **(b)**

Figure 7.8: Shu-Osher oscillatory shock tube. $C_{FV} = 0.2$, $t = 1.8$, $\kappa = 300$, $M = 2$. Numerical results of the density for two different topcell resolutions (a) $N_{top} = 50$ and (b) $N_{top} = 200$ is plotted together with reference solution which stems from a simulation with the same parameters but on one thousand topcells.

have a Mach 3 shock front located at $x = -4$ on the computational domain $[-5, 5]$, i.e.

$$(\rho, u, p) = \begin{cases} (3.857143,\ 2.629369,\ 10.333333)\ , & \text{if } x < -4 \\ (1 + 0.2\sin(5x),\ 0,\ 1)\ , & \text{if } x \geq -4 \end{cases} . \tag{7.16}$$

In Figure 7.8 we plotted the density at time $t = 1.8$ for two different numbers of topcells. The reference solution was obtained by a third order DG-RK$_2$ scheme on one thousand cells.

In [Qiu and Shu, 2005] (Figure 13) the very same test is perfomed using a different DG method with WENO limiting. In comparison, our results are almost identical, giving more reasons to believe in the correct implementation of the DG-RK method.

7.3 Predictor Tests

In this section we want to compare the two predictor methods shown in Part II. To do so, we introduce the number of flux evaluations $\#\mathbf{F}$ as a measure of their efficiency. Even though the predictor schemes consist of other parts, this is a reasonable approach because already the calculation of the simple fluxes in our tests make up the major portion of the computational costs. We will see that, considering a certain overhead, the run time is closely related to said number.

For each of the predictor schemes we will determine the number of flux evaluations and denote them by $\#\mathbf{F}_{iter}$ and $\#\mathbf{F}_{s^*}$ for the iteration scheme and the local Runge-Kutta predictor, respectively. Together with the evaluations of the corrector step (3.11) or (3.23), which is the same for both algorithms, we will obtain a value which allows for a meaningful comparison. Note that all these estimates are with regard to one cell.

In Section 5.2 we proved that for any initial guess the iteration procedure, to obtain the space-time predictor, takes at most $M+1$ steps. Associated with one iteration step is the update of all space-time coefficients $\hat{\mathbf{q}}_{ps}$, each requiring one flux evaluation. This means for polynomials of degree M the iteration scheme computes

$$\#\mathbf{F}_{iter} = \underbrace{(M+1)}_{spatial} \cdot \underbrace{(M+1)}_{temporal} \cdot \underbrace{iter}_{\#iterations} \tag{7.17}$$

fluxes due to the $M+1$ collocation points in each direction. In short, $(M+1)^3$ flux evaluations are necessary in the worst case.

In contrast to the predictor scheme proposed by [Dumbser et al., 2014] it is possible to calculate an exact constant number of flux evaluations applying the local Runge-Kutta predictor for a given order of the scheme. For every spatial collocation point we have to compute s^* right hand sides including one flux function each (see (6.7)), hence

$$\#\mathbf{F}_{s^*} = s^* \cdot (M+1) \ . \tag{7.18}$$

The number of flux evaluations for the corrector steps is given by

$$\#\mathbf{F}_{cor} = (M+1)^2 \ . \tag{7.19}$$

In the following we want to compare the total number of flux evaluations needed in the DG-ADER$_M$- and in the DG-RK$_M$-algorithm for different values of M. The quotients

$$\frac{\#\mathbf{F}_{ADER}}{\#\mathbf{F}_{RK}} = \frac{\#\mathbf{F}_{iter} + \#\mathbf{F}_{cor}}{\#\mathbf{F}_{s^*} + \#\mathbf{F}_{cor}} \tag{7.20}$$

for the convergence test done in Section 7.1.1 and a convergence test for the advection equation are shown in Figure 7.9. The local Runge-Kutta schemes for $M = 6, 7$ were not actually implemented but [Hairer et al., 1993] showed that there are dense output formulas with $s^* = 13, 16$ stages, respectively.

Each point in Figure 7.9 represents the mean value of the quotient over the different resolutions used in the convergence tests, i.e. $N = 20, 40, 80, 160, 320$. We observed that with increasing resolution the number of iteration steps in the DG-ADER$_M$ method decreases monotonously. This is illustrated by the bars reaching from $N = 20$ (top) to $N = 320$ (bottom). Tests with an unnecessarily small CFL

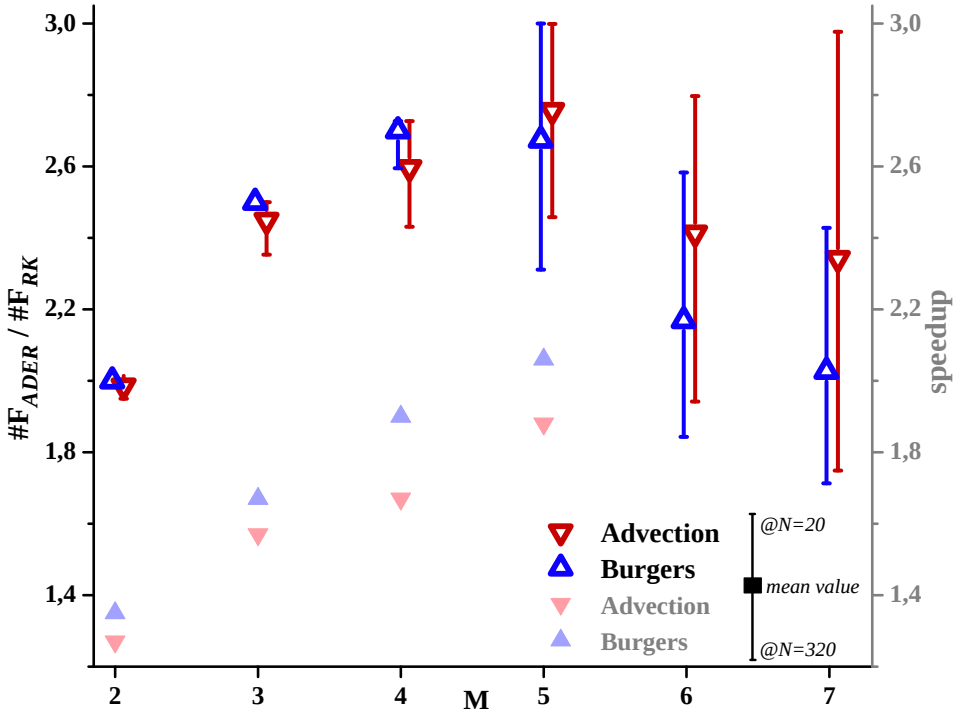


Figure 7.9: Quotient $\#\mathbf{F}_{ADER}/\#\mathbf{F}_{RK}$ for different degrees M of polynomial data representation. We compare the results for the advection (red) and Burgers equation (blue). The markers show the mean value over all the resolutions in the according convergence test. The bar represents the value at $N = 20$ and $N = 320$ at its top end and bottom end, respectively. Pale data points refer to the actual speedup of the predictor-corrector routines achieved for the convergence tests with $N = 320$ topcells.

number $C_{FV} = 0.01$ (see Appendix C.2) show that there seems to be a minimum number of iterations, namely $iter_{min} = 3$.

We want to put emphasis on the fact that for all different setups in Figure 7.9 the quotient is larger than 1.7, meaning the algorithm applying the local Runge-Kutta predictor method requires fewer flux evaluations than the corresponding scheme with the iterative predictor. This directly translates to a speedup depicted by the pale markers in the same figure. We measured the computational time of the predictor and corrector routines with the `timer`-functions from the `bamps` code [Hilditch et al., 2015] for each order at $N = 320$. Since these routines include more calculations than only the flux evaluations the actual speedup is smaller than the quotient of the number of flux evaluations.

Keeping in mind that the fluxes for the advection as well as the Burgers equation are fairly simple, it is safe to state that for a more complicated hyperbolic conservation laws the achieved speedup would be even closer to the quotient (7.20).

Nevertheless, for $M = 5$ it is about twice as fast to use the local Runge-Kutta

predictor method compared to the iteration scheme. With respect to the total run time, the time used to compute the predictor solution was reduced from $\approx 60\%$ to $\approx 40\%$ in this case.

The results of Figure 7.9 are also valid for higher dimensional setups under the assumption that the iteration scheme converges as fast as in the 1D examples. This is due to the fact that both numbers of flux evaluations, $\#\mathbf{F}_{ADER}$ and $\#\mathbf{F}_{RK}$, scale with the same factor.

Chapter 8

Conclusion

Summary

[Dumbser et al., 2014] presented a very promising a posteriori finite volume subcell limiter technique for the ADER-DG method to solve nonlinear systems of hyperbolic conservation laws in multiple spatial dimensions with arbitrary high order of accuracy in space and time. To perfom one time step we have to compute a space-time predictor locally within each cell, which is based on an iteration scheme. This weak solution, neglecting the influence of the neighbouring cells, is then used in the DG method to obtain a high order approximation for the numerical flux function. The task of this thesis was to evaluate the efficiency of the predictor-corrector approach by [Dumbser et al., 2014]. Based on the matrices used during the predictor step we were able to show (for the linear case) that the iteration converges to the exact solution in a finite number of steps depending on the overall order of the scheme.

We implemented the full algorithm for one spatial dimension with two alterations: (1) modified troubled cell indication and (2) added possibility to simulate equations with non-zero source terms. The correctness of the implementation was shown in several standard numerical convergence tests for scalar and vectorial nonlinear equations (see Chapter 7 and Appendix C). A comparison between the original numerical admissibility criterion proposed in [Dumbser et al., 2014] and the one from [Qiu and Shu, 2005], which introduces a parameter κ, showed that the latter one does a better job detecting problematic domains, given a suitable value for κ. This means the simulation runs more stable in the presence of discontinuities. Furthermore, a well chosen parameter also reduces the number of cells that are unnecessarily marked as problematic, i.e. local extrema, thus increasing the efficiency of the code. Several tests with nonlinear source terms strenghened this statement.

It turned out that the iteration scheme to obtain the element local space-time predictor on both the top- and subgrid makes up a significant portion of the com-

putational costs for one time step. In order to increase the efficiency of the whole scheme we tried to improve the computation of the predictor solution. We developed a new alternative way to get a high order space-time weak solution based on a explicit local Runge-Kutta method. Within a cell a Runge-Kutta step is done at each collocation point, calculating the right hand side flux derivative analytically due to the polynomial representation. Together with an appropriate dense output formula we can obtain the coefficients which uniquely define the space-time polynomial. Convergence tests with the alternative predictor step lead to almost identical ($< 1\%$ difference) error values.

Tests for linear and nonlinear scalar hyperbolic conservation laws demonstrated that our new approach is more efficient than the ADER iteration scheme in the sense of needing fewer flux function evaluations to obtain the same accuracy. There seems to be a maximal gain of efficiency for polynomial degrees of four and/or five, which is directly related to the observation that for higher degrees the number of iterations is significantly smaller than the theoretically predicted upper limit.

Future Work

After succesfully testing the original as well as the newly developed algorithm on different test equations in one spatial dimension, the next consequent steps are: (1) to extend the implementation to multiple spatial dimensions and (2) to include additional flux functions for more complicated setups.

The first point will be straightforward, since the formalism already exists for an arbitrary number of dimensions. A closer look reveals that the computations can be reduced to multiple one dimensional operations for which we already know the matrices involved.

What we have in mind when we talk about more complicated setups are the equations of general relativistic hydrodynamics (GRHD). Starting with the simulation of a single stationary neutron star (TOV star) in 3D as recently done in [Bugner et al., 2015], we aim to investigate a head-on collision of two such stars.

Appendix A

Explicit Computation of Matrices

A.1 DG Scheme

The explicit analytically exact expressions for the matrices used in (4.8) are

$$\begin{aligned}
\mathbf{M}_{kl} &= \int_0^1 \varphi_k(\xi)\varphi_l(\xi)d\xi \\
&= \frac{1}{2}\sum_{i=0}^{M} w_i \varphi_k(\lambda_i)\varphi_l(\lambda_i) \\
&= \frac{1}{2}\sum_{i=0}^{M} w_i \delta_{ki}\delta_{li} \\
&= \frac{1}{2} w_k \delta_{kl} \ , \qquad \text{(A.1)}
\end{aligned}$$

$$\begin{aligned}
\mathbf{G}^{ab}_{kps} &= \frac{1}{2}\frac{\Delta t}{\Delta x_{top}}\varphi_k(a)\varphi_p(b) \int_0^1 \varphi_s(\tau)d\tau \\
&= \frac{1}{4}\frac{\Delta t}{\Delta x_{top}}\varphi_k(a)\varphi_p(b)\ w_s \ , \qquad \text{(A.2)}
\end{aligned}$$

where we used what we already know from (4.10), and

$$\begin{aligned}
\mathbf{L}_{kps} &= \int_0^1 \frac{\partial \varphi_k(\xi)}{\partial \xi}\varphi_p(\xi)d\xi \int_0^1 \varphi_s(\tau)d\tau \\
&= \frac{1}{4}\Delta t \sum_{i=0}^{M} w_i \left.\frac{\partial \varphi_k}{\partial \xi}\right|_{\lambda_i} \delta_{pi}\ w_s \\
&= \frac{1}{4}\Delta t\ w_p \left.\frac{\partial \varphi_k}{\partial \xi}\right|_{\lambda_p} w_s \ . \qquad \text{(A.3)}
\end{aligned}$$

A.2 FV Scheme

Similar to the DG matrices we have the following exact expressions for the matrices used in the finite volume scheme

$$\begin{aligned}\tilde{\mathbf{G}}^a_{ps} &= \frac{1}{2}\frac{\Delta t}{\Delta x_{sub}}\varphi_p(a)\ \int_0^1 \varphi_s(\tau)d\tau \\ &= \frac{1}{4}\frac{\Delta t}{\Delta x_{sub}}\ \varphi_p(a)\ w_s\ ,\end{aligned} \tag{A.4}$$

and

$$\begin{aligned}\tilde{\mathbf{H}}_{ps} &= \Delta t \int_0^1 \varphi_p(\xi)d\xi\ \int_0^1 \varphi_s(\tau)d\tau \\ &= \frac{1}{4}\Delta t\ w_p\ w_s\ .\end{aligned} \tag{A.5}$$

A.3 ADER Iteration Scheme

The matrices used in the iteration scheme (5.13) can also be computed exactly with the Gauss-Legendre quadrature formula (4.4). In fact we can reduce them to combinations of mass-, stiffness- and average-matrices similar to the corrector step matrices, hence

$$\begin{aligned}\mathbf{K}^1_{qrps} &= \int_0^1 \theta_{qr}(\xi,1)\ \theta_{ps}(\xi,1)\ d\xi - \int_0^1\int_0^1 \left(\frac{\partial}{\partial\tau}\theta_{qr}\right)\theta_{ps}\ d\xi\ d\tau \\ &= \varphi_r(1)\varphi_s(1)\mathbf{M}_{qp} - \mathbf{S}_{rs}\mathbf{M}_{qp} \\ &= \frac{1}{2}\varphi_r(1)\varphi_s(1)w_q\delta_{qp} - \frac{1}{4}w_s\ \left.\frac{\partial\varphi_r}{\partial\tau}\right|_{\lambda_s} w_q\delta_{qp}\ ,\end{aligned} \tag{A.6}$$

$$\begin{aligned}\mathbf{K}^\xi_{qrps} &= \int_0^1\int_0^1 \theta_{qr}\left(\frac{\partial}{\partial\xi}\theta_{ps}\right)d\xi\ d\tau = \mathbf{S}_{pq}\mathbf{M}_{rs} \\ &= \frac{1}{4}w_q\ \left.\frac{\partial\varphi_p}{\partial\tau}\right|_{\lambda_q} w_r\delta_{rs}\ ,\end{aligned} \tag{A.7}$$

$$\begin{aligned}\mathbf{M}_{qrps} &= \int_0^1\int_0^1 \theta_{qr}\theta_{ps}\ d\xi\ d\tau = \mathbf{M}_{qp}\mathbf{M}_{rs} \\ &= \frac{1}{4}w_q\delta_{qp}w_r\delta_{rs}\ ,\end{aligned} \tag{A.8}$$

$$\mathbf{I}^0_{qrl} = \int_0^1 \theta_{qr}(\xi,0)\ \varphi_l\ d\xi = \varphi_r(0)\mathbf{M}_{ql}$$
$$= \frac{1}{2}\varphi_r(0) w_q \delta_{ql}\ . \tag{A.9}$$

Appendix B

Runge-Kutta Formulas

B.1 Butcher Tableaus

The Butcher tableaus for order three to six we used for our numerical tests in Chapter 7 are: (see [Hairer et al., 1993; Owren and Zennaro, 1991])

– $p = 3,\ s = 3$ "Heun"

$$\begin{array}{c|ccc} 0 & & & \\ 1/3 & 1/3 & & \\ 2/3 & 0 & 2/3 & \\ \hline & 1/4 & 0 & 3/4 \end{array} \tag{B.1}$$

– $p = 4,\ s = 4$ "The Runge-Kutta"

$$\begin{array}{c|cccc} 0 & & & & \\ 1/2 & 1/2 & & & \\ 1/2 & 0 & 1/2 & & \\ 1 & 0 & 0 & 1 & \\ \hline & 1/6 & 2/6 & 2/6 & 1/6 \end{array} \tag{B.2}$$

– $p = 5,\ s = 6$ "Dormand-Prince"

$$\begin{array}{c|cccccc} 0 & & & & & & \\ 1/5 & 1/5 & & & & & \\ 3/10 & 3/40 & 9/40 & & & & \\ 4/5 & 44/45 & -56/15 & 32/9 & & & \\ 8/9 & 19372/6561 & -25360/2187 & 64448/6561 & -212/729 & & \\ 1 & 9017/3168 & -355/33 & 46732/5247 & 49/176 & -5103/18656 & \\ \hline & 35/384 & 0 & 500/1113 & 125/192 & -2187/6784 & 11/84 \end{array} \tag{B.3}$$

– $p = 6,\ s = 8$ "Owren-Zennaro"

0								
1/4	1/4							
1/4	1/8	1/8						
1/2	0	-1/2	1					
1/2	1/12	0	1/3	1/12				
3/4	0	-9/8	3/2	-3/4	9/8			
1	0	4/5	0	-3/5	0	4/5		
1	1/6	0	0	4/15	2/5	0	1/6	
	7/90	0	16/45	-4/15	2/5	16/45	5/18	-1/5

(B.4)

B.2 Dense Output

The dense output formulas for the ERK shown in appendix B.1 used for our numerical tests in Chapter 7 are: (see [Hairer et al., 1993; Owren and Zennaro, 1991])
– $p^* = 2,\ s^* = 3$ "Heun" (see (B.1))

$$b_1(\vartheta) = -\frac{3}{4}\vartheta^2 + \vartheta \tag{B.5a}$$

$$b_2(\vartheta) = 0 \tag{B.5b}$$

$$b_3(\vartheta) = \frac{3}{4}\vartheta^2 \tag{B.5c}$$

– $p^* = 3,\ s^* = 4$ "The Runge-Kutta" (see (B.2))

$$b_1(\vartheta) = \frac{2}{3}\vartheta^3 - \frac{3}{2}\vartheta^2 + \vartheta \tag{B.6a}$$

$$b_2(\vartheta) = -\frac{2}{3}\vartheta^3 + \vartheta^2 \tag{B.6b}$$

$$b_3(\vartheta) = -\frac{2}{3}\vartheta^3 + \vartheta^2 \tag{B.6c}$$

$$b_4(\vartheta) = \frac{2}{3}\vartheta^3 - \frac{1}{2}\vartheta^2 \tag{B.6d}$$

– $p^* = 4,\ s^* = 6$ "Dormand and Prince" (see (B.3))

$$b_1(\vartheta) = -\frac{1163}{1152}\vartheta^4 + \frac{1039}{360}\vartheta^3 - \frac{1337}{480}\vartheta^2 + \vartheta \tag{B.7a}$$

$$b_2(\vartheta) = 0 \tag{B.7b}$$

$$b_3(\vartheta) = \frac{7580}{3339}\vartheta^4 - \frac{18728}{3339}\vartheta^3 + \frac{4216}{1113}\vartheta^2 \tag{B.7c}$$

$$b_4(\vartheta) = -\frac{415}{192}\vartheta^4 + \frac{9}{2}\vartheta^3 - \frac{27}{16}\vartheta^2 \tag{B.7d}$$

$$b_5(\vartheta) = -\frac{8991}{6784}\vartheta^4 + \frac{2673}{2120}\vartheta^3 - \frac{2187}{8480}\vartheta^2 \tag{B.7e}$$

$$b_6(\vartheta) = \frac{187}{84}\vartheta^4 - \frac{319}{105}\vartheta^3 + \frac{33}{35}\vartheta^2 \tag{B.7f}$$

– $p^* = 5,\ s^* = 8$ "Owren and Zennaro" (see (B.4))

$$b_1(\vartheta) = \frac{32}{15}\vartheta^5 - \frac{20}{3}\vartheta^4 + \frac{70}{9}\vartheta^3 - \frac{25}{6}\vartheta^2 + \vartheta \tag{B.8a}$$

$$b_2(\vartheta) = 0 \tag{B.8b}$$

$$b_3(\vartheta) = -\frac{128}{15}\vartheta^5 + 24\vartheta^4 - \frac{208}{9}\vartheta^3 + 8\vartheta^2 \tag{B.8c}$$

$$b_4(\vartheta) = \frac{32}{5}\vartheta^5 - 12\vartheta^4 + \frac{16}{3}\vartheta^3 \tag{B.8d}$$

$$b_5(\vartheta) = \frac{32}{5}\vartheta^5 - 20\vartheta^4 + 20\vartheta^3 - 6\vartheta^2 \tag{B.8e}$$

$$b_6(\vartheta) = -\frac{128}{15}\vartheta^5 + \frac{56}{3}\vartheta^4 - \frac{112}{9}\vartheta^3 + \frac{8}{3}\vartheta^2 \tag{B.8f}$$

$$b_7(\vartheta) = -\frac{20}{3}\vartheta^5 + 15\vartheta^4 - \frac{95}{9}\vartheta^3 + \frac{5}{2}\vartheta^2 \tag{B.8g}$$

$$b_8(\vartheta) = \frac{44}{5}\vartheta^5 - 19\vartheta^4 + 13\vartheta^3 - 3\vartheta^2 \tag{B.8h}$$

Appendix C

Further Numerical Tests

C.1 Wave Equation

In this section we want to test both numerical schemes, the ADER-DG (3.11) and the FV-WENO (3.23), separately on the respective grid. To do so we simulate the wave equation on $N_{top} = 20, 40, 80, 160, 320$ topcells and explicitely switch off the troubled cell indication for the DG scheme, whereas we consequently set all cells troubled to test the FV scheme. In both tests we apply the local RK method to calculate the space-time predictor solution.

We can rewrite the 1D wave equation $U_{tt} + cU_{xx} = 0$ in flux conservative form (3.1) by introducing two additional variables $\phi = U_x$ and $\pi = U_t$:

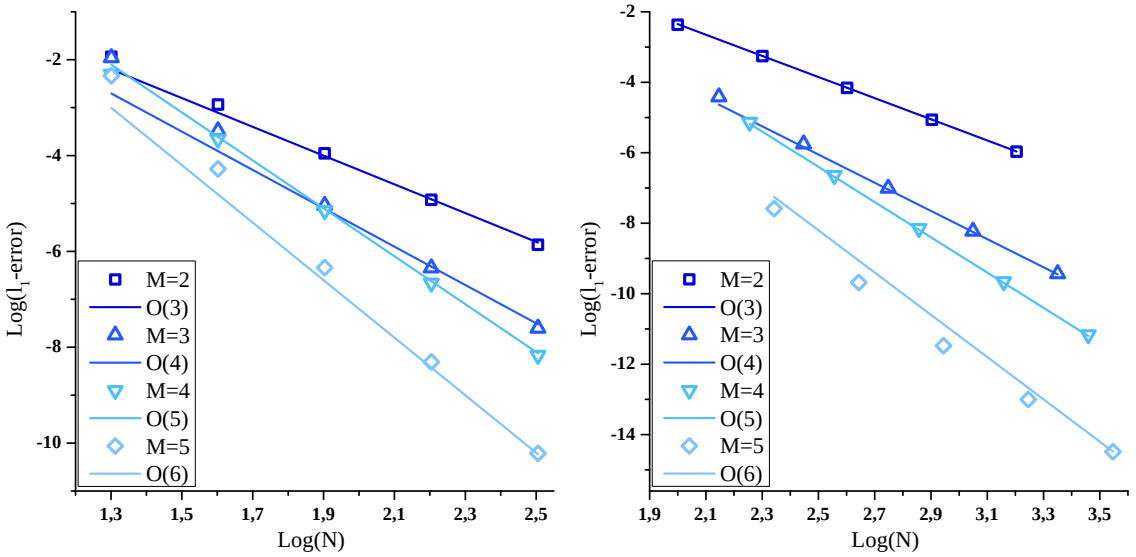
$$\frac{\partial}{\partial t}\begin{pmatrix} U \\ \phi \\ \pi \end{pmatrix} + \frac{\partial}{\partial x}\begin{pmatrix} 0 \\ -\pi \\ -c^2\phi \end{pmatrix} = \begin{pmatrix} \pi \\ 0 \\ 0 \end{pmatrix} \ . \tag{C.1}$$

As initial condition we set

$$\begin{aligned} U_0(x) &= 2\, \exp\left(-150\,[x-0.5]^2\right) \\ \phi_0(x) &= -600\,(x-0.5)\ \exp\left(-150\,[x-0.5]^2\right)\ . \\ \pi_0(x) &= 0 \end{aligned} \tag{C.2}$$

For the given setup we know the analytic solution. It consists of two partial waves travelling with the same speed c in positive and negative x-direction. We compare the numerical solution for U with the exact one at time t, where it conincides with the initial data, due to the periodic boundary conditions.

The errors in l_1-norm (7.1) together with the theoretically expected convergence orders are shown in Figure C.1. We see that they match quite well for all M tested. When comparing the error norms at a fixed number of cells we observe that the



(a) DG convergence test.

(b) FV convergence test.

Figure C.1: Convergence tests for the wave equation with initial condition (C.2) and periodic boundary conditions on $[0, 1]$. We show the l_1-error obtained for different M over the number of cells N in a double logarithmic plot. Straight lines represent the theoretically expected convergence order of $M + 1$. ($C_{FV} = 0.5$, $t = \sqrt{2}$)
(a) Test for the unlimited ADER-DG scheme (3.11) on the topgrid.
(b) Test for the FV-WENO scheme (3.23) on the subgrid. Here we started from the same number of topcells N, resulting in $(2M + 1)N$ subcells.

value for the DG-scheme is much lower than the one obtained by the FV-scheme. This is expected since we use a polynomial data representation for the first one, whereas the latter one works with cell average data.

C.2 Additional Plots for the Predictor Test

Figure 7.9 in Section 7.3 showed the quotient $\#\mathbf{F}_{ADER}/\#\mathbf{F}_{RK}$ for different degrees M of the polynomial data representation. To be more precise we depicted the mean value over multiple resolutions as used in the corresponding convergence test (see Table 7.1) and expressed the highest and lowest values with bars. Here, we want to show the complete dependency on the topcell resolution and additionally plot the results obtained in runs with very small CFL number $C_{FV} = 0.01$.

In all cases the quotient is smaller for a small CFL number, because here the iterative predictor procedure converges faster, whereas the RK predictor scheme evaluates the same number of flux functions. As already mentioned in Section 7.3 there seems to be a minimal number of iterations $iter_{min} = 3$. This can be seen

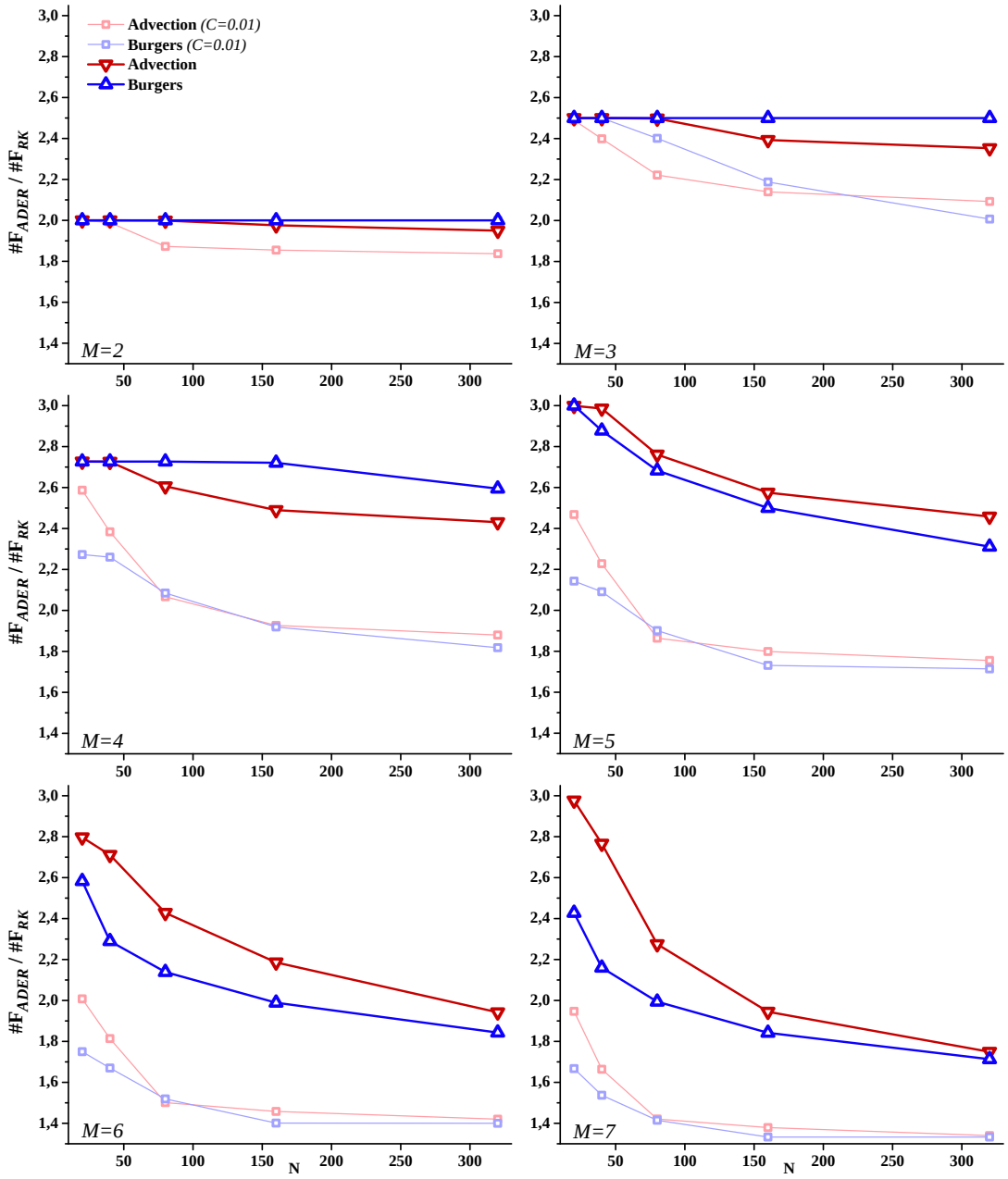


Figure C.2: Quotient $\#\mathbf{F}_{ADER}/\#\mathbf{F}_{RK}$ for different topcell resolutions N obtained from advection- and Burgers equation convergence tests. Each panel shows the results for a different degree M of the polynomial data representation. The values of the bold coloured data points are summarised in Figure 7.9. Pale markers stem from runs with very small CFL number $C = 0.01$.

in the pale data points converging to the respective value. It is worth remarking that even in these cases the quotient is larger than one which means that the newly proposed RK method for calculating the predictor solution is more efficient than the iteration of [Dumbser et al., 2013].

Bibliography

[Ben-Artzi and Falcovitz, 2003] Ben-Artzi, M. and Falcovitz, J. (2003). *Generalized Riemann Problems in Computational Fluid Dynamics*. Cambridge University Press.

[Bugner et al., 2015] Bugner, M., Dietrich, T., Bernuzzi, S., Weyhausen, A., and Brügmann, B. (2015). Solving 3D Relativistic Hydrodynamical Problems with WENO Discontinuous Galerkin Methods. *arXiv*, arXiv:1508.07147.

[Cockburn and Shu, 1989] Cockburn, B. and Shu, C. (1989). TVB Runge-Kutta Local Projection Discontiuous Galerkin Finite Element Method for Conservation Laws II: General Framework. *Mathematics of Computation*, 52:411–435.

[Dormand, 1996] Dormand, J. (1996). *Numerical Methods for Differential Equations*. CRC Press.

[Dumbser et al., 2008a] Dumbser, M., Balsara, D., Toro, E., and Munz, C. (2008a). A Unified Framework for the Construction of One-Step Finite Volume and Discontinuous Galerkin Schemes on Unstructured Meshes. *Journal of Computational Physics*, 227:8209–8253.

[Dumbser et al., 2008b] Dumbser, M., Enaux, C., and Toro, E. (2008b). Finite Volume Schemes of Very High Order of Accuracy for Stiff Hyperbolic Balance Laws. *Journal of Computational Physics*, 227:3971–4001.

[Dumbser et al., 2013] Dumbser, M., Zanotti, O., Hidalgo, A., and Balsara, D. (2013). ADER-WENO Finite Volume Schemes with Space-Time Adaptive Mesh Refinement. *Journal of Computational Physics*, 248:257–286.

[Dumbser et al., 2014] Dumbser, M., Zanotti, O., Loubère, R., and Diot, S. (2014). A Posteriori Subcell Limiting of the Discontiuous Galerkin Finite Element Method for Hyperbolic Conservation Laws. *Journal of Computational Physics*, 278:47–75.

[Gustafsson et al., 1995] Gustafsson, B., Kreiss, H., and Oliger, J. (1995). *Time Dependent Problems and Difference Methods*. John Wiley & Sons, Inc.

[Hairer et al., 2006] Hairer, E., Lubich, C., and Wanner, G. (2006). *Geometric Numerical Integration*. Springer.

[Hairer et al., 1993] Hairer, E., Nørsett, S., and Wanner, G. (1993). *Solving Ordinary Differential Equations I*. Springer.

[Harten et al., 1987] Harten, A., Engquist, B., Osher, S., and Chakravarthy, S. (1987). Uniformly High Order Accurate Essentially Non-oscillatory Schemes, III. *Journal of Computational Physics*, 71:231–303.

[Hidalgo and Dumbser, 2011] Hidalgo, A. and Dumbser, M. (2011). ADER Schemes for Nonlinear Systems of Stiff Advection-Diffusion-Reaction Equations. *Journal of Scientific Computing*, 48:173–189.

[Hilditch et al., 2015] Hilditch, D., Weyhausen, A., and Brügmann, B. (2015). A Pseudospectral Method for Gravitational Wave Collapse. *arXiv*, arXiv:1504.04732.

[Horn and Johnson, 1985] Horn, R. and Johnson, C. (1985). *Matrix Analysis*. Cambridge University Press.

[Montecinos, 2015] Montecinos, G. (2015). Analytic Solutions for the Burgers Equation with Source Terms. *arXiv*, 1503.09079v1.

[Montecinos et al., 2012] Montecinos, G., Castro, C., Dumbser, M., and Toro, E. (2012). Comparison of Solvers for the Generalized Riemann Problem for Hyperbolic Systems with Source Terms. *Journal of Computational Physics*, 231:6472–6494.

[Owren and Zennaro, 1991] Owren, B. and Zennaro, M. (1991). Continuous Explicit Runge-Kutta Methods. unpublished; url:`http://www.math.ntnu.no/~bryn/reports/imaproc.pdf`.

[Press et al., 1988] Press, W., Flannery, B., Teukolsky, S., and Vetterling, W. (1988). *Numerical Recipes in C*. Cambridge University Press.

[Qiu and Shu, 2005] Qiu, J. and Shu, C. (2005). Runge-Kutta Discontinuous Galerkin Method Using WENO Limiters. *Journal of Scientific Computing*, 26:907–929.

[Reed and Hill, 1973] Reed, W. and Hill, T. (1973). Triangular Mesh Methods for Neutron Transport Equation. *Los Alamos Scientific Laboratory*, Technical Report LA-UR-73-479.

[Shu, 1997] Shu, C. (1997). Essentially Non-oscillatory and Weighted Essentially Non-oscillatory Schemes for Hyperbolic Conservation Laws. Lecture Notes.

[Shu and Osher, 1989] Shu, C. and Osher, S. (1989). Efficient Implementation of Essentially Non-oscillatory Shock-Capturing Schemes, ii. *Journal of Computational Physics*, 83:32–78.

[Titarev and Toro, 2002] Titarev, V. and Toro, E. (2002). ADER: Arbitrary High Order Godunov Approach. *Journal of Scientific Computing*, 17:609–618.

[Toro, 2009] Toro, E. (2009). *Riemann Solvers and Numerical Methods for Fluid Dynamics*. Springer.

[Toro and Titarev, 2002] Toro, E. and Titarev, V. (2002). Solution of the Generalized Riemann Problem for Advection-Reaction Equations. In *Proceedings: Mathematical, Physical and Engineering Sciences*, pages 271–281. The Royal Society.

[Zanotti et al., 2015a] Zanotti, O., Fambri, F., and Dumbser, M. (2015a). Solving the Relativistic Magnetohydrodynamics Equations with ADER Discontinuous Galerkin Methods, A Posteriori Subcell Limiting and Adaptive Mesh Refinement. *MNRAS*, 452:3010–3029.

[Zanotti et al., 2015b] Zanotti, O., Fambri, F., Dumbser, M., and Hidalgo, A. (2015b). Space-Time Adaptive ADER Discontinuous Galerkin Finite Element Schemes with A Posteriori Sub-Cell Finite Volume Limiting. *Computers & Fluids*, 118:204–224.

Acknowledgement

I wish to express my gratitude to Prof. Bernd Brügmann for providing an opportunity to take part in his research group. We engaged in many fruitful discussions where he answered my questions and pointed me towards new, exciting problems to work on.

Special thanks goes to Marcus Bugner, Dr. David Hilditch and Tim Dietrich for their supervision, without which it would have been unthinkable to finish this thesis. I want to put emphasis on the fact that it was possible to ask questions at any time and get a qualified answer promptly. I would like to thank Niclas Moldenhauer for taking care of my computer related problems and installing any software immediately when needed. I acknowledge with thanks the organisation of various freetime activities by Tim, Marcus, Niclas and Enno, which brought our group closer together on a personal level.

I am truly thankful to Tim Dietrich, Tom Dörffel, Enno Harms, Oliver Heinsohn, David Hilditch, Martin Lewin, Jan Reislöhner, Zhanna Samsonova and Marcus Thierfelder for proofreading.

This thesis marks the end of a wonderful time as a student in Jena and Umeå. I am grateful to my girlfriend, Johanna, and all my friends who accompanied me on this way, in good times and in bad.

Zu guter Letzt möchte ich mich bei meinen Eltern bedanken, die mir mein Studium ermöglicht und mich in all meinen Entscheidungen unterstützt haben.

Erklärung

Selbstständigkeitserklärung

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie Zitate und gedankliche Übernahmen kenntlich gemacht habe.

Matthias Pilz

Bibliothekserklärung

Seitens des Verfassers bestehen keine Einwände, die vorliegende Masterarbeit für die öffentliche Nutzung in der Thüringer Universitäts- und Landesbibliothek zur Verfügung zu stellen.

Matthias Pilz