CONVERGENCE OF AN IMPLICIT RUNGE-KUTTA DISCONTINUOUS GALERKIN METHOD USING SMOOTH LIMITERS

Jacob Middag

FACULTY OF ELECTRICAL ENGINEERING, MATHEMATICS AND COMPUTER SCIENCE
CHAIR: MATHEMATICS OF COMPUTATIONAL SCIENCE

UNIVERSITY OF TWENTE.

AUGUST 23, 2012

Abstract

In higher order discontinuous Galerkin methods limiters are used to remove non-physical numerical oscillations. The limiters are used as a postprocessing step after each stage or time step. However, the limited solution is not a solution to the DG formulation and often limiters contain switches which are non-smooth. This results into a limit cycle behavior which hampers convergence of iterative methods used for the solution of algebraic equations resulting from an implicit time integration method.

We will investigate a new smooth limiter in this thesis, the Weighted Biased Averaging Procedure (WBAP), to address this problem. The limiter is adapted to be used in an implicit discontinuous Galerkin method and the modifications necessary for a DG algorithm will be discussed in detail.

The WBAP limiter is applied both in a one dimensional and a two dimensional setting. In the one dimensional setting this is done using shock tube problems described by the Euler equations for gas dynamics and steady state problems described by the Burgers equation. For the two dimensional problem we consider the shallow water equations for the flow in a channel with contraction.

The application of the DG method is successful in both dimensions. However, the result of the simulations do not confirm the hypothesis that smoothness is beneficial for the convergence of the implicit time integration method. The most discontinuous WBAP variant is only slightly better than the discontinuous minmod-TVB limiter and the other variants are worse. Compared to the results without limiter there is still a large gap in convergence rate. For future work it is recommended to look for methods that can deal with the discontinuous properties of limiters such as semi-smooth Newton methods.

Contents

Chapter 1

Introduction

Scientific problems in many fields of physics are described by partial differential equations (PDE's). Except for simple problems or problems with enough symmetry, most of them cannot be solved analytically. Therefore numerical methods have been developed to solve the PDE's with a computer. The most important classes of numerical methods are finite difference methods (FDM), finite volume methods (FVM) and finite element methods (FEM). A relatively new method is the *discontinuous Galerkin* (DG) method. Originally it was used to solve hyperbolic PDE's, but the method is extended to solve elliptic and parabolic PDE's as well.

The DG method is based on the finite element method, but uses *discontinuous* basis functions – in most cases polynomials – instead of continuous basis functions as is the case in classical FEM. Due to the discontinuous basis it frees the method from some restrictions in classical FEM, e.g. arbitrary triangulation, hanging nodes and changing the polynomial order per element (p-adaptivity) are easily implemented. The discontinuous basis functions make the method appealing for problems with discontinuities, eg. shocks and material interfaces. Also, due to the local nature of the basis functions, only the nearest neighbors of an element have to be considered, which makes the algorithm very attractive for parallel computing.

The use of discontinuous basis functions also has its disadvantages. As known under the name Godunov's Theorem, the DG method lacks the property of not generating new extrema in the solution when using linear or higher order polynomial basis functions. This is only possible for constant basis functions in combination with an upwind (or approximate Riemann) flux. In order to be sufficiently accurate, the DG method generally uses first or higher order polynomials and encounters therefore non-physical numerical oscillations in problems with discontinuities, interfaces and very steep gradients. These oscillations seriously affect the solution and can cause non-physical values in applications, such as negative density in a compressible flow or negative water height. Not only is this unrealistic, but it can also cause the method to blow up or halt.

These numerical oscillations also arise in finite volume methods. DG methods are very similar to finite volume methods and share important components, such as the flux functions at the element faces. In fact, a zeroth order DG method has the same formulation as a zeroth order finite volume method. The main solution to remove the numerical oscillations is the use of slope or flux limiters. Originating from finite volume methods, they compare different gradients in the reconstruction process (MUSCL framework with TVD limiters [32, 14, 29]) or alter the reconstruction process such that it reduces the oscillations (ENO/WENO schemes [15, 23, 18]).

Limiters and numerical fluxes developed for finite volume methods are also suitable for DG methods. Since DG methods do not have to use a reconstruction method as in MUSCL and ENO/WENO methods – DG methods only use basis functions inside the elements – the limiters have to be altered. The limited solution is, however, no weak solution to the DG formulation of the PDE and the key issue in applying limiters in a DG discretization is that they are used as a post- or preprocessing step.

The main difficulty in the use of limiters lies in the fact that most limiters are formulated using conditions and selection procedures which result in a switching behavior and non-smooth flux functions. This makes it difficult to incorporate the limiter in a space-time DG method (i.e. a method that discretizes space and time simultaneously using a DG method) and in iterative methods for the solution of non-linear equations resulting from implicit time integration methods [33].

Li et al. [22],[21] recently presented a new smooth limiter which suffers less from these drawbacks of limiters. This smooth limiter is successfully applied in a finite volume framework on structured and unstructured grids using explicit time integration methods. This would make the limiter a good candidate to use in direct combination with the DG discretization. In this thesis we will explore its suitability for one dimensional and two dimensional problems discretized with an implicit time integration method.

The thesis is organized in the following way. In Chapter 2 we will introduce the DG formulation for one dimensional and two dimensional conservation laws. We will proof that it satisfies the entropy condition for the scalar one dimensional case. Furthermore we will look into Runge-Kutta time discretization methods. Implementation details such as basis functions and transformations to the reference element will be given in Chapter 3. The WBAP limiter and its variations will be introduced in Chapter 4. Since the limiter is originating from a finite volume method we will show how the limiter is applied in the DG method in both the one dimensional and two dimensional setting. The chapter concludes with the derivatives needed to solve the algebraic equations. In Chapter 5 the solution to the non-linear algebraic equations will be treated and the Jacobians of the operators will be calculated.

In the second part of the thesis we will test the smooth limiter in three different setups. In Chapter 6 we use the Euler equations to see how the radius of convergence is affected by including the limiter and compare it to the basic and reliable TVB-minmod [8] limiter. How the limiter behaves in the case of convergence towards a steady state solution is discussed in Chapter 7 with the use of the Burgers equation. Chapter 8 will look into a more realistic two dimensional system given by the shallow water equations in a channel with contraction and we will compare the WBAP limiter to other limiters [4, 33, 25]. The final chapter will discuss the results, draw conclusions and presents an outlook for future work.

Chapter 2

Discontinuous Galerkin method

The first discontinuous Galerkin (DG) method was introduced in 1973 by Reed and Hill [27] to solve the hyperbolic neutron mass transport equation. Although we will look only into hyperbolic equations, DG can also be applied to elliptic and parabolic problems [3]. The DG method is based on the finite element method (FEM) and uses components from the finite volume method (FVM). Like the Galerkin formulation of FEM, the DG formulation starts with the weak formulation and uses the same functions as test and trial functions. The discretization is obtained by using a finite basis.

Within the DG method there are two separate approaches to handle the time derivative in time dependent problems. The first is to treat it as a spatial variable, the space-time DG method [16, 19]. The second approach is to first discretize in space with DG and then use a time integration method to solve the resulting ordinary differential equations. A widely used time integration method is the Runge-Kutta method, known in combination with DG as the Runge-Kutta discontinuous Galerkin (RKDG) method [9].

In this chapter we will discretize one and two dimensional conservation laws using a DG formulation. This leads to the definition of a spatial operator $\mathcal{D}$ which will result in a system of ordinary differential equations. In the last section we will show how this system is solved.

2.1 One dimensional hyperbolic systems

Let us consider a one dimensional conservation law of d variables, with time dependent variable t, spatial variable x, flux term $\mathbf{f}$ and a source term $\mathbf{s}$:

$$\mathbf{u}_t + \mathbf{f}(\mathbf{u})_x = \mathbf{s}(\mathbf{u}) \; , \tag{2.1}$$

$\mathbf{f}$ is a $\mathbb{R}^d \to \mathbb{R}^d$ function which differs per physical phenomena and is often non-linear. Examples are the advection, Burgers and shallow water equations. V is defined as the space in which the solution $\mathbf{u}$ lives. Furthermore we assume periodic boundary conditions in this chapter to ease the theoretical treatment.

In the one dimensional DG method we divide the domain into N elements, denoted by

$$I_i = \Big\{ x : x_{i-1/2} \le x \le x_{i+1/2} \Big\}, \quad 1 \le i \le N \ , \tag{2.2}$$

with

$$x_{\text{left}} = x_{1/2} < x_1 < \cdots < x_N < x_{N+1/2} = x_{\text{right}} \ , \tag{2.3}$$

where the $i - 1/2$ and $i + 1/2$ notation is used to denote the left, respectively, right boundary of element i and x_i to denote the midpoints. Note that the elements do not have to be distributed uniformly. The width of the element and the minimum width are given by:

$$\Delta x_i = x_{i+1/2} - x_{i-1/2}, \qquad\qquad h = \min_{1 \le i \le N} \Delta x_i \ . \tag{2.4}$$

We assume that the mesh is regular, i.e. there exists a constant c independent of h such that:

$$\Delta x_i \ge ch, \quad 1 \le i < N \ .$$

For a discretization we have to define a finite subset of the solution space. This is where the discretization takes place. Although one can propose any function space with a finite basis span, it is numerically advantageous to use local polynomial test and trial functions. Therefore we define the finite element space consisting of polynomials up to the order p as

$$V_h^{p,d} = \{ \mathbf{v} \in \left(L^2(\Omega)\right)^d : \mathbf{v}|_{I_i} \in \left(P^p(I_i)\right)^d ; 1 \le i \le N \} \subset V^d \ . \tag{2.5}$$

The trial and test functions are in contrast with continuous FEM truly local and due to the use of basis functions, which are discontinuous at element faces, the function $\mathbf{u} \in V_h^{p,d}$ is not necessarily continuous at the faces of the elements.

The DG method is derived from the weak formulation and can be formulated as finding $\mathbf{u} \in V_h^{p,d}$, such that for all $\mathbf{v} \in V_h^{p,d}$ the following holds:

$$\sum_{i=1}^{N} \left\{ \int_{I_i} \big[\mathbf{u}_t - \mathbf{s}(\mathbf{u}) \big] \, \mathbf{v} \, \mathrm{d}x + \int_{I_i} \mathbf{f}(\mathbf{u})_x \, \mathbf{v} \, \mathrm{d}x \right\} = 0$$

at which we can integrate the second term by parts and obtain:

$$\sum_{i=1}^{N} \left\{ \int_{I_i} \big[\mathbf{u}_t - \mathbf{s}(\mathbf{u}) \big] \, \mathbf{v} \, \mathrm{d}x - \int_{I_i} \mathbf{f}(\mathbf{u}) \, \mathbf{v}_x \, \mathrm{d}x + \hat{\mathbf{f}}(\mathbf{u}) \, \mathbf{v} \Big|_{x_{i-1/2}}^{x_{i+1/2}} \right\} = 0 \ . \tag{2.6}$$

At the boundaries $\mathbf{f}$ is replaced with $\hat{\mathbf{f}}$, the numerical flux, since the flux at the element faces is not guaranteed to be single valued due to the use of basis functions which are discontinuous at element faces. The integrals can be approximated by numerical quadratures, e.g. Gauss or Gauss-Lobatto rules. To complete the DG space discretization we only have to define the numerical flux.

2.1.1 Numerical flux

Due to the discontinuous basis functions the trace at the element faces is double valued and therefore a numerical flux is introduced. This flux should depend on the elements which constitute the face. The choice of an appropriate numerical flux in DG includes two main ideas. The first one is to let the numerical flux only depend on the values at the faces. This is convenient since we only have to evaluate the solution at that point. The second one is to choose the numerical flux such that it results in a monotone finite volume scheme for piecewise-constant basis function. The motivation is that only first order, monotone schemes seem to be stable and convergent to the exact solution. Along with the natural consistency condition that if the face values are equal then the numerical flux should return the same value as the original flux, we obtain the following three conditions:

1. Consistency: $\hat{\mathbf{f}}(\mathbf{u},\mathbf{u}) = \mathbf{f}(\mathbf{u})$
2. Continuity: $\hat{\mathbf{f}}(\mathbf{u}^-,\mathbf{u}^+)$ is at least Lipschitz continuous with respect to both arguments $\mathbf{u}^-$ and $\mathbf{u}^+$
3. Monotonicity: $\hat{\mathbf{f}}(\mathbf{u}^-,\mathbf{u}^+)$ is a non-decreasing function in its first argument $\mathbf{u}^-$ and a non-increasing function in its second argument $\mathbf{u}^+$.

Using this notation, (2.6) can be written as:

$$\sum_{i=1}^{N}\Big\{\int_{I_i}\big[\mathbf{u}_t - \mathbf{s}(\mathbf{u})\big]\,\mathbf{v}\,\mathrm{d}x - \int_{I_i}\mathbf{f}(\mathbf{u})\,\mathbf{v}_x\,\mathrm{d}x + \hat{\mathbf{f}}\big(\mathbf{u}(x^-_{i+1/2}),\mathbf{u}(x^+_{i+1/2})\big)\,\mathbf{v}(x^-_{i+1/2}) - \hat{\mathbf{f}}\big(\mathbf{u}(x^-_{i-1/2}),\mathbf{u}(x^+_{i-1/2})\big)\,\mathbf{v}(x^+_{i-1/2})\Big\} = 0\,. \tag{2.7}$$

We will use the well-known Lax-Friedrichs flux in this report. Although it introduces more artificial viscosity than the Godunov flux, the Lax-Friedrichs flux is smooth and very easy to calculate. This is important because we need to solve a non-linear equation introduced by the implicit time integration method. The Lax-Friedrichs flux is given by:

$$\hat{\mathbf{f}}^{\mathrm{LF}}(\mathbf{a},\mathbf{b}) = \frac{1}{2}\Big[\mathbf{f}(\mathbf{a}) + \mathbf{f}(\mathbf{b}) - \alpha(\mathbf{b}-\mathbf{a})\Big]\,, \tag{2.8}$$

with α the maximum characteristic wave speed of the system, i.e. the maximum modulus of the eigenvalues of the Jacobian of $\mathbf{f}$. This can be imposed locally or globally. Local imposition reduces the dissipation introduced by α in many elements, but costs extra time to compute and – more importantly – introduces a dependency on the solution which makes it harder to calculate the derivative. For the one dimensional problem we chose to impose the α globally.

2.1.2 Boundary conditions

Homogeneous Neumann boundary conditions

If we want to study a shock tube problem to test the limiter for suppressing numerical oscillations, we have to implement homogeneous Neumann boundary conditions. The Neumann boundary condition of $g(t) = u_x(0,t)$ is implemented using a *ghost element* with the same width as the boundary element via a finite difference relation:

$$\frac{u_1 - u_0}{\Delta x_i} = g(t) \tag{2.9}$$

For homogeneous boundary conditions the ghost element is equal to the boundary element and the numerical flux is reduced to:

$$\hat{f}(u_0, u_1) = \hat{f}(u_1, u_1) = f(u_1) \tag{2.10}$$

The boundary condition on the other side is completely analogue.

Dirichlet boundary conditions

Dirichlet boundary conditions are enforced via the numerical flux. The data component outside the domain which is necessary to compute the flux at the boundary is preset to the Dirichlet condition. Although this means that the Dirichlet condition is not enforced pointwise, it results in less oscillations.[10]

2.1.3 Entropy

The DG method uses a weak formulation of (2.1). Solutions to this weak formulation may not be unique (e.g. when faced with discontinuous initial conditions). The physically unique solution must satisfy the following *entropy inequality*:

$$U(u)_t + F(u)_x \leq 0 \ , \tag{2.11}$$

in the sense of a distribution, for any convex entropy U, satisfying $U'' \geq 0$ and the corresponding entropy flux

$$F(u) = \int^u U'(v) f'(v) \, \mathrm{d}v \ .$$

This solution is therefore also known as the *entropy solution.*

To calculate physically realistic phenomena it is desirable for a numerical approximation to share the same entropy inequality. In contrast with finite difference or finite volume schemes, it is easy to prove that the DG scheme satisfies a cell entropy inequality [17]. We will show in this section that for a scalar conservation law with no source term, the physically unique solution is obtained using the DG method. This can be proved element-wise. For element i the scalar version of (2.6) is given as:

$$\int_{I_i} u_t v \,\mathrm{d}x - \int_{I_i} f(u)\, v_x \,\mathrm{d}x + \hat{f}_{i+1/2} v(x^-_{i+1/2}) - \hat{f}_{i-1/2} v(x^+_{i-1/2}) = 0 \ , \tag{2.12}$$

with $\hat{f}_{i\pm1/2}$ being defined as:

$$\hat{f}_{i\pm1/2} = \hat{f}\big(u(x^-_{i\pm1/2}), u(x^+_{i\pm1/2})\big) \ . \tag{2.13}$$

This leads to the following proposition:

Proposition 2.1. *The solution u to the scalar semi-discrete DG scheme* (2.12) *satisfies the following cell entropy inequality:*

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{I_i} U(u)\,\mathrm{d}x + \hat{F}_{i+1/2} - \hat{F}_{i-1/2} \leq 0 \ , \tag{2.14}$$

for the square entropy $U(u) = \frac{1}{2}u^2$, for some consistent entropy flux

$$\hat{F}_{i+1/2} = \hat{F}\big(u(x^-_{i+1/2}), u(x^+_{i+1/2})\big) \ ,$$

satisfying $\hat{F}(u,u) = F(u)$.

Proof. If we take our weak formulation (2.12) and use as test function u itself, we get the following equality:

$$\int_{I_i} u_t u \,\mathrm{d}x - \int_{I_i} f(u)\, u_x \,\mathrm{d}x + \hat{f}_{i+1/2} u(x^-_{i+1/2}) - \hat{f}_{i-1/2} u(x^+_{i-1/2}) = 0 \ . \tag{2.15}$$

The first term can be written in terms of U as:

$$\int_{I_i} u_t u \,\mathrm{d}x = \frac{1}{2}\int_{I_i} \frac{\partial}{\partial t} u^2 \,\mathrm{d}x = \int_{I_i} U(u)_t \,\mathrm{d}x = \frac{\mathrm{d}}{\mathrm{d}t}\int_{I_i} U(u)\,\mathrm{d}x \ ,$$

and if we denote $\tilde{F}$ as:

$$\tilde{F}(u) = \int^u f(v)\,\mathrm{d}v \ ,$$

we can change the second term by

$$-\int_{I_i} f(u)\, u_x \,\mathrm{d}x = -\int_{I_i} \left\{ \frac{\partial}{\partial x}\tilde{F}(u) \right\} \mathrm{d}x = -\tilde{F}\big(u(x^-_{i+1/2})\big) + \tilde{F}\big(u(x^+_{i-1/2})\big) \ .$$

Introducing the contributions into (2.15) gives:

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{I_i} U(u)\,\mathrm{d}x - \tilde{F}\big(u(x^-_{i+1/2})\big) + \tilde{F}\big(u(x^+_{i-1/2})\big) + \hat{f}_{i+1/2} u(x^-_{i+1/2}) - \hat{f}_{i-1/2} u(x^+_{i-1/2}) = 0 \ . \tag{2.16}$$

Now we will define our consistent entropy flux to be:

$$\hat{F}_{i\pm1/2} = -\tilde{F}\big(u(x^-_{i\pm1/2})\big) + \hat{f}_{i\pm1/2} u(x^-_{i\pm1/2}) \ . \tag{2.17}$$

Note that although it looks like $\hat{F}$ does only depend on one variable, it is $\hat{f}_{i\pm 1/2}$ that defines the other variable automatically being $u(x^+_{i\pm 1/2})$ in (2.13). This entropy flux needs to be consistent, which can be proven using the consistency property of the numerical flux and integration by parts. Working backwards:

$$\begin{aligned} F(u) = \int^u U'(v) f'(v) \, \mathrm{d}v = \int^u v f'(v) \, \mathrm{d}v \\ = - \int^u f(v) \, \mathrm{d}v + u f(u) = -\tilde{F}\big(u\big) + u\hat{f}(u,u) = \hat{F} \; . \end{aligned}$$

Using our entropy flux (2.17) we can rewrite (2.16) to a cell entropy inequality)

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_{I_i} U(u) \, \mathrm{d}x + \hat{F}_{i+1/2} - \hat{F}_{i-1/2} + \underbrace{\Big[\hat{F}_{i-1/2} + \tilde{F}\big(u(x^+_{i-1/2})\big) - \hat{f}_{i-1/2} u(x^+_{i-1/2})\Big]}_{\Theta} = 0 \; . \quad (2.18)$$

To proof the entropy inequality we only have to prove that $\Theta \geq 0$. Since all functions are evaluated at $x_{i-1/2}$ we will drop the subscript and adopt the notation that $u^\pm = u(x^\pm)$. Writing out $\hat{F}_{i-1/2}$, applying the mean value theorem[1] and using the fact that $\tilde{F}'(v) = f(v)$ gives:

$$\begin{aligned} \Theta &= \big[-\tilde{F}(u^-) + \hat{f}\,u^- \big] + \tilde{F}(u^+) - \hat{f}\,u^+ = \big[\tilde{F}(u^+) - \tilde{F}(u^-)\big] - (u^+ - u^-)\hat{f} \\ &= (u^+ - u^-)\big[\tilde{F}'(\xi) - \hat{f}\,\big] = (u^+ - u^-)\big[f(\xi) - \hat{f}\,\big] = (u^+ - u^-)\big[\hat{f}(\xi,\xi) - \hat{f}(u^-,u^+)\big] \; , \end{aligned}$$

with ξ between u^+ and u^-. We used again the consistency property of the numerical flux which states: $\hat{f}(\xi,\xi) = f(\xi)$. Using the monotonicity property of the numerical flux it follows that $u^+ - u^-$ should have the same sign as $\hat{f}(\xi,\xi) - \hat{f}(u^-,u^+)$. Which concludes that $\Theta \geq 0$ and finishes the proof. $\square$

Note that the proof does not depend on the polynomial order of the basis functions nor the accuracy of the scheme.

2.2 Two dimensional hyperbolic systems

A two dimensional conservation law is the same as a one dimensional conservation, but with an extra flux term for the extra spatial dimension:

$$\mathbf{u}_t + \mathbf{f}(\mathbf{u})_x + \mathbf{g}(\mathbf{u})_y = \mathbf{s}(\mathbf{u}) \; . \quad (2.19)$$

For two (and higher) dimensional equations one has to choose a geometry for the elements. Triangles and quadrilaterals are often used for this purpose in two dimensional problems. We divide our spatial domain up into triangular elements denoted by $\mathcal{T}_h$, because the transformation to the reference element is linear and is simple to implement. Note that elements with different geometries can be mixed in one problem.

[1] $\tilde{F}$ satisfies the conditions of differentiable on the open and continuous on the closed interval because it is defined using an integral

The weak formulation of (2.19) is given by:

$$\sum_{K_i \in \mathcal{T}_h} \left\{ \int_{K_i} \left[\mathbf{u}_t - \mathbf{s}(\mathbf{u})\right] \mathbf{v} \, \mathrm{d}x\mathrm{d}y + \int_{K_i} \mathbf{f}(\mathbf{u})_x \, \mathbf{v} \, \mathrm{d}x\mathrm{d}y + \int_{K_i} \mathbf{g}(\mathbf{u})_y \, \mathbf{v} \, \mathrm{d}x\mathrm{d}y \right\} = 0 \ ,$$

or if we introduce $\mathbf{F} = [\mathbf{f}, \mathbf{g}]^{\mathrm{T}}$ we can write it as:[2]

$$\sum_{K_i \in \mathcal{T}_h} \left\{ \int_{K_i} \left[\mathbf{u}_t - \mathbf{s}(\mathbf{u})\right] \mathbf{v} \, \mathrm{d}x\mathrm{d}y + \int_{K_i} \nabla \cdot \mathbf{F}(\mathbf{u}) \, \mathbf{v} \, \mathrm{d}x\mathrm{d}y \right\} = 0 \ ,$$

at which we can apply the two dimensional divergence theorem and obtain:

$$\sum_{K_i \in \mathcal{T}_h} \left\{ \int_{K_i} \left[\mathbf{u}_t - \mathbf{s}(\mathbf{u})\right] \mathbf{v} \, \mathrm{d}x\mathrm{d}y - \int_{K_i} \mathbf{F}(\mathbf{u}) \cdot \nabla \mathbf{v} \, \mathrm{d}x\mathrm{d}y + \int_{\partial K_i} \hat{\mathbf{F}}(\mathbf{u}^-, \mathbf{u}^+) \cdot \hat{\mathbf{n}} \, \mathbf{v} \, \mathrm{d}s \right\} = 0 \ , \quad (2.20)$$

with $\mathbf{u}^-$ to be defined as the trace from inside the element and $\mathbf{u}^+$ (and thus inside the neighboring element) to be defined as the trace from outside the element, $\hat{\mathbf{n}}$ is the outward pointing normal of element K_i. Note that $\mathbf{u}^+$ is in a different element for each element face. To simplify the numerical implementation it is easier to transfer the sum of the element faces into a sum over interior and boundary faces:

$$\begin{aligned} \sum_{K_i \in \mathcal{T}_h} \left\{ \int_{K_i} \left[\mathbf{u}_t - \mathbf{s}(\mathbf{u})\right] \mathbf{v} \, \mathrm{d}x\mathrm{d}y \right\} - \int_{K_i} \mathbf{F}(\mathbf{u}) \cdot \nabla \mathbf{v} \, \mathrm{d}x\mathrm{d}y & \\ + \sum_{S \in S_i} \int_{\partial K_i} \left(\hat{\mathbf{F}}(\mathbf{u}^-, \mathbf{u}^+) \cdot \mathbf{n}^- \, \mathbf{v}^- + \hat{\mathbf{F}}(\mathbf{u}^+, \mathbf{u}^-) \cdot \mathbf{n}^+ \, \mathbf{v}^+ \right) \mathrm{d}s & \\ + \sum_{S \in S_b} \int_{\partial K_i} \hat{\mathbf{F}}(\mathbf{u}^-, \mathbf{u}^+) \cdot \hat{\mathbf{n}}^- \, \mathbf{v} \, \mathrm{d}s = 0 \ , & \quad (2.21) \end{aligned}$$

where the normal $\hat{\mathbf{n}}^-$ has to be taken outward with respect to the element associated with $\mathbf{v}^-$.

The Lax-Friedrichs flux is given by the relation:

$$\hat{\mathbf{F}}^{\mathrm{LF}}(\mathbf{u}^-, \mathbf{u}^+) \cdot \hat{\mathbf{n}} = \frac{1}{2} \left\{ \left[\mathbf{F}(\mathbf{u}^-) + \mathbf{F}(\mathbf{u}^+) \right] \cdot \hat{\mathbf{n}} - \alpha(\mathbf{u}^-, \mathbf{u}^+) \, (\mathbf{u}^+ - \mathbf{u}^-) \right\} \ , \quad (2.22)$$

where α is taken as an upper bound for the characteristic wave speeds in the normal direction at the element face, i.e. the maximum modules of the eigenvalues of the Jacobian of the inner product of $\mathbf{F}$ with $\hat{\mathbf{n}}$. The coefficient α is dependent on $\mathbf{u}^-$ and $\mathbf{u}^+$ because we will calculate it locally at each element face. The line integral and volume integral can be obtained by numerical quadrature just as in the one dimensional case. Note when using the Lax-Friedrichs flux as numerical flux in (2.7) we only have to calculate the flux once. The flux for the element connected to the face can be calculated via the skew-symmetric relation:

$$\hat{\mathbf{F}}^{\mathrm{LF}}(\mathbf{u}^-, \mathbf{u}^+) \cdot \hat{\mathbf{n}}^- = -\hat{\mathbf{F}}^{\mathrm{LF}}(\mathbf{u}^+, \mathbf{u}^-) \cdot \hat{\mathbf{n}}^+ \ . \quad (2.23)$$

[2]Abusing notation a little bit, but the dot product has to be taken with respect to the two dimensional field created.

2.3 DG discretization

After the discretization in space discussed in the previous section, the time derivative is still present in the system of equations. To simplify this section we will base the equations on the two dimensional case, but the one dimensional case can be done analogously.

2.3.1 Trial functions

Let's denote the trial functions by $\varphi_{k,i}(x,y)$, where k denotes the index of the L different basis functions per element and i the element number of the N elements:

$$\mathbf{u} = (u_1, \cdots, u_d), \qquad u_j(t,x,y) = \sum_{i=1}^{N} \sum_{k=1}^{L} \vec{u}_{j,k,i}(t)\, \varphi_{k,i}(x,y) \qquad j = 1, \cdots, d\,, \tag{2.24}$$

where $\vec{u}$ shall be used to denote the coefficients in contrast with $\mathbf{u}$ (and u_j) which denotes the solution itself (or the j component). The coefficients have to depend on t, since the basis functions do not.

If we choose the basis functions to be orthonormal the time derivative[3]

$$\int_{K_i} \mathbf{u}_t\, \mathbf{v}\, \mathrm{d}x\mathrm{d}y = \frac{\mathrm{d}}{\mathrm{d}t} \int_{K_i} \mathbf{u}\, \mathbf{v}\, \mathrm{d}x\mathrm{d}y$$

is explicit and no mass matrix has to be inverted. But even if the basis is not orthonormal, the mass matrix is local per element and has size $L \times L$. This matrix can be inverted beforehand and the cost is only a matrix multiplication of the element mass matrix with the DG coefficients. We define the mass matrix on element i as M_i, which is defined per entry as:

$$(M_i)_{kl} = \int_{K_i} \varphi_{k,i}\, \varphi_{l,i}\, \mathrm{d}x\mathrm{d}y\,. \tag{2.25}$$

When the basis is orthogonal but not orthonormal, the mass matrix is diagonal and the coefficients only have to multiplied with the inverse of the diagonal entry.

2.3.2 Method of lines for time integration

In this section we will define a spatial DG operator to apply standard time integration techniques to the resulting system of ordinary differential equations (ODE's). Our starting point is the weak formulation in (2.20) and the basis functions as used in (2.24) which will also be used as test functions. The test space is therefore given by the span of the basis functions and leads to the same number of equations. In this section we will use this basis span and take an arbitrary basis function as test function.

[3]Note that the test function $\mathbf{v}$ is not dependent on t.

If we manipulate (2.20) such that the time derivative is at one side we obtain for u_j, the j-th component of $\mathbf{u}$, the following equation:

$$\sum_{K_i\in\mathcal{T}_h} \frac{\mathrm{d}}{\mathrm{d}t}\int_K u_j\,\varphi_{k,i}\,\mathrm{d}x\mathrm{d}y = \sum_{K_i\in\mathcal{T}_h}\Bigg\{\int_{K_i} s(\mathbf{u})_j\,\varphi_{k,i}\,\mathrm{d}x\mathrm{d}y + \int_{K_i} F(\mathbf{u})_j\cdot\nabla\varphi_{k,i}\,\mathrm{d}x\mathrm{d}y - \int_{\partial K_i}\Big[\hat{F}(\mathbf{u}^-,\mathbf{u}^+)\cdot\hat{\mathbf{n}}\Big]_j\,\varphi_{k,i}\,\mathrm{d}s\Bigg\}\,.$$

The test function is only non-zero in element i and therefore singles out the integral over element i. Introducing the expansion of (2.24) for u_j in the left hand side gives:

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{K_i}\Big(\sum_{l=1}^{L}\vec{u}_{j,l,i}\,\varphi_{l,i}\Big)\varphi_{k,i} = \int_{K_i} s(\mathbf{u})_j\,\varphi_{k,i}\,\mathrm{d}x\mathrm{d}y + \int_{K_i} F(\mathbf{u})_j\cdot\nabla\varphi_{k,i}\,\mathrm{d}x\mathrm{d}y - \int_{\partial K_i}\Big[\hat{F}(\mathbf{u}^-,\mathbf{u}^+)\cdot\hat{\mathbf{n}}\Big]_j\,\varphi_{k,i}\,\mathrm{d}s\,.$$

The integral on the left hand side can be replaced with the mass matrix entries after switching the order of the integral, sum, and time derivative. This results into:

$$\sum_{l=1}^{L}(M_i)_{k,l}\,\frac{\mathrm{d}}{\mathrm{d}t}\Big(\vec{u}_{j,l,i}\Big) = \int_{K_i} s(\mathbf{u})_j\,\varphi_{k,i}\,\mathrm{d}x\mathrm{d}y + \int_{K_i} F(\mathbf{u})_j\cdot\nabla\varphi_{k,i}\,\mathrm{d}x\mathrm{d}y - \int_{\partial K_i}\Big[\hat{F}(\mathbf{u}^-,\mathbf{u}^+)\cdot\hat{\mathbf{n}}\Big]_j\,\varphi_{k,i}\,\mathrm{d}s\,.$$

The summation on the left hand can be written as a matrix multiplication if we take all trial function of element i and component j. We will use the Matlab based notation : to denote all the entries available in that index position. This leads to the system of (L) equations for element i:

$$\mathrm{M}_i\cdot\frac{\mathrm{d}}{\mathrm{d}t}\Big(\vec{u}_{j,:,i}\Big) = \int_{K_i} s(\mathbf{u})_j\,\varphi_{:,i}\,\mathrm{d}x\mathrm{d}y + \int_{K_i} F(\mathbf{u})_j\cdot\nabla\varphi_{:,i}\,\mathrm{d}x\mathrm{d}y - \int_{\partial K_i}\Big[\hat{F}(\mathbf{u}^-,\mathbf{u}^+)\cdot\hat{\mathbf{n}}\Big]_j\,\varphi_{:,i}\,\mathrm{d}s \equiv \mathrm{M}_i\cdot\mathcal{D}^{(2)}_{i,j}(\vec{u}) \quad (2.26)$$

from which we can derive a spatial operator $\mathcal{D}$ and write:[4]

$$\frac{\mathrm{d}}{\mathrm{d}t}\vec{u} = \mathcal{D}^{(2)}(\vec{u}) \qquad (2.27)$$

which we have to solve by marching in time.

[4]The $^{(2)}$ is used to distinguish the two dimensional DG operator from the one dimensional one.

The one dimensional spatial DG operator can analogously be derived from:

$$\begin{aligned}\mathrm{M}_i \cdot \frac{\mathrm{d}}{\mathrm{d}t}\Big(\vec{u}_{j,:,i}\Big) &= \int_{I_i} s(\mathbf{u})_j \, \varphi_{:,i} \, \mathrm{d}x + \int_{I_i} f(\mathbf{u})_j \, \frac{\mathrm{d}}{\mathrm{d}x}\Big(\varphi_{:,i}\Big) \, \mathrm{d}x \\ &\quad - \hat{f}\big(\mathbf{u}(x^-_{i+1/2}), \mathbf{u}(x^+_{i+1/2})\big)_j \, \varphi_{:,i}(x^-_{i+1/2}) + \hat{f}\big(\mathbf{u}(x^-_{i-1/2}), \mathbf{u}(x^+_{i-1/2})\big)_j \, \varphi_{:,i}(x^+_{i-1/2}) \\ &\equiv \mathrm{M}_i \cdot \mathcal{D}^{(1)}_{i,j}(\vec{u}) \ , \qquad (2.28)\end{aligned}$$

where M_i is the local mass matrix for an element in the one dimensional setting.

2.3.3 Time integration

One can distinguish two classes of time integration methods: explicit and implicit. While the intermediate stages in explicit methods only depend on the previous ones, implicit methods also depend on the current stage. For non-linear systems of equations this usually involves using iterative methods to solve the corresponding non-linear algebraic equations.

We will first consider explicit Runge-Kutta (RK) methods for time integration. Runge-Kutta methods are used to solve the ordinary differential equations (ODE) as (2.27). They are generalized Euler methods in the sense that they allow for a number of evaluations of the derivative to take place in one time step [5]. The reason to do this is to obtain a higher order discretization in time. This is accomplished by using intermediate stages.

Explicit methods can be straightforwardly used. The time step is, however, restricted for these methods and a relation exists between the maximum time step, Δt and the minimum element size h. This condition is known as the CFL-condition[5]. For the method of lines they look like:

$$c\frac{\Delta t}{h} \leq \mathrm{CFL} \ , \qquad (2.29)$$

with c the magnitude of the largest wave velocity of the hyperbolic system and CFL a constant depending on the time step and spatial discretization. For a DG method with polynomials of order p, an RK method of $p+1$ stages and linear flux, this CFL number is upper bounded by:

$$\mathrm{CFL} = \frac{1}{2p+1} \ , \qquad (2.30)$$

being optimal for $p = 0$ and $p = 1$. For $p \geq 2$ it is numerically shown that this upper bound is less than 5% smaller than the optimal CFL number [9]. This condition is a necessary condition for non-linear flux functions. We will use the variable Δt_{CFL} as the maximal allowed time step which satisfies the CFL condition.

A widely applied scheme in DG is the following third order strongly stability preserving

[5]Named after Richard Courant, Kurt Friedrichs, and Hans Lewy who described it in their 1928 paper [11]

(SSP) RK scheme:[6]

$$
\begin{aligned}
u^{(1)} &= u^n + \Delta t\, \mathcal{D}(u^n)\ , \\
u^{(2)} &= \frac{3}{4}u^n + \frac{1}{4}u^{(1)} + \frac{1}{4}\Delta t\, \mathcal{D}(u^{(1)})\ , \\
u^{n+1} &= \frac{1}{3}u^n + \frac{2}{3}u^{(2)} + \frac{2}{3}\Delta t\, \mathcal{D}(u^{(2)})\ ,
\end{aligned}
\tag{2.31}
$$

Strongly stability preserving means that it will maintain the stability property when the time step is restricted as would be for the forward Euler method [13]. Any convex combination of the forward Euler method would have the same property.

The use of implicit methods leads to a system of algebraic equations that has to be solved each time step. Frequently these equations are non-linear. It is clear that this is a disadvantage with respect to explicit methods, but it generally comes with the advantage of being unconditionally stable, i.e. no restrictions on the time step. For steady state problems this is convenient as we can make the time step larger as the solution is closer to steady state in order to speed up convergence to the steady state solution. Furthermore, we can test the limiter directly coupled with the DG method. For simplicity we will primarily use the backward Euler method[7]

$$
u^{n+1} = u^n + \Delta t\, \mathcal{D}(u^{n+1})\ , \tag{2.32}
$$

and the Crank-Nicolson method

$$
u^{n+1} = u^n + \frac{\Delta t}{2}\left[\mathcal{D}(u^n) + \mathcal{D}(u^{n+1})\right]\ . \tag{2.33}
$$

The backward Euler method is only first order accurate in time and the Crank-Nicolson method second order accurate. But for large time steps the Crank-Nicolson method is known to introduce numerical oscillations.

[6]Dropping the arrow and indices out the notation for clarity

[7]Again dropping the arrow and the indices for clarity

Chapter 3

Implementation details

In the previous chapter the DG formulation was presented. However, some details in the implementation were omitted. This chapter fills in those blanks by discussing the used basis functions and numerical quadrature. The non-linear algebraic equations resulting from the implicit Euler method will be discussed in Chapter 5 in combination with the limiter.

3.1 Basis functions

The DG formulation uses basis functions which are also used as test functions. There is no standard choice since no set of basis functions has only advantageous properties. One can look for orthogonality, but also for functions that can be efficiently evaluated inside the element or have an easy representation at the faces. The integrations are done on a reference element, so we will first describe the transformations of the integrals to the reference element. Due to the conservation requirements of the limiters, the equations are based on the mean value of an element. In this section we also describe the formulation of the mean value for our chosen basis functions.

3.1.1 1d: Legendre polynomials

For one dimensional problems we use Legendre polynomials as basis functions because they are orthogonal in the $L^2[-1,1]$ space, which makes the mass matrix diagonal. This leads to the choice of the interval $[-1,1]$ as our reference element on which the integration is done. The spatial coordinate in the reference element will be denoted by ξ. The first four Legendre polynomials are given by:

$$\begin{aligned} P_0(\xi) &= 1, & P_2(\xi) &= \tfrac{1}{2}(3\xi^2 - 1), \\ P_1(\xi) &= \xi, & P_3(\xi) &= \tfrac{1}{2}(5\xi^3 - 3\xi), \end{aligned}$$

and the orthogonality is given by the following relation:

$$\int_{-1}^{1} P_i\, P_j \,\mathrm{d}\xi = \frac{2}{2k+1}\delta_{ij} \ , \tag{3.1}$$

in which δ_{ij} represents the usual Kronecker delta.[1] Another pleasant feature of the Legendre polynomials is that they have nice values at the boundaries, given by:

$$\begin{aligned} P_i(-1) &= (-1)^i \ , \\ P_i(1) &= 1 \ . \end{aligned} \tag{3.2}$$

Using these basis functions, the solution in an element can be written as:

$$u_j(x)\bigg|_{I_i} = \sum_{k=0}^{p} \vec{u}_{j,k,i}\,\varphi_{k,i}(x), \qquad \varphi_{k,i}(x) = P_k\left(\frac{2(x-x_i)}{\Delta x_i}\right) \ , \tag{3.3}$$

where $\vec{u}_{j,k,i}$ stands for the coefficient of the k-th basis function in the i-th element of the j-th component. The actual calculations are more convenient to do in the interval $[-1, 1]$, because the basis functions are equal in each element and their values at the quadrature points only have to be computed once. For the source term we get a transformation coefficient in front of the integral:

$$\int_{I_i} s(\mathbf{u})_j\, \varphi_{k,i}(x)\,\mathrm{d}x = \frac{\Delta x_i}{2}\int_{-1}^{1} s(\mathbf{u})_j\, P_k(\xi)\,\mathrm{d}\xi \ .$$

The conversion of the flux integral introduces also the inverse of the transformation coefficient due the derivative with respect to the x-variable and becomes:

$$\int_{I_i} f(\mathbf{u})_j\,\frac{\mathrm{d}}{\mathrm{d}x}\big(\varphi_{k,i}(x)\big)\,\mathrm{d}x = \int_{-1}^{1} f(\mathbf{u})_j\,\frac{\mathrm{d}}{\mathrm{d}\xi}\Big(P_k(\xi)\Big)\,\mathrm{d}\xi \ .$$

The mean value of an element is easily computed and given by the coefficient of the zeroth order polynomial:

$$\begin{aligned} \bar{u}_{j,i} \equiv \frac{1}{\Delta x_i}\int_{I_i} u_j(x)\,\mathrm{d}x &= \frac{1}{\Delta x_i}\int_{I_i}\sum_{k=0}^{p}\vec{u}_{j,k,i}\,\varphi_{k,i}(x)\,1\,\mathrm{d}x \\ &= \frac{1}{\Delta x_i}\sum_{k=0}^{p}\vec{u}_{j,k,i}\int_{I_i}\varphi_{k,i}(x)\,\varphi_{0,i}(x)\,\mathrm{d}x = \frac{\Delta x_i}{2\Delta x_i}\sum_{k=0}^{p}\vec{u}_{j,k,i}\frac{2}{2k+1}\delta_{k0} = \vec{u}_{j,0,i} \ . \end{aligned} \tag{3.4}$$

3.1.2 2d: Reference triangle

For two dimensional problems we chose the triangle as our element shape. Our reference triangle is shown in Figure 3.1. We will denote the coordinates in the reference frame by ξ and η. The trial functions will be constructed such that the coefficients are equal to the mean for $p = 0$, the midpoints $p = 1$ (squares in figure) and the midpoints and vertices (circles in figure) for $p = 2$ This choice is orthogonal for $p \leq 1$ and makes it easy to calculate the minimum or maximum on an element using just the coefficients and simplifies the calculation of the coefficient for the Lax-Friedrichs flux. For $p = 0$ the trial function is:

$$\psi_1(\xi,\eta) = 1, \tag{3.5a}$$

for $p = 1$ the trial functions are:

$$\psi_1(\xi,\eta) = 1+\xi+\eta, \qquad \psi_2(\xi,\eta) = -\xi, \qquad \psi_3(\xi,\eta) = -\eta\ , \tag{3.5b}$$

and for $p = 2$ the trial functions are:

$$\begin{aligned} \psi_1(\xi,\eta) &= 1+\xi+\eta+\xi\eta, & \psi_4(\xi,\eta) &= \tfrac{1}{2}(\xi+\eta)(1+\xi+\eta), \\ \psi_2(\xi,\eta) &= -(\xi+\eta)(1+\eta), & \psi_5(\xi,\eta) &= \tfrac{1}{2}\xi(1+\xi), \\ \psi_3(\xi,\eta) &= -(\xi+\eta)(1+\xi), & \psi_6(\xi,\eta) &= \tfrac{1}{2}\eta(1+\eta)\ . \end{aligned} \tag{3.5c}$$

Since the trial equations are not orthonormal for all p we will state the mass matrices. For $p = 0$ it is given by:

$$\begin{bmatrix} 2 \end{bmatrix}, \tag{3.6a}$$

for $p = 1$:

$$\frac{2}{3}\begin{bmatrix} 1 & & \\ & 1 & \\ & & 1 \end{bmatrix}, \tag{3.6b}$$

and for $p = 2$

$$\frac{1}{90}\begin{bmatrix} 32 & 16 & 16 & -4 & & \\ 16 & 32 & 16 & & -4 & \\ 16 & 16 & 32 & & & -4 \\ -4 & & & 6 & -1 & -1 \\ & -4 & & -1 & 6 & -1 \\ & & -4 & -1 & -1 & 6 \end{bmatrix}. \tag{3.6c}$$

[1]The Kronecker delta is one if $i = j$ and zero otherwise.

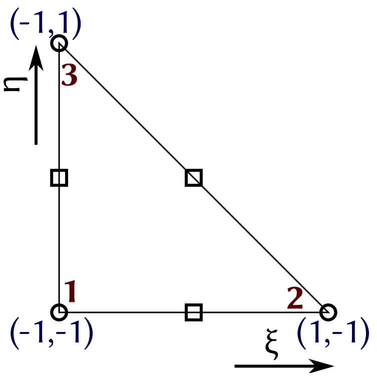


Figure 3.1: The reference triangle for 2d equations.

For the numerical computation it is desirable to transform an element to the reference triangle. If we use the labels as shown in Figure 3.1, we have the following transformation for every element i:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \frac{1}{2}\begin{bmatrix} x_2 - x_1 & x_3 - x_1 \\ y_2 - y_1 & y_3 - y_1 \end{bmatrix}\begin{bmatrix} \xi \\ \eta \end{bmatrix} + \frac{1}{2}\begin{bmatrix} x_2 + x_3 \\ y_2 + y_3 \end{bmatrix} \equiv \mathrm{A}_i \begin{bmatrix} \xi \\ \eta \end{bmatrix} + \mathrm{B}_i \ . \tag{3.7}$$

This transformation is linear for the faces and the interior of the element in contrast to quadrilaterals for which the transformation is only linear on the faces.

Using these basis functions, the solution in an element can be written as:

$$u(x)\Big|_{I_i} = \sum_{k=1}^{L} \vec{u}_{k,i}\, \varphi_{k,i}(\mathbf{x}), \qquad \varphi_{k,i}(\mathbf{x}) = \psi_k\Big(\mathrm{A}_i^{-1}\big[\mathbf{x} - \mathrm{B}_i\big]\Big) \ , \tag{3.8}$$

where L is the number of basis functions. This is not equal to $p + 1$ anymore since we have now more coordinates. The mean for $p = 0$ is simply the coefficient of the constant. For $p = 1$ and $p = 2$ it is given by:

$$\bar{u}_i = \frac{1}{3}\big[\vec{u}_{1,i} + \vec{u}_{2,i} + \vec{u}_{3,i}\big] \ . \tag{3.9}$$

The integral terms need to be converted to the reference triangle. Using the determinant of A_i (3.7) this is given for the source term by:

$$\int_{K_i} s(\mathbf{u})_j\, \varphi_{k,i}(x,y)\, \mathrm{d}x\mathrm{d}y = \det \mathrm{A}_i \int_{K_{\text{ref}}} s(\mathbf{u})_j\, \psi(\xi,\eta)\, \mathrm{d}\xi\mathrm{d}\eta \ .$$

To integrate the element flux integrals we need the derivatives of the transformation. They can be derived from (3.7) and for element i they are given by:

$$\begin{aligned} \frac{\mathrm{d}\xi}{\mathrm{d}x} &= (A_i^{-1})_{1,1} = \frac{(A_i)_{2,2}}{\det A}, & \frac{\mathrm{d}\eta}{\mathrm{d}x} &= (A_i^{-1})_{2,1} = -\frac{(A_i)_{2,1}}{\det A}, \\ \frac{\mathrm{d}\xi}{\mathrm{d}y} &= (A_i^{-1})_{1,2} = -\frac{(A_i)_{1,2}}{\det A}, & \frac{\mathrm{d}\eta}{\mathrm{d}y} &= (A_i^{-1})_{2,2} = \frac{(A_i)_{1,1}}{\det A} \ . \end{aligned}$$

This leads to the following flux integrals on the reference triangle:

$$\begin{aligned} &\int_{K_i} f(\mathbf{u})_j \frac{\mathrm{d}}{\mathrm{d}x}\Big(\varphi_{k,i}(x,y)\Big)\, \mathrm{d}x\mathrm{d}y \\ &\qquad = \int_{K_{\text{ref}}} f(\mathbf{u})_j \left[(A_i)_{2,2}\frac{\mathrm{d}}{\mathrm{d}\xi}\Big(\psi(\xi,\eta)\Big) - (A_i)_{2,1}\frac{\mathrm{d}}{\mathrm{d}\eta}\Big(\psi(\xi,\eta)\Big)\right] \mathrm{d}\xi\mathrm{d}\eta \ , \end{aligned}$$

$$\begin{aligned} &\int_{K_i} g(\mathbf{u})_j \frac{\mathrm{d}}{\mathrm{d}y}\Big(\varphi_{k,i}(x,y)\Big)\, \mathrm{d}x\mathrm{d}y \\ &\qquad = \int_{K_{\text{ref}}} g(\mathbf{u})_j \left[(A_i)_{1,1}\frac{\mathrm{d}}{\mathrm{d}\eta}\Big(\psi(\xi,\eta)\Big) - (A_i)_{1,2}\frac{\mathrm{d}}{\mathrm{d}\xi}\Big(\psi(\xi,\eta)\Big)\right] \mathrm{d}\xi\mathrm{d}\eta \ . \end{aligned}$$

The line integrals are on the reference triangle are on the standard domain for Gaussian quadrature for a one dimensional line. No extra transformation has to be done. The line integral is multiplied with $\ell/2$ where ℓ is the length of the boundary.

3.2 Numerical quadrature

As outlined in the previous chapter, we will calculate the integrals via Gauss quadrature rules. The Gauss quadrature rules are, like many other quadrature methods, a weighted sum over specific points – also called abscissae – of the integrand. In one dimension the quadrature is done on the reference interval $[-1, 1]$ and is accurate for polynomials up to order $2n - 1$ for n integration points. In formula:

$$\int_{-1}^{1} f(\xi)\,\mathrm{d}\xi \approx \sum_{i=1}^{n} f(x_i)\,w_i\ . \tag{3.10}$$

The formulas for specific abscissae x_i and weights w_i, can be found in numerous places, for example in Abramowitz and Stegun [1] or Wikipedia. We used a five point approximation for the integrals in one dimension and for the line integrals in two dimensions They are given up to double precision in Table 3.1

A two dimensional integral in a non-square shape is more difficult. For a square the abscissae and weights can be used of the one dimensional case in each direction. In a triangle the abscissae and weights can be derived using the same principles as in the one dimensional case. The resulting equations are, however, difficult to solve. We used the table from Solin et al. [28] as displayed in Table 3.2, which is accurate for polynomials up to fifth order and is given in double precision. The position of the abscissae can be seen in Figure 3.2. The formula for applying the Gauss quadrature is given by:

$$\int_{\Delta} f(\xi, \eta)\,\mathrm{d}\xi\mathrm{d}\eta \approx \sum_{i=1}^{n} f(x_i, y_i)\,w_i\ . \tag{3.11}$$

Table 3.1: Numerical quadrature abscissae and weights for the interval $[-1, 1]$ order $p = 9$, [28]

x_i	w_i (weight)
-0.906179845938664	0.236926885056189
-0.538469310105683	0.478628670499366
0	0.568888888888889
0.538469310105683	0.478628670499366
0.906179845938664	0.236926885056189

Table 3.2: Numerical quadrature abscissae and weights for the reference triangle order $p = 5$, [28]

x_i	y_i	w_i (weight)
-0.333333333333333	-0.333333333333333	0.450000000000000
-0.059715871789770	-0.059715871789770	0.264788305577012
-0.059715871789770	-0.880568256420460	0.264788305577012
-0.880568256420460	-0.059715871789770	0.264788305577012
-0.797426985353088	-0.797426985353088	0.251878361089654
-0.797426985353088	0.594853970706174	0.251878361089654
0.594853970706174	-0.797426985353088	0.251878361089654

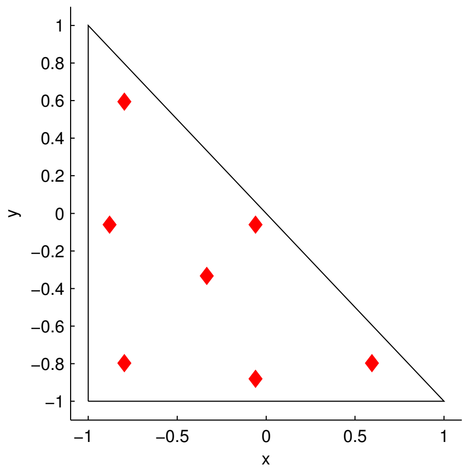


Figure 3.2: Numerical quadrature abscissae for the reference triangle order $p = 5$.

Chapter 4

WBAP limiter

4.1 Introduction

The DG method introduces non-physical oscillations near discontinuities or steep gradients when solving the hyperbolic equations with high resolution schemes. An important method in suppressing these oscillations is the use of limiters. Limiters use information from neighboring elements to limit the gradients in the cell to physical values.

An important class of limiters for conservation laws are slope limiters with total variation diminishing property. The total variation is given analytically by [30]:

$$\mathrm{TV}(u) = \int_\Omega |u'(x)| \mathrm{d}x \ , \tag{4.1}$$

which for convergence purposes can be viewed at fixed times $t = t_n$ and is given on a mesh by:

$$\mathrm{TV}(u^n) = \sum_i |u^n_{i+1} - u^n_i| \ . \tag{4.2}$$

The total variation diminishing property is then given by:

$$\mathrm{TV}(u^{n+1}) \leq \mathrm{TV}(u^n), \quad \forall n \ . \tag{4.3}$$

If the TVD property holds, no new local extrema are generated and existing local minima and maxima do not decrease, respectively, increase. More importantly, it also implies no numerical oscillations. The admissible region for slope limiters which guarantees that they satisfy the TVD property is the Sweby region [29]. Minmod and superbee limiters are well known examples.

However, schemes with a TVD property are only second order accurate – an improvement on the first order accuracy due to Godunov's theorem – and have to be applied dimension by dimension on higher dimensional problems and unstructured grids. The WBAP limiter tries to overcome these problems. Our interest in this limiter lies in the fact that it is smooth, which is expected to be helpful in the iterative method used to solve the algebraic equations resulting from an implicit time integration method.

4.2 Definition of the WBAP

The WBAP limiter is introduced by Li et al. [22] for finite volume schemes on unstructured grids. It is based on the *biased averaging procedure* (BAP) of Choi and Liu [6] because this method has a number of attractive properties: very simple, efficient, parameter free, differentiable and applicable on unstructured grids. They improved the method in [6] by using the weight function in the averaging procedure and therefore call it the *weighted biased averaging procedure* (WBAP). Additionally, they introduced a free parameter to control the dissipation and introduced a way to preserve self-similarity.

The limiter is introduced in the finite volume framework and is based on the gradients computed by a central difference scheme:

$$\bar{\sigma}_i = \frac{1}{2}\left[\frac{\bar{u}_{i+1}-\bar{u}_i}{\frac{1}{2}(\Delta x_i+\Delta x_{i+1})}+\frac{\bar{u}_i-\bar{u}_{i-i}}{\frac{1}{2}(\Delta x_{i-1}+\Delta x_i)}\right] = \frac{\bar{u}_{i+1}-\bar{u}_i}{\Delta x_i+\Delta x_{i+1}}+\frac{\bar{u}_i-\bar{u}_{i-i}}{\Delta x_{i-1}+\Delta x_i}\ , \tag{4.4}$$

where $\bar{u}$ denotes the mean value as computed in the finite volume scheme. For uniform grids it simplifies greatly to

$$\bar{\sigma}_i = \frac{1}{2\Delta x}\left[\bar{u}_{i+1}-\bar{u}_{i-i}\right]\ . \tag{4.5}$$

Li et al. proposed the limiter in two versions, the first one is mainly of theoretical use to prove the TVD property in the MUSCL framework and to show why the WBAP limiter is capable of removing oscillations. This proof does not extend to the second version. The second version is used in practice because it is easier to extend to multiple dimensions. The WBAP limiter is given as:

$$L(\bar{\sigma}_0,\bar{\sigma}_1,\bar{\sigma}_2,\cdots,\bar{\sigma}_J) = \bar{\sigma}_0\cdot W(1,\tfrac{\bar{\sigma}_1}{\bar{\sigma}_0},\tfrac{\bar{\sigma}_2}{\bar{\sigma}_0},\cdots,\tfrac{\bar{\sigma}_J}{\bar{\sigma}_0})\ , \tag{4.6}$$

with $\bar{\sigma}_0$, the reconstructed gradient in the element and $\bar{\sigma}_j$, $j=1,\cdots,J$, the gradients of the neighboring elements. The easiness in extending the limiter to multiple dimensions lies in the fact that one can just add gradients to the limiter for each extra neighboring element. The WBAP limiter compares the gradients divided by its first argument, the unlimited gradient, and limits the ratio's in W, which is given by:

$$W(\theta_0,\theta_1,\theta_2,\cdots,\theta_J) = B^{-1}\left(\sum_{j=0}^{J}\omega_j B(\theta_j)\right)\ , \tag{4.7}$$

where B is the biased function and ω_j, $j = 0, \cdots, J$ are the weights. They consider the following bias and weight function:

$$B(\chi) = \frac{\chi}{\sqrt{\epsilon^2 + \chi^2}}, \qquad \omega_j = \frac{\alpha_j}{\sum_{j=0}^{J} \alpha_j}, \tag{4.8a}$$

$$B^{-1}(\chi) = \frac{\epsilon\chi}{\sqrt{1 - \chi^2}}, \qquad \alpha_j = \begin{cases} \frac{n}{B(\theta_j)^2 + \delta} & \text{if } j = 0 \\ \frac{1}{B(\theta_j)^2 + \delta} & \text{otherwise} \end{cases}, \tag{4.8b}$$

with ϵ a positive parameter, which is a measure for the dissipativity of the limiter and δ a small positive number to prevent division by zero. We will use throughout this report a value of 10^{-10} for δ. The n in the formula for α_j is the weight of the unlimited gradient – i.e. the gradient which would be used without limiting – of an element relatively to its neighboring elements. This n is adjustable to let the unlimited gradient weigh more or less and thus alter the strength of the limiter. In the case of $n \to \infty$ the limiter does not act at all.

4.2.1 Different versions

Together with the previous definition, two limited versions of (4.7) are introduced in [22] to reduce the complexity of the formula's and the associated computational cost and numerical roundoff errors. The ϵ in (4.8) can have a value in $(0, \infty)$. Based on numerical results it has a positive correlation with the dissipative property of the limiter. Limiting ϵ to the lower and upper bound of this range, give two different versions:

$$W^{L1}(1, \theta_1, \cdots, \theta_J) = \lim_{\epsilon \to 0} W(1, \theta_1, \cdots, \theta_J) = \begin{cases} \sqrt{\frac{n+J}{n + \sum_{j=1}^{J} 1/\theta_j^2}} & \text{if } \theta_1, \cdots, \theta_J > 0 \\ 0 & \text{otherwise} \end{cases}, \tag{4.9}$$

$$W^{L2}(1, \theta_1, \cdots, \theta_J) = \lim_{\epsilon \to \infty} W(1, \theta_1, \cdots, \theta_J) = \frac{n + \sum_{j=1}^{J} 1/\theta_j}{n + \sum_{j=1}^{J} 1/\theta_j^2}. \tag{4.10}$$

It is easily seen that the first case, (4.9), is not smooth anymore for the lines $\theta_i = 0$. It is, however, continuous in zeroth order and continuously differentiable for all other points. In addition to those limiter versions, they propose an alternative for the WBAP-L2 limiter, using the same zero case as with the WBAP-L1.

$$W^{L2a}(1, \theta_1, \cdots, \theta_J) = \begin{cases} W^{L2}(1, \theta_1, \cdots, \theta_J) & \text{if } \theta_1, \cdots, \theta_J > 0 \\ 0 & \text{otherwise} \end{cases}. \tag{4.11}$$

This is motivated by the Sweby region [29] as for $x < 0$ the limiter should be zero. In Figure 4.1 the WBAP-limiters are shown for $J = 1$ and $n = 1$.[1] One can see from the figure that the WBAP-L1 and the alternative WBAP-L2 fall in the Sweby region and hence satisfy the TVD property. However, for $J > 1$, as used even for the one dimensional setup, only a TVD-like property can be proven. Namely, a maximum principle that the new values are contained within the values of a stencil of five neighboring elements.

[1]The alternative WBAP-L2 limiter is just the WBAP-L1 limiter in the second quadrant and the WBAP-L2 limiter in the first.

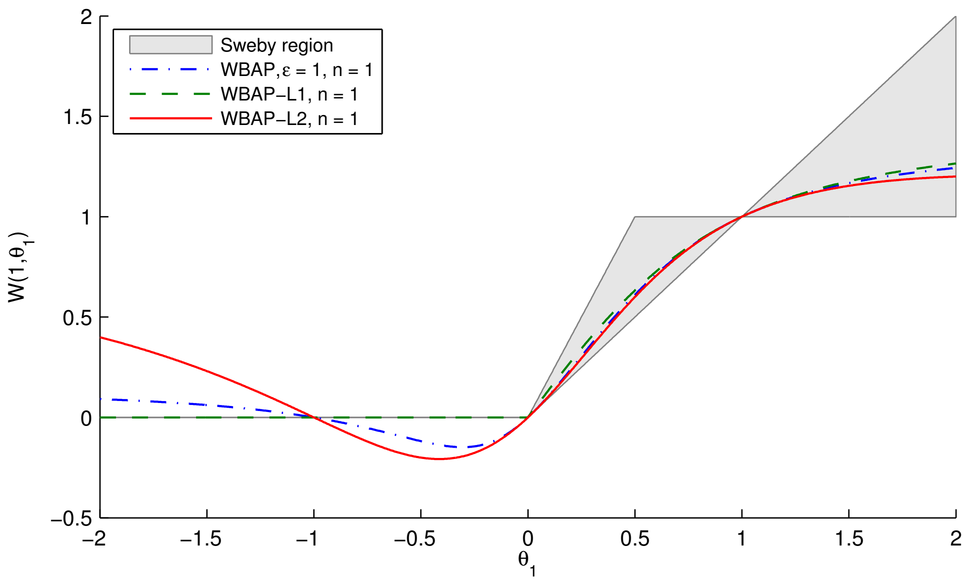


Figure 4.1: The different WBAP-limiters for $J = 1$ with Sweby-TVD region for comparison.

4.2.2 Smoothness

In [22], the WBAP limiter W is stated to be smooth, i.e. continuously differentiable. But the function L is not smooth when $\bar{\sigma}_j = 0$ for all $j \in 0, \cdots, J$. The problem lies in the fact that all arguments are divided by $\bar{\sigma}_0$ in the argument of the limiter function W and its result is multiplied by $\bar{\sigma}_0$ again. If we look at the first derivative in the second term in the simple case of $J = 1$:

$$\frac{\partial}{\partial \bar{\sigma}_1} L(\bar{\sigma}_0, \bar{\sigma}_1) = \frac{\partial}{\partial \bar{\sigma}_1} \left[\bar{\sigma}_0 W(1, {}^{\bar{\sigma}_1}\!/_{\bar{\sigma}_0}) \right] = \mathrm{D}_2 W(1, {}^{\bar{\sigma}_1}\!/_{\bar{\sigma}_0}) \;, \tag{4.12}$$

we can show that the limit of $(0, 0)$ is undetermined. First, we transform the problem into polar coordinates:

$$\bar{\sigma}_0 = r \cos \varphi \qquad\qquad \bar{\sigma}_1 = r \sin \varphi \;,$$

which makes (4.12):

$$\frac{\partial}{\partial \bar{\sigma}_1} L(\bar{\sigma}_0, \bar{\sigma}_1) = \mathrm{D}_2 W(1, \tan \varphi) \;. \tag{4.13}$$

We see that the derivative is independent of the Euclidean distance, $r = |(\bar{\sigma}_0, \bar{\sigma}_1)|$. However, $(0, 0)$ is equivalent with $r = 0$ and any φ. So if this derivative $\mathrm{D}_2 L$ depends on φ, the limit in $(0, 0)$ cannot exist, and hence the derivative cannot be continuous at that point. Looking at what $\varphi = \arctan {}^{\bar{\sigma}_1}\!/_{\bar{\sigma}_0}$ represents, it is clearly from the purpose of a limiter – changing the ratio of σ_0 and σ_1 – that it will not be independent of φ. It can easily be extended to $J > 2$ by setting all $\bar{\sigma}_j = 0$ for $j > 1$ and repeat the argument.

The non-smoothness for the limiters can cause problems in solving our algebraic equations. Due to the non-linearity of the spatial DG operator and limiter, methods will be used which need the Jacobian of the function that has to be solved. The problem is that the Jacobian due to the non-smoothness is not defined. We will use the generalized Jacobian in the sense of Qi [26] to solve the problem. In the generalized Jacobian we formally define the function to be zero at those points or planes along with their derivatives.

4.3 Application to the Discontinuous Galerkin method

In contrast with the finite volume method, the DG method does not have to reconstruct a gradient from the mean values of its neighboring elements because it describes a solution on the entire element. However, we cannot compare the gradient of the DG method with the gradients inside neighboring elements since none of those gradients can be excluded from being distorted with numerical oscillations. To limit gradients in an element we need to construct gradients between the elements using the mean values, which are free from numerical oscillations.

For the one dimensional case we will use gradients computed by using the means of two neighboring elements. For the two dimensional case a reconstruction method from [22] is used to apply the limiter in a finite volume method. The result is that our modification will only use higher order DG information in the first argument of the WBAP limiter.

In the above sections the limiter was explained in a scalar approach. For a system of equations, there are two approaches. The first is to limit the conservative variables, which means all conservative variables will be limited independently and the equations of the scalar case can be copied for each variable. The second is to limit the characteristic variables. A transformation to and from the conservative variables is needed and is given by the left eigenvectors of the flux Jacobian and its inverse. This approach is usually more successful in limiting, but we will use the first method since it simplifies our equations. Simple equations are better for direct application in the DG method, which is more an issue than its limiting capability. If it works for the conserved variables it can of course be extended to the characteristic variables.

The following subsections will describe how the limiter is applied in the DG method for scalars. For convenient notation we drop the index of the component in the equations and coefficients, and describe the scalar case.

4.3.1 One dimensional application

The gradient of the DG method at an element face is calculated via the value of the solution at the faces subtracted by its mean value.

$$\tilde{u}_i^{(1)} = \frac{u(x_{i+1/2}) - \bar{u}_i}{\frac{1}{2}\Delta x_i}, \qquad \tilde{u}_i^{(2)} = \frac{\bar{u}_i - u(x_{i-1/2})}{\frac{1}{2}\Delta x_i}, \tag{4.14}$$

By subtracting the mean value we assure conservation of the mean value of the solution. The gradients with the neighboring elements are computed by a finite difference relation using the mean value in the elements. Using the operators Δ_+ and Δ_-, defined as:

$$\Delta_+\bar{u}_i = \frac{\bar{u}_{i+1} - \bar{u}_i}{\frac{1}{2}(\Delta x_i + \Delta x_{i+1})}, \qquad \Delta_-\bar{u}_i = \frac{\bar{u}_i - \bar{u}_{i-1}}{\frac{1}{2}(\Delta x_{i-1} + \Delta x_i)}, \tag{4.15}$$

we can write the limiting procedure as:

$$\tilde{u}_i^{(1,\text{mod})} = L(\tilde{u}^{(1)}, \Delta_+\bar{u}_i, \Delta_-\bar{u}_i), \qquad \tilde{u}_i^{(2,\text{mod})} = L(\tilde{u}^{(2)}, \Delta_+\bar{u}_i, \Delta_-\bar{u}_i)\ . \tag{4.16}$$

Using this limited gradient the coefficients of the basis functions are recovered using the same relation as (4.14), resulting in:

$$u(x_{i+1/2}) = \frac{1}{2}\Delta x_i\big(\tilde{u}^{(1,\text{mod})} - \bar{u}(x_i)\big), \qquad u(x_{i-1/2}) = \frac{1}{2}\Delta x_i\big(\bar{u}(x_i) - \tilde{u}^{(2,\text{mod})}\big)\ . \tag{4.17}$$

Note that for $k \leq 2$ the recovery is unique. For $k > 2$ we have extra freedom in the recovery.

To solve the algebraic equations originating from the implicit time integration we would like an explicit formula for the limiting procedure. This will be needed for the Jacobian of the operator. Using our Legendre polynomials as basis functions, the values at the boundaries are given by:

$$u(x_{i+1/2}) = \vec{u}_{0,i} + \vec{u}_{1,i} + \cdots + \vec{u}_{p,i}, \qquad u(x_{i-1/2}) = \vec{u}_{0,i} - \vec{u}_{1,i} + \cdots + (-1)^p\,\vec{u}_{p,i}\ .$$

This results in an explicit formula which is for $k = 1$

$$\vec{u}_{1,i}^{\text{lim}} = \frac{\Delta x}{2} L\Big(\tfrac{2}{\Delta x}\vec{u}_{1,i},\ \Delta_+\vec{u}_{0,i},\ \Delta_-\vec{u}_{0,i}\Big), \tag{4.18}$$

and for $k = 2$:

$$\vec{u}_{1,i}^{\text{lim}} = \frac{\Delta x}{4}\Big(L_A + L_B\Big), \qquad \vec{u}_{2,i}^{\text{lim}} = \frac{\Delta x}{4}\Big(L_A - L_B\Big), \tag{4.19a}$$

with L_A and L_B being defined as:

$$L_A = L\Big(\tfrac{2}{\Delta x}(\vec{u}_{1,i} + \vec{u}_{2,i}), \Delta_+\vec{u}_{0,i}, \Delta_-\vec{u}_{0,i}\Big), \tag{4.19b}$$

$$L_B = L\Big(\tfrac{2}{\Delta x}(\vec{u}_{1,i} - \vec{u}_{2,i}), \Delta_+\vec{u}_{0,i}, \Delta_-\vec{u}_{0,i}\Big)\ . \tag{4.19c}$$

4.3.2 Two dimensional application

In two dimensions the limiting procedure is more complicated, as almost all things are in dimensions higher than one. We have to compare the gradients in two directions, but because of the triangulation and the use of our reference element, the elements are rotated relative to each other and the gradients are given in different directions than the x,y relative to the domain. This will be addressed in the first part of this section. In the second part the reconstruction using information from the neighbors will be addressed. Both parts will end in explicit formula which will be combined such that the derivatives can easily be derived in Chapter 5.

Converting a local element gradient to global gradient

We will use the absolute x and y position of the grid to limit the solution in two dimensions, Therefore, we first have to project our solution in the trial space into the linear space $Z = (1/2, \xi + 1/3, \eta + 1/3)$ relative to the reference triangle. The mean in this basis is given *only* by the coefficient of the first basis function in our reference triangle. Using (3.5) we can formulate the transformation matrix $\hat{W}$ and its inverse as:

$$\hat{W} = \begin{bmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ 1 & -1 & 0 \\ 1 & 0 & -1 \end{bmatrix} , \qquad \hat{W}^{-1} = \begin{bmatrix} 1 & 1/3 & 1/3 \\ 1 & -2/3 & 1/3 \\ 1 & 1/3 & -2/3 \end{bmatrix} , \tag{4.20}$$

We only need the transformation to and from the last two components. Therefore we define that by W and the pseudo inverse W^{-1}.

$$W = \begin{bmatrix} 1 & -1 & 0 \\ 1 & 0 & -1 \end{bmatrix} , \qquad W^{-1} = \begin{bmatrix} 1/3 & 1/3 \\ -2/3 & 1/3 \\ 1/3 & -2/3 \end{bmatrix} , \tag{4.21}$$

We do not loose any information since we will add the mean value after the limiter is applied. In this way the mean value is conserved which is a natural requirement for a limiter.

Using this basis the transformation to the absolute x and y components is done using the inverse transpose of the transform matrix (3.7). The total transform is given by:

$$\begin{bmatrix} \sigma_x \\ \sigma_y \end{bmatrix} = \mathrm{A}_i^{-T} \cdot \mathrm{W} \cdot \vec{u}_i \ , \tag{4.22}$$

and going back is done using the inverse and adding the mean value:

$$\vec{u}_i = \mathrm{W}^{-1} \cdot \mathrm{A}_i^T \cdot \begin{bmatrix} \sigma_x \\ \sigma_y \end{bmatrix} + \bar{u}_i \ . \tag{4.23}$$

The limiter is applied to the x and y gradient separately. Introducing $\widetilde{L}$ as a 2×2 operator defined by:

$$\widetilde{L}([x_1, y_1]^{\mathrm{T}}, \cdots, [x_n, y_n]^{\mathrm{T}}) = \begin{bmatrix} L(x_1, \cdots, x_n) \\ L(y_1, \cdots, y_n) \end{bmatrix} \ , \tag{4.24}$$

we describe this mathematically and the procedure can be given for an element i as:

$$\vec{u}_{:,i}^{\mathrm{lim}} = \mathrm{W}^{-1} \cdot \mathrm{A}^T \cdot \widetilde{L}\Big(\mathrm{A}_i^{-T} \cdot \mathrm{W} \cdot \vec{u}_{:,i}, \cdots\Big) + \bar{u}_{:,i} \ . \tag{4.25}$$

Computing gradients using neighboring elements

To calculate the various gradients used in the limiter from information in neighboring elements we will use a method from finite volume schemes. For every element i we define a so-called first order reconstruction polynomial by:

$$\tilde{u}_i(\mathbf{x} - \mathbf{x}_i) = \bar{u}_i + \nabla u_i \cdot (\mathbf{x} - \mathbf{x}_i) \ , \tag{4.26}$$

in which $\bar{u}_i$ is the mean value of the element, $\mathbf{x}_i$ the centroid and ∇u_i are the two coefficients to be determined. For a linear reconstruction it is sufficient to choose the neighboring elements, labeled $S_i = j_1, j_2, j_3$, as the reconstruction stencil. The best reconstruction preserves the cell average on every cell of the stencil. This leads to the equations:

$$\frac{1}{|K_j|}\int_{K_i} \tilde{u}_i(\mathbf{x}-\mathbf{x}_i)\,\mathrm{d}x\mathrm{d}y = \bar{u}_j, \quad j \in S_i \ , \tag{4.27}$$

with $|K_i|$ the area of element i. The equations constitute an overdetermined system of equations for elements not connected to the boundary. For elements connected to the boundary, however, we have an equal number of unknowns and equations or an underdetermined problem. In those cases the system of equations is too stiff and elements with a common vertex in the element considered are added to the system. We will designate the set used for the reconstruction with $\hat{S}_i \supset S_i$ to prevent unnecessary separation of interior and boundary elements. Common practice is to give neighboring elements a larger weight when they are closer to the element where the data is limited. Therefore we introduce ω_i as weight function defined as the inverse distance between the centers:

$$\omega_{ij} = \frac{1}{|\mathbf{x}_j - \mathbf{x}_i|} \ . \tag{4.28}$$

If we move the given mean value $\bar{u}_i$ to the right hand side in (4.27), we obtain the final reconstruction equations:

$$\frac{\omega_{ij}}{|K_j|}\int_{K_i} \nabla u_i \cdot (\mathbf{x}-\mathbf{x}_i)\,\mathrm{d}x\mathrm{d}y = \omega_{ij}\big(\bar{u}_j - \bar{u}_i\big), \quad j \in \hat{S}_i \ . \tag{4.29}$$

The weight is on both sides since we will convert the overdetermined system to a minimization problem and the weights are to be considered with respect to the residue of the equation.

The overdetermined system of equations can easily be solved by the method of least squares. Since (4.29) is linear we can write it in matrix form as:

$$\min||\mathrm{B}_i \cdot \nabla u_i - \beta_i||_{L_2} \ ,$$

with B_i and β_i being defined as:

$$\begin{aligned}(B_i)_{j,1} &= \frac{\omega_{ij}}{|K_j|}\int_{K_i} \big(\nabla u_i\big)_1 (x - x_i)\,\mathrm{d}x\mathrm{d}y,\\ (B_i)_{j,2} &= \frac{\omega_{ij}}{|K_j|}\int_{K_i} \big(\nabla u_i\big)_2 (y - y_i)\,\mathrm{d}x\mathrm{d}y,\\ (\beta_i)_j &= \omega_{ij}\big(\bar{u}_{(\hat{S}_i)_j} - \bar{u}_i\big) \ .\end{aligned}$$

Solving a minimization problem is equal to solving the following normal equations

$$\mathrm{B}_i^{\mathrm{T}} \cdot \mathrm{B}_i \cdot \nabla u_i = \mathrm{B}_i^{\mathrm{T}} \cdot \beta_i \ ,$$

which has as solution

$$\nabla u_i = \mathrm{R}_i^{-1}\mathrm{Q}_i^{\mathrm{T}} \cdot \beta_i \ , \tag{4.30}$$

with Q_i and R_i being the QR decomposition of B_i. Note that the left hand side of (4.29) does not depend on $\vec{u}$, but only on the chosen mesh. So with a fixed mesh the other gradients are a simple matrix multiplication with the means of the elements of $\hat{S}_i$.

Applying the limiter

The limiter is applied only to elements that share a side, even if more neighboring elements were used in the reconstruction process. This set is still denoted by S_i. With (4.30) we can fill the second and higher arguments into (4.25) and give a complete expression of how to apply the limiter

$$\vec{u}_i^{\mathrm{lim}} = \mathrm{W}^{-1} \cdot \mathrm{A}_i^T \cdot \widetilde{L}\Big(\mathrm{A}_i^{-T} \cdot \mathrm{W} \cdot \vec{u}_i,\ \mathrm{R}_{(S_i)_1}^{-1} \mathrm{Q}_{(S_i)_1}^{\mathrm{T}} \cdot \beta_{(S_i)_1}, \cdots\Big) + \bar{u}_i \ . \tag{4.31}$$

4.3.3 Time integration with limited DG discretization

The application of the limiter with an explicit Runge-Kutta method is trivial. This can be done after each time step or stage of the method. For an implicit method, the limiter has to be integrated into the system of algebraic equations. This changes (2.32) for the implicit Euler time integration method to[2]

$$u^{n+1} = \mathcal{L}\Big(u^n + \Delta t\, \mathcal{D}(u^{n+1})\Big) \ , \tag{4.32}$$

and for the Crank-Nicolson time integration method (2.33) to

$$u^{n+1} = \mathcal{L}\Big(u^n + \frac{\Delta t}{2} \left[\mathcal{D}(u^n) + \mathcal{D}(u^{n+1})\right]\Big) \ , \tag{4.33}$$

where $\mathcal{L}$ is the limiter operator which is simply the explicit formula of the limiters, (4.18), (4.19) in one dimension and (4.31) in two dimensions, applied for each component of the hyperbolic system and to each element in the mesh.

4.4 Derivatives of the WBAP

Since we need to take the derivative of the limiter operator in the Newton method used to solve the non-linear algebraic equations, we will also need the derivatives of the WBAP limiters. This section will give the equations of the derivatives. The WBAP-L1 limiter is given explicitly by:

$$L_1(\sigma_0, \sigma_1, \cdots, \sigma_l) = \begin{cases} \sqrt{\dfrac{n+l}{n/\sigma_0^2 + 1/\sigma_1^2 + \cdots + 1/\sigma_l^2}} & \text{if } \sigma_0, \sigma_1, \cdots, \sigma_l > 0, \\ -\sqrt{\dfrac{n+l}{n/\sigma_0^2 + 1/\sigma_1^2 + \cdots + 1/\sigma_l^2}} & \text{if } \sigma_0, \sigma_1, \cdots, \sigma_l < 0, \\ 0 & \text{otherwise,} \end{cases} \tag{4.34}$$

[2]Dropping the arrow and indices out the notation for clarity

for which the derivatives for the non-zero case are given by:

$$\mathrm{D}_i\, L_1(\sigma_0, \sigma_1, \cdots, \sigma_l) = \frac{n_i}{(n+l)\sigma_i^3} L_1(\sigma_0, \sigma_1, \cdots, \sigma_l)^3, \tag{4.35}$$

with

$$n_i = \begin{cases} n & \text{if } i = 1, \\ 1 & \text{otherwise.} \end{cases} \tag{4.36}$$

The WBAP-L2 limiter is given explicitly by:

$$L_2(\sigma_0, \sigma_1, \cdots, \sigma_l) = \begin{cases} 0, & \sigma_0 = 0 \vee \sigma_1 = 0 \vee \cdots \vee \sigma_l = 0, \\ \dfrac{n/\sigma_0 + 1/\sigma_1 + \cdots + 1/\sigma_l}{n/\sigma_0^2 + 1/\sigma_1^2 + \cdots + 1/\sigma_l^2}, & \text{otherwise,} \end{cases} \tag{4.37}$$

for which the derivatives for the non-zero case are given by:

$$\begin{aligned} \mathrm{D}_i\, L_2(\sigma_0, \sigma_1, \cdots, \sigma_l) = {} & \frac{2n_i}{\sigma_i^3} \frac{n/\sigma_0 + 1/\sigma_1 + \cdots + 1/\sigma_l}{\left(n/\sigma_0^2 + 1/\sigma_1^2 + \cdots + 1/\sigma_l^2\right)^2} \\ & - \frac{n_i}{\sigma_i^2} \frac{1}{n/\sigma_0^2 + 1/\sigma_1^2 + \cdots + 1/\sigma_l^2} \,. \end{aligned} \tag{4.38}$$

The original WBAP limiter is given explicitly by:

$$L(\sigma_0, \sigma_1, \cdots, \sigma_l) = \frac{\epsilon\, \sigma_0\, B}{\sqrt{1 - B^2}} \,, \tag{4.39a}$$

with

$$B = \frac{\frac{n}{C_0\sqrt{\epsilon^2+1}} + \frac{\sigma_1/\sigma_0}{C_1\sqrt{\epsilon^2+\left(\frac{\sigma_1}{\sigma_0}\right)^2}} + \cdots + + \frac{\sigma_l/\sigma_0}{C_l\sqrt{\epsilon^2+\left(\frac{\sigma_l}{\sigma_0}\right)^2}}}{\frac{n}{C_0} + \frac{1}{C_1} + \cdots + \frac{1}{C_l}} \,, \tag{4.39b}$$

and

$$C_0 = \delta + \frac{1}{\epsilon^2 + 1}, \qquad C_i = \delta + \frac{\left(\frac{\sigma_i}{\sigma_0}\right)^2}{\epsilon^2 + \left(\frac{\sigma_i}{\sigma_0}\right)^2}, \quad i > 1 \,. \tag{4.39c}$$

The derivatives of this function are quite complicated. They are:

$$\mathrm{D}_1\, L_2(\sigma_0, \sigma_1, \cdots, \sigma_l) = \epsilon\, \frac{B - B^3 - \frac{\sigma_1}{\sigma_0}\frac{\partial B}{\partial \sigma_1} - \cdots - \frac{\sigma_l}{\sigma_0}\frac{\partial B}{\partial \sigma_l}}{(1 - B^2)^{3/2}} \,, \tag{4.40a}$$

$$\mathrm{D}_i\, L_2(\sigma_0, \sigma_1, \cdots, \sigma_l) = \frac{\epsilon\, \frac{\partial B}{\partial \sigma_i}}{(1 - B^2)^{3/2}}, \quad i > 1 \,, \tag{4.40b}$$

with

$$\frac{\partial B}{\partial \sigma_i} = \left[\frac{1 - \frac{\sigma_i}{\sigma_0} \frac{\frac{\mathrm{d}C_i}{\mathrm{d}\sigma_i}}{C_i} - \frac{(\sigma_i/\sigma_0)^2}{\epsilon^2 + \left(\frac{\sigma_i}{\sigma_0}\right)^2}}{C_i \sqrt{\epsilon^2 + \left(\frac{\sigma_i}{\sigma_0}\right)^2}} + B \frac{\frac{\mathrm{d}\sigma_i}{\mathrm{d}C_i}}{C_i^2} \right] \Bigg/ \left\{ \frac{n}{C_0} + \frac{1}{C_1} + \cdots + \frac{1}{C_l} \right\} \tag{4.40c}$$

$$\frac{\mathrm{d}C_i}{\mathrm{d}\sigma_i} = 2 \frac{\sigma_i/\sigma_0}{\epsilon^2 + \left(\frac{\sigma_i}{\sigma_0}\right)^2} \left\{ 1 - \frac{(\sigma_i/\sigma_0)^2}{\epsilon^2 + \left(\frac{\sigma_i}{\sigma_0}\right)^2} \right\} . \tag{4.40d}$$

Chapter 5

Non-linear solvers

From the previous chapters we were left with non-linear equations that we have to solve each time step, equations (2.32), (4.32) for the backward Euler method and (2.33), (4.33) for the Crank-Nicolson method. This chapter deals with the (damped) Newton method to solve these non-linear algebraic equations. Several methods exist to solve non-linear equations. Well-known methods are the secant method, the Newton method, quasi-Newton methods, conjugate gradient methods (CG), steepest descent, Levenberg-Marquadt and many others. All these methods are iterative methods, i.e. each iterative step refines the solution and the process is repeated until a certain tolerance is reached.

We use the damped Newton method which is a variant of the Newton method. The Newton method uses a second order approximation and is easy to implement. The second order approximation results in second order local convergence in the number of iterations. The damped Newton method is a modification to the Newton method that damps the step size when the solution is not in the area of local convergence and has good results for global convergence.

The Newton method needs the Jacobian of the operators $\mathcal{D}$ and $\mathcal{L}$. These will be derived first. Subsequently, the Newton method is introduced and the damped Newton modification to deal with the limited convergence radius of the original Newton method.

5.1 Jacobian

5.1.1 One dimensional DG operator

To calculate $\mathcal{D}^{(1)}$ we will use again the spatial DG operator per element i and component j, $\mathcal{D}^{(1)}_{i,j}$ in (2.28), as starting point. The neighbors of an interior element are just $(i-1)$ and $(i+1)$, which limits the derivative to the components and basis functions of elements $(i-1, i, i+1)$.

They are given by:

$$\begin{aligned}
&\frac{\partial}{\partial \vec{u}_{b,a,i}}\Big(\mathrm{M}_i \cdot \mathcal{D}^{(1)}_{i,j}(\vec{u})\Big) \\
&\quad = \int_{I_i} \big[\nabla \mathbf{s}(\mathbf{u})\big]_{b,j}\, \varphi_{:,i}\, \varphi_{a,i}\, \mathrm{d}x + \int_{I_i} \big[\nabla \mathbf{f}(\mathbf{u})\big]_{b,j} \frac{\mathrm{d}}{\mathrm{d}x}\Big(\varphi_{:,i}\Big)\, \varphi_{a,i}\, \mathrm{d}x \\
&\qquad - \Big[\nabla_1 \hat{\mathbf{f}}\big(\mathbf{u}(x^-_{i+1/2}), \mathbf{u}(x^+_{i+1/2})\big)\Big]_{b,j} \varphi_{:,i}(x^-_{i+1/2})\, \varphi_{a,i}(x^-_{i+1/2}) \\
&\qquad + \Big[\nabla_2 \hat{\mathbf{f}}\big(\mathbf{u}(x^-_{i-1/2}), \mathbf{u}(x^+_{i-1/2})\big)\Big]_{b,j} \varphi_{:,i}(x^+_{i-1/2})\, \varphi_{a,i}(x^+_{i-1/2}),
\end{aligned} \tag{5.1a}$$

$$\begin{aligned}
&\frac{\partial}{\partial \vec{u}_{b,a,i+1}}\Big(\mathrm{M}_i \cdot \mathcal{D}^{(1)}_{i,j}(\vec{u})\Big) = \\
&\qquad - \Big[\nabla_2 \hat{\mathbf{f}}\big(\mathbf{u}(x^-_{i+1/2}), \mathbf{u}(x^+_{i+1/2})\big)\Big]_{b,j} \varphi_{:,i}(x^-_{i+1/2})\, \varphi_{a,i+1}(x^+_{i+1/2}),
\end{aligned} \tag{5.1b}$$

$$\begin{aligned}
&\frac{\partial}{\partial \vec{u}_{b,a,i-1}}\Big(\mathrm{M}_i \cdot \mathcal{D}^{(1)}_{i,j}(\vec{u})\Big) = \\
&\qquad \Big[\nabla_1 \hat{\mathbf{f}}\big(\mathbf{u}(x^-_{i+1/2}), \mathbf{u}(x^+_{i+1/2})\big)\Big]_{b,j} \varphi_{:,i}(x^+_{i-1/2})\, \varphi_{a,i-1}(x^-_{i-1/2}),
\end{aligned} \tag{5.1c}$$

with $\nabla_1 \hat{\mathbf{f}}$ and $\nabla_2 \hat{\mathbf{f}}$ defined as the gradient in the first and second argument in the numerical flux. In the one dimensional case we used a global α so the gradients of the numerical flux are given by:

$$\nabla_1 \hat{\mathbf{f}}(\mathbf{a}, -) = \frac{1}{2}\big[\nabla \mathbf{f}(\mathbf{a}) + \alpha\, \mathbf{a}\, \mathrm{I}\big], \qquad \nabla_2 \hat{\mathbf{f}}(-, \mathbf{b}) = \frac{1}{2}\big[\nabla \mathbf{f}(\mathbf{b}) - \alpha\, \mathbf{b}\, \mathrm{I}\big]\ . \tag{5.2}$$

with I being the identity matrix. Note that the Lax-Friedrichs flux is linear in its arguments, which simplifies the expressions for the gradients.

5.1.2 Two dimensional DG operator

To calculate $\mathcal{D}^{(2)}$ we will use again the spatial DG operator per element i and component j, $\mathcal{D}^{(1)}_{i,j}$ in (2.28), as basis. Furthermore we use the notation $\partial \mathbf{F}$, $\partial_1 \mathbf{F}$ and $\partial_2 \mathbf{F}$ for

$$\partial \mathbf{F} = (\nabla \mathbf{f}, \nabla \mathbf{g}), \qquad \partial_1 \hat{\mathbf{F}} = (\nabla_1 \mathbf{f}, \nabla_1 \mathbf{g}), \qquad \partial_2 \hat{\mathbf{F}} = (\nabla_2 \mathbf{f}, \nabla_2 \mathbf{g})\ . \tag{5.3}$$

Although our α depends on the solution we do not include this in the notation, but we do take the solution dependence of α into account in the derivative. Otherwise notation would be unnecessarily complicated. The gradients of the fluxes are given by (for $\mathbf{g}$ similar relations are obtained):

$$\nabla_1 \hat{\mathbf{f}}(\mathbf{a}, -) = \frac{1}{2}\big[\nabla \mathbf{f}(\mathbf{a}) \cdot \hat{\mathbf{n}} + \alpha\, \mathrm{I}\, \mathbf{a}\big], \qquad \nabla_2 \hat{\mathbf{f}}(-, \mathbf{b}) = \frac{1}{2}\big[\nabla \mathbf{f}(\mathbf{b}) \cdot \hat{\mathbf{n}} - \alpha\, \mathrm{I}\, \mathbf{b}\big]\ . \tag{5.4a}$$

The derivatives are then given by

$$\frac{\partial}{\partial \vec{u}_{b,a,i}}\Big(\mathrm{M}_i \cdot \mathcal{D}^{(2)}_{i,j}(\vec{u})\Big) = \int_{K_i} \Big[\mathbf{s}(\mathbf{u})\Big]_{b,j} \varphi_{:,i}\,\varphi_{a,i}\,\mathrm{d}x\mathrm{d}y + \int_{K_i} \big[\partial \mathbf{F}(\mathbf{u}) \cdot \nabla \varphi_{:,i}\big]_{b,j}\,\varphi_{a,i}\,\mathrm{d}x\mathrm{d}y \\ - \int_{\partial K_i} \left\{ \Big[\partial_1 \hat{\mathbf{F}}(\mathbf{u}^-,\mathbf{u}^+)_j \cdot \hat{\mathbf{n}}\Big]_{b,j} \varphi_{a,i} - \frac{1}{2}\frac{\partial \alpha(\vec{u}_{:,:,i})}{\partial \vec{u}_{b,a,i}}\big(\mathbf{u}^+ - \mathbf{u}^-\big)\right\} \varphi_{:,i}\,\mathrm{d}s, \quad (5.5a)$$

$$\frac{\partial}{\partial \vec{u}_{b,a,l}}\Big(\mathrm{M}_i \cdot \mathcal{D}^{(2)}_{i,j}(\vec{u})\Big) = -\int_{\partial K_i \cap \partial K_l} \left\{ \Big[\partial_2 \hat{\mathbf{F}}(\mathbf{u}^-,\mathbf{u}^+)_j \cdot \hat{\mathbf{n}}\Big]_{b,j} \varphi_{a,l} \right. \\ \left. - \frac{1}{2}\frac{\partial \alpha(\vec{u}_{:,:,i})}{\partial \vec{u}_{b,a,l}}\big(\mathbf{u}^+ - \mathbf{u}^-\big)\right\} \varphi_{:,i}\,\mathrm{d}s, \quad l \in S_i\,. \quad (5.5b)$$

The derivative with respect to α is given for our basis functions as

$$\frac{\partial \alpha(\vec{u}_{:,:,i})}{\partial \vec{u}_{b,a,l}} = \begin{cases} \dfrac{\partial \lambda_f(\vec{u}_{:,a,i})}{\partial \vec{u}_{b,a,l}}\hat{n}_x + \dfrac{\partial \lambda_g(\vec{u}_{:,a,i})}{\partial \vec{u}_{b,a,l}}\hat{n}_y & \text{if } (\cdot, a, l) = \underset{x=[1,d],y=[1,p],z=(i,l)}{\operatorname{maxloc}} \lambda(\vec{u}_{x,y,z}) \\ 0 & \text{otherwise} \end{cases}, \quad (5.6)$$

where the maxloc operator returns the position of the maximum value of its argument and λ_f and λ_g are the eigenvalues of the flux functions.

5.1.3 One dimensional WBAP limiter

For $k = 1$ the Jacobian of the limiter viewed as an operator on the DG coefficients, can be calculated straightforwardly from (4.18). $\mathcal{L}$ is used again to denote the limiter operator. We drop the index for the components in the equations, similar to Chapter 4.

$$\frac{\partial \mathcal{L}(\vec{u})_{0,i}}{\partial \vec{u}_{0,j}} = \delta_{ij}, \quad (5.7a)$$

$$\frac{\partial \mathcal{L}(\vec{u})_{1,i}}{\partial \vec{u}_{1,j}} = \delta_{ij}\,\mathrm{D}_1 L(\tfrac{2}{\Delta x_i} u_{1,i}, \Delta_+ \vec{u}_{0,i}, \Delta_- \vec{u}_{0,i}), \quad (5.7b)$$

and off diagonal by

$$\frac{\partial \mathcal{L}(\vec{u})_{1,i}}{\partial \vec{u}_{0,i-1}} = -\,\omega_{i,i-1}\,\mathrm{D}_3 L(\tfrac{2}{\Delta x_i} u_{1,i}, \Delta_+ \vec{u}_{0,i}, \Delta_- \vec{u}_{0,i}), \quad (5.7c)$$

$$\frac{\partial \mathcal{L}(\vec{u})_{1,i}}{\partial \vec{u}_{0,i}} = -\,\omega_{i,i-1}\,\mathrm{D}_2 L(\tfrac{2}{\Delta x_i} u_{1,i}, \Delta_+ \vec{u}_{0,i}, \Delta_- \vec{u}_{0,i}), \\ +\,\omega_{i,i+1}\,\mathrm{D}_3 L(\tfrac{2}{\Delta x_i} u_{1,i}, \Delta_+ \vec{u}_{0,i}, \Delta_- \vec{u}_{0,i}), \quad (5.7d)$$

$$\frac{\partial \mathcal{L}(\vec{u})_{1,i}}{\partial \vec{u}_{0,i+1}} = \omega_{i,i+1}\,\mathrm{D}_2 L(\tfrac{2}{\Delta x_i} u_{1,i}, \Delta_+ \vec{u}_{0,i}, \Delta_- \vec{u}_{0,i}), \quad (5.7e)$$

with ω being defined as

$$\omega_{i,j} = \frac{\Delta x_i}{\Delta x_i + \Delta x_j} \ . \tag{5.8}$$

Using (4.19) the derivatives for $k = 2$ with respect to the first argument are:

$$\frac{\partial \mathcal{L}(\vec{u})_{0,i}}{\partial \vec{u}_{0,j}} = \delta_{ij}, \tag{5.9a}$$

$$\frac{\partial \mathcal{L}(\vec{u})_{1,i}}{\partial \vec{u}_{1,j}} = \frac{\delta_{ij}}{2}\Big(\mathrm{D}_1 L_A + \mathrm{D}_1 L_B\Big), \qquad \frac{\partial \mathcal{L}(\vec{u})_{2,i}}{\partial \vec{u}_{1,j}} = \frac{\delta_{ij}}{2}\Big(\mathrm{D}_1 L_A - \mathrm{D}_1 L_B\Big), \tag{5.9b}$$

$$\frac{\partial \mathcal{L}(\vec{u})_{1,i}}{\partial \vec{u}_{2,j}} = \frac{\delta_{ij}}{2}\Big(\mathrm{D}_1 L_A - \mathrm{D}_1 L_B\Big), \qquad \frac{\partial \mathcal{L}(\vec{u})_{2,i}}{\partial \vec{u}_{2,j}} = \frac{\delta_{ij}}{2}\Big(\mathrm{D}_1 L_A + \mathrm{D}_1 L_B\Big), \tag{5.9c}$$

and with respect to the second arguments:

$$\frac{\partial \mathcal{L}(\vec{u})_{1,i}}{\partial \vec{u}_{0,i}} = -\frac{\omega_{i,i-1}}{2}\Big(\mathrm{D}_3 L_A + \mathrm{D}_3 L_B\Big) + \frac{\omega_{i,i+1}}{2}\Big(\mathrm{D}_2 L_A + \mathrm{D}_2 L_B\Big), \tag{5.9d}$$

$$\frac{\partial \mathcal{L}(\vec{u})_{2,i}}{\partial \vec{u}_{0,i}} = -\frac{\omega_{i,i-1}}{2}\Big(\mathrm{D}_3 L_A - \mathrm{D}_3 L_B\Big) + \frac{\omega_{i,i+1}}{2}\Big(\mathrm{D}_2 L_A - \mathrm{D}_2 L_B\Big), \tag{5.9e}$$

$$\frac{\partial \mathcal{L}(\vec{u})_{1,i}}{\partial \vec{u}_{0,i-1}} = -\frac{\omega_{i,i-1}}{2}\Big(\mathrm{D}_3 L_A + \mathrm{D}_3 L_B\Big), \qquad \frac{\partial \mathcal{L}(\vec{u})_{2,i}}{\partial \vec{u}_{0,i-1}} = -\frac{\omega_{i,i-1}}{2}\Big(\mathrm{D}_3 L_A - \mathrm{D}_3 L_B\Big), \tag{5.9f}$$

$$\frac{\partial \mathcal{L}(\vec{u})_{1,i}}{\partial \vec{u}_{0,i+1}} = \frac{\omega_{i,i+1}}{2}\Big(\mathrm{D}_2 L_A + \mathrm{D}_2 L_B\Big), \qquad \frac{\partial \mathcal{L}(\vec{u})_{2,i}}{\partial \vec{u}_{0,i+1}} = \frac{\omega_{i,i+1}}{2}\Big(\mathrm{D}_2 L_A - \mathrm{D}_2 L_B\Big) \ . \tag{5.9g}$$

5.1.4 Two dimensional WBAP limiter

For the two dimensional case the limiter viewed as operator on the DG coefficients can be calculated straightforwardly from (4.31). $\mathcal{L}$ is used again to denote the limiter operator. With respect to the first argument of the limiter:

$$\frac{\partial \mathcal{L}(\vec{u})_{k,i}}{\partial \vec{u}_{a,i}} = \mathrm{W}^{-1} \cdot \mathrm{A}_i^T \cdot \nabla\left[\widetilde{\mathrm{D}_1 L}\Big(\cdots\Big)\right] \cdot \mathrm{A}_i^{-T} \cdot \mathrm{W} + \frac{\partial \bar{u}_i}{\partial \vec{u}_{a,i}}, \tag{5.10a}$$

$$\frac{\partial \mathcal{L}(\vec{u})_{k,i}}{\partial \vec{u}_{a,l}} = \mathrm{W}^{-1} \cdot \mathrm{A}_i^T \cdot \nabla\left[\widetilde{\mathrm{D}_l L}\Big(\cdots\Big)\right] \cdot \mathrm{Y}_l, \quad l \in S_i, \tag{5.10b}$$

$$\frac{\partial \mathcal{L}(\vec{u})_{k,i}}{\partial \vec{u}_{a,s}} = \mathrm{W}^{-1} \cdot \mathrm{A}_i^T \cdot \nabla\left[\widetilde{\mathrm{D}_l L}\Big(\cdots\Big)\right] \cdot \mathrm{Z}_{l,s}, \quad s \in \hat{S}_l, l \in S_i \ . \tag{5.10c}$$

Note that $\hat{S}$ denotes the elements in the reconstruction process and $S \subset \hat{S}$ denotes the neighboring elements used in the WBAP limiter. These equations are not unique, but will overlap.

Their contributions should therefore be summed up. The new matrices Y_l and $Z_{l,s}$ are given by:

$$(Y_l)_{j,k} = -\left(\mathrm{R}_l^{-1}\mathrm{Q}_l^{\mathrm{T}} \cdot \omega_l\right)_k \frac{\partial \bar{u}_l}{\partial \vec{u}_{j,l}} \ , \tag{5.11}$$

$$(Z_{l,s})_{j,k} = \left(\mathrm{R}_l^{-1}\mathrm{Q}_l^{\mathrm{T}}\right)_{j,s} (\omega_l)_s \frac{\partial \bar{u}_s}{\partial \vec{u}_{j,s}} \ . \tag{5.12}$$

Note that in the last equation the s is used as index for the matrix $\mathrm{R}^{-1}Q^{\mathrm{T}}$ and vector ω_l. In this context not the element number is meant, but the position in the set $\hat{S}_l$.

5.1.5 Numerical Tests

The analytic Jacobians of the previous sections were also numerically tested. Using the central difference scheme:

$$J_F(\vec{u})_{:,j} = \frac{F(\vec{u}_1, \cdots, \vec{u}_i + \delta, \cdots, \vec{u}_n) - F(\vec{u}_1, \cdots, \vec{u}_i - \delta, \cdots, \vec{u}_n)}{2\delta}, \quad \delta = 10^{-6}, \tag{5.13}$$

for every coefficient the functions were tested with random vectors several times. The difference between the analytic and numerical Jacobian were 10^{-7} and 10^{-8} in the L_1 and L_∞ norm.

5.2 Newton method

The Newton(-Raphson) is a well-known method to solve non-linear root problems. The idea is that if we have the equation:

$$f(x) = 0,$$

we can Taylor expand it to its unknown solution a from x by:

$$f(a) = f(x) + f'(x)(x-a) + O\Big((x-a)^2\Big) = 0 \ .$$

Assuming that we are already in the neighborhood, $(x - a) \ll 1$, we can truncate the second and higher order terms and get a new and better approximation. So given x^m we can get x^{m+1} by:

$$x^{m+1} = x^m - \frac{f(x^m)}{f'(x^m)} \ ,$$

continuing till the absolute value of the update $|x^{m+1} - x^m|$, or the function value $|f(x^{m+1})|$, comes below a certain tolerance. Usually this is taken close to the order of the numerical approximation on the computer. All we need is some initial guess.

This idea can be extended to a general $\mathbb{R}^Q \to \mathbb{R}^Q$ function:

$$v^{m+1} = v^m - J_F^{-1}(v^m)\, F(v^m) \ ,$$

with J_F being the Jacobian of F. Now we will iterate until the difference between the second to last and the last in some norm is smaller than the prescribed tolerance. Note that in the actual computations we will solve a linear system instead of calculating the inverse.

Locally the Newton method can be proved to have second order convergence in the number of iterations. The radius of convergence can, however, be quite small. We will will use the previous time step as initial guess, but if the time step is too large, it can diverge from the solution.

All the above leads to the following algorithm.[1]

$$J_F(v^m)s^m = -F(v^m), \tag{5.14a}$$
$$v^{m+1} = v^m + s^m, \tag{5.14b}$$

with the following equations for F and J_F for the backward Euler time integration method (4.32)

$$F(x) = x - \mathcal{L}\big[u^n + \Delta t\, \mathcal{D}(x)\big], \tag{5.15a}$$
$$J_F(x) = I - \Delta t\, J_{\mathcal{L}}\big(u^n + \Delta t\, \mathcal{D}(x)\big)\, J_{\mathcal{D}}(x)\ . \tag{5.15b}$$

and for the Crank-Nicolson time integration method (4.33)

$$F(x) = x - \mathcal{L}\Big[u^n + \tfrac{\Delta t}{2}\big\{\mathcal{D}(u^n) + \mathcal{D}(x)\big\}\Big], \tag{5.16a}$$
$$J_F(x) = I - \tfrac{\Delta t}{2}\, J_{\mathcal{L}}\Big(u^n + \tfrac{\Delta t}{2}\big\{\mathcal{D}(u^n) + \mathcal{D}(x)\big\}\Big)\, J_{\mathcal{D}}(x), \tag{5.16b}$$

with I being the identity matrix.

5.3 Damped Newton method

As mentioned before, the quadratic convergence of the Newton method is local. It has no global convergence properties. This can be a problem for large time steps or a small radius of convergence due to complicated algebraic equations. That is why we also need a damped Newton iteration, which does have global convergence properties. The difference between the damped Newton and the Newton method is that the directional variable s, calculated in (5.14), is multiplied with a damping factor λ. This leads to the following algorithm:

$$J_F(v^n)s^n = -F(v^n), \tag{5.17a}$$
$$v^{n+1} = v^n + \lambda s^n, \qquad \lambda \in (0,1)\ . \tag{5.17b}$$

Note that if $\lambda = 1$ we obtain the original Newton method.

The main idea is that you take the direction s, but control the step size λ. The only question remaining is how to choose λ. The process is called the line-search procedure.

[1]We drop the arrow and the indices from the notation for clarity

5.3.1 Line-search

The process of determining λ is called line-search because until the 1960's it was a prevailing believe that λ should be chosen such that it minimizes the function in $\lambda \in (0, 1)$. But computational experience has shown the importance of taking a full Newton step whenever possible [12].

In our algorithm we therefore first try a $\lambda = 1$ step. A preset improvement is demanded to apply λ in order to assure convergence

$$\log_{10}\left[\|F(v^{n+1})\| - \|F(v^{n})\|\right] < -\lambda\,\mathrm{tol}\;. \tag{5.18}$$

Note that if $\mathrm{tol} = 0$, then this is equivalent to the Armijo rule [26]. But demanding a set improvement makes the line search faster as the method will sooner switch to damped Newton. If this condition does not hold the λ value will be multiplied by a certain coefficient ρ until a lower bound is reached. The set improvement is multiplied with λ to relax the condition when a smaller step size is taken.

It should be mentioned that in the literature more sophisticated algorithms are present to calculate λ. For example Dennis and Schnabel [12] use the Euclidean norm and convert the non-linear equation in an optimization problem (Gauss-Newton method). Doing so gives nice properties and gives the opportunity to use the algorithms developed for $\mathbb{R}^n \to \mathbb{R}$ problems such as backtracking or by using a trust-region. The Levenberg-Marquadt [24] algorithm uses both, namely a Gauss-Newton method combined with a trust-region method.

However, experience with the Burgers equation indicated that the Levenberg-Marquadt algorithm performed equally well and backtracking performed worse than our simple algorithm. One main difference between our algorithm and the backtracking algorithm, is that we calculate the norm of the actual function rather than the norm of the coefficients. The former is more meaningful, for the objective is to solve this underlying problem. In a test with the Burgers equation our simple algorithm performed reasonably well, i.e. halve of the time it only needs one step, even for large time steps.

Chapter 6

Implicit time integration of Euler equations

The compressible Euler equations are a set of equations governing the flow of inviscid compressible gasses. In this hyperbolic system mass, momentum and energy are conserved. In one dimension the equations are given by [30]:

$$\begin{pmatrix} \rho \\ \rho u \\ E \end{pmatrix}_t + \begin{pmatrix} \rho u \\ \rho u^2 + p \\ u(E+p) \end{pmatrix}_x = \mathbf{0} \equiv \mathbf{u}_t + \mathbf{F}(\mathbf{u})_x \ , \tag{6.1}$$

with E the total energy per unit area given by:

$$E = \rho\Big(\frac{1}{2}u^2 + e\Big) \ , \tag{6.2}$$

where e is the specific internal energy. We have four unknowns in these equations (ρ, u, p, e) and only three equations. For inviscid fluids one can add the assumption that the divergence of the velocity field is zero. For gasses we have to use an equation of state (EOS) such that we can relate one state variable to another one. We use the caloric EOS of an ideal gas, which is given by the simple expression:

$$e = e(\rho, p) = \frac{p}{(\gamma - 1)\rho} \ , \tag{6.3}$$

in which γ is the ratio of the specific heats at constant pressure and volume, which is always greater than 1.

An important concept in gas dynamics is the sound speed. The sound speed is a relative measure for the speed of the fluid and characterizes the wave speeds. The speed of sound for a calorically perfect gas is given by:

$$a = \sqrt{\frac{\gamma p}{\rho}} \ . \tag{6.4}$$

If we transform (6.1) into quasi-linear form we have:

$$\mathbf{u}_t + \mathbf{A}(\mathbf{u})\,\mathbf{u}_x = \mathbf{0}\ , \tag{6.5}$$

with

$$\mathbf{A} = \frac{\partial \mathbf{F}}{\partial \mathbf{u}} = \begin{pmatrix} 0 & 1 & 0 \\ -\frac{1}{2}(\gamma-3)u^2 & (3-\gamma)u & \gamma-1 \\ -\gamma uE/\rho + (\gamma-1)u^3 & \gamma E/\rho - \frac{3}{2}(\gamma-1)u^2 & \gamma u \end{pmatrix}\ , \tag{6.6}$$

in which the characteristic wave speeds are given by the eigenvalues of **A** – the Jacobian of **F** – which are $\{u-a, u, u+a\}$.

In this chapter we will test the accuracy of the different versions of the limiter and the convergence properties and limiting capabilities of the WBAP limiter compared with the minmod-TVB limiter [8]. The limiting capabilities will be tested using the Harten-Lax problem and the blast wave problem. In these problems the minmod-TVB limiter will also be used as troubled cell indicator. When the limiter returns something else in the first argument it is a troubled cell and will be limited.

This troubled cell indicator is motivated by the notion that the limiter introduces extra difficulty in solving the algebraic equations and restricts the radius of convergence of the Newton method. If we restrict the limiter only to those elements which need limiting we remove unnecessary complications in solving of the algebraic equations and hopefully achieve better convergence properties. The sensitivity to indicate troubled cells is governed by the M variable in the minmod-TVB limiter. If M is too low, elements will wrongly be marked as troubled, and if M is too large necessary limiting will be omitted.

We will use a uniform grid and a non-uniform grid to check if the limiter also works for the latter grid. The non-uniform grid is given by:

$$\Delta x_i = \begin{cases} \frac{3H}{2N} & \text{if } i \text{ is odd} \\ \frac{1H}{2N} & \text{if } i \text{ is even} \end{cases} \tag{6.7}$$

with H the length of the domain, which is equal to 1 for this problem. This grid is highly artificial since it has a 3 to 1 ratio in every neighboring element.

6.1 Accuracy

To test the order of accuracy we need a continuous problem since discontinuous problems always have some artificial viscosity in the numerical solution. This results in a smooth transitions and a large error in the discontinuity. We will use the following initial condition on the domain of $x \in [0,1]$ with periodic boundary conditions:

$$\begin{pmatrix} \rho \\ u \\ p \end{pmatrix}(x) = \begin{pmatrix} 1 + \frac{1}{2}\sin(2\pi\,x) \\ 1 \\ 1 \end{pmatrix}\ . \tag{6.8}$$

The exact solution is given by:

$$\begin{pmatrix} \rho \\ u \\ p \end{pmatrix} (x,t) = \begin{pmatrix} 1 + \frac{1}{2}\sin(2\pi(x-t)) \\ 1 \\ 1 \end{pmatrix} . \tag{6.9}$$

Results for a uniform mesh are given in Table 6.1 and for a non-uniform mesh in Table 6.2. From the results we see that the WBAP limiter achieves second order accuracy in the L_1 norm and around 1.5 in the L_∞ norm, which is suboptimal. The WBAP-L1 limiter is better than the WBAP-L2 limiter, with the original WBAP limiter somewhere in between. The minmod-TVB limiter is as good as the unlimited case on a uniform grid as can be expected from the proof and the fact that we have a smooth solution. The absolute error of the minmod-TVB limiter is the same as the WBAP limiters on non-uniform grids, but the order of the error is lower than the order 1.5 of the WBAP limiters.

6.2 Harten-Lax problem

The Harten-Lax problem is a shock tube problem. It is used to test the capability of limiters in capturing strong shock waves and contact discontinuities. Aside from those it also creates a rarefaction wave. The initial solution is given by:

$$(\rho, u, p)^{\mathrm{T}}(x) = \begin{cases} (0.445, 0.698, 3.528)^{\mathrm{T}} & \text{if } x \leq \frac{1}{2} \\ (0.5, 0, 0.0571)^{\mathrm{T}} & \text{if } x > \frac{1}{2} \end{cases} . \tag{6.10}$$

The constant M for the minmod-TVB limiter is set at $M = 0.1$, which is reasonable for this problem. We want to test two things with this test case. First, we want to test the convergence properties of the limiter as a function of the time step and compare with the minmod-TVB limiter and the unlimited case. Our hope is that the convergence properties remain unchanged or that the number of iterations is smaller than used in the forward Euler method with maximum possible time step given by the CFL condition. Secondly, we want to investigate what the properties of the various limiters are. The analytic solutions of this Riemann problem are obtained from Chapter 4 in [30]. The non-linear equation (4.5) in that chapter is solved using the Newton-Raphson method as described in Section 5.2.

6.2.1 Convergence of the iterative method as a function of the time step

In Figure 6.1 the number of iterations is plotted against the relative time step for a uniform grid and in Figure 6.2 for a non-uniform grid.

The most important thing we notice, is that limiters worsen the convergence properties of the non-linear solver. If no limiter is applied, no damping in the Newton method is needed for any time step up to $8\,\Delta t_{\mathrm{CFL}}$. The damping makes the convergence of the Newton method slower and grows when the time step is enlarged. However, the WBAP-L1 limiter can still be applied at $t = 12\,\Delta t_{\mathrm{CFL}}$ whereas the case without limiter halts for this time step.

Table 6.1: Density accuracy test for the Euler equations using the Crank-Nicolson time integration method, $p = 1$, Δt= 2.0 CFL and $t_{\text{final}} = 0.5$ on a *uniform* grid.

Case	N	L_1 error	Order	L_∞ error	Order
Unlimited	10	5.59e-03		1.62e-02	
	20	1.28e-03	2.13	4.19e-03	1.95
	40	3.11e-04	2.04	1.07e-03	1.97
	80	7.70e-05	2.01	2.71e-04	1.98
	160	1.92e-05	2.00	6.83e-05	1.99
WBAP-L1, $n = 1$	10	9.27e-02		1.84e-01	
	20	2.85e-02	1.70	6.60e-02	1.48
	40	7.71e-03	1.89	2.51e-02	1.39
	80	1.80e-03	2.10	9.32e-03	1.43
	160	4.12e-04	2.13	3.42e-03	1.45
WBAP-L1, $n = 5$	10	3.94e-02		1.00e-01	
	20	1.38e-02	1.51	3.69e-02	1.44
	40	3.06e-03	2.17	1.34e-02	1.46
	80	6.73e-04	2.18	4.83e-03	1.47
	160	1.51e-04	2.15	1.75e-03	1.46
WBAP-L2, $n = 1$	10	1.24e-01		2.30e-01	
	20	3.81e-02	1.70	8.37e-02	1.46
	40	1.12e-02	1.77	3.25e-02	1.37
	80	2.61e-03	2.10	1.21e-02	1.42
	160	5.91e-04	2.14	4.43e-03	1.45
WBAP-L2, $n = 5$	10	5.38e-02		1.32e-01	
	20	2.08e-02	1.37	4.91e-02	1.43
	40	4.63e-03	2.16	1.81e-02	1.44
	80	1.03e-03	2.16	6.57e-03	1.46
	160	2.27e-04	2.19	2.36e-03	1.48
WBAP, $\epsilon = 1, n = 3$	10	6.81e-02		1.53e-01	
	20	2.47e-02	1.46	5.57e-02	1.46
	40	5.66e-03	2.12	2.07e-02	1.43
	80	1.28e-03	2.15	7.54e-03	1.46
	160	2.81e-04	2.19	2.71e-03	1.47
Minmod-TVB	10	5.59e-03		1.62e-02	
	20	1.28e-03	2.13	4.19e-03	1.95
	40	3.11e-04	2.04	1.07e-03	1.97
	80	7.70e-05	2.01	2.71e-04	1.98
	160	1.92e-05	2.00	6.83e-05	1.99

Table 6.2: Density accuracy test for the Euler equations using the Crank-Nicolson time integration method, $p = 1$, Δt= 2.0 CFL and $t_{\text{final}} = 0.5$ on a *non-uniform* grid.

Case	N	L_1 error	Order	L_∞ error	Order
Unlimited	10	1.47e-02		4.06e-02	
	20	4.09e-03	1.85	1.43e-02	1.50
	40	1.05e-03	1.97	4.16e-03	1.79
	80	2.60e-04	2.01	1.10e-03	1.92
	160	6.47e-05	2.01	2.80e-04	1.97
WBAP-L1, $n = 1$	10	8.47e-02		1.61e-01	
	20	2.59e-02	1.71	5.50e-02	1.55
	40	7.61e-03	1.77	2.15e-02	1.36
	80	1.94e-03	1.97	8.02e-03	1.42
	160	4.79e-04	2.02	2.98e-03	1.43
WBAP-L1, $n = 5$	10	4.47e-02		9.92e-02	
	20	1.53e-02	1.55	3.59e-02	1.47
	40	4.11e-03	1.89	1.30e-02	1.46
	80	1.05e-03	1.97	4.90e-03	1.41
	160	2.58e-04	2.03	1.83e-03	1.43
WBAP-L2, $n = 1$	10	1.03e-01		1.94e-01	
	20	3.28e-02	1.66	6.88e-02	1.49
	40	9.87e-03	1.73	2.71e-02	1.34
	80	2.49e-03	1.99	1.02e-02	1.41
	160	6.11e-04	2.03	3.81e-03	1.42
WBAP-L2, $n = 5$	10	5.98e-02		1.26e-01	
	20	2.01e-02	1.57	4.55e-02	1.47
	40	5.23e-03	1.95	1.66e-02	1.45
	80	1.32e-03	1.98	6.26e-03	1.41
	160	3.22e-04	2.04	2.33e-03	1.42
WBAP, $\epsilon = 1, n = 3$	10	6.73e-02		1.37e-01	
	20	2.23e-02	1.60	4.82e-02	1.50
	40	5.98e-03	1.90	1.80e-02	1.42
	80	1.50e-03	1.99	6.75e-03	1.42
	160	3.67e-04	2.03	2.50e-03	1.43
Minmod-TVB	10	3.64e-02		1.01e-01	
	20	1.33e-02	1.46	3.64e-02	1.47
	40	3.58e-03	1.89	1.70e-02	1.10
	80	8.89e-04	2.01	7.16e-03	1.25
	160	2.14e-04	2.06	2.76e-03	1.38

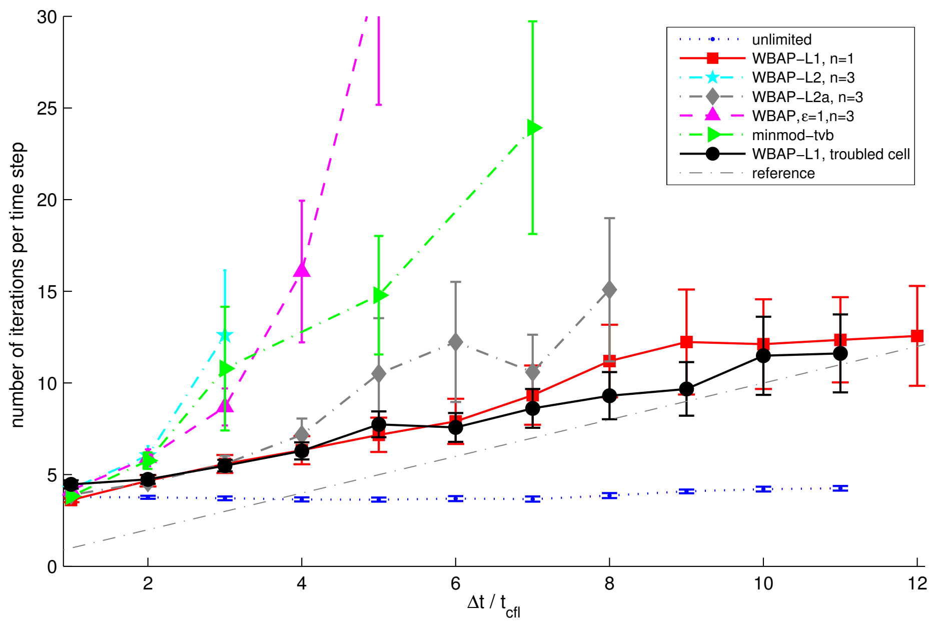


Figure 6.1: The mean number of iterations per time step with the standard deviations (divided by five) of the Harten-Lax problem for different time step sizes with different limiters on a *uniform* grid ($N = 100$) with $p = 1$ and $t_{\text{final}} = 0.1$ using the backward Euler time integration method. A reference line is included to denote the number of time steps needed for an explicit method with $\Delta t = \Delta t_{\text{CFL}}$

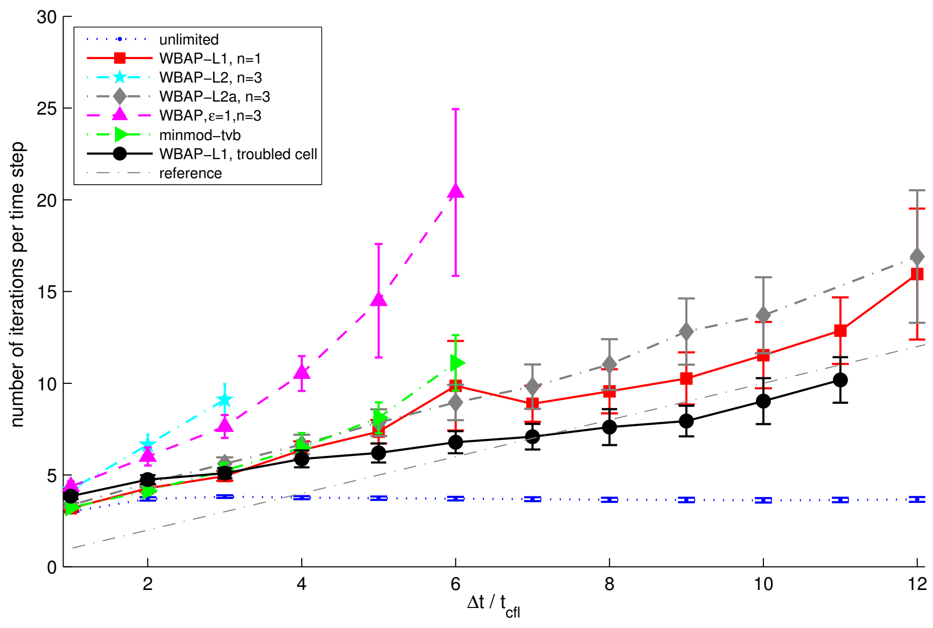


Figure 6.2: The mean number of iterations per time step with the standard deviations (divided by five) of the Harten-Lax problem for different time step sizes with different limiters on a *non-uniform* grid ($N = 100$) with $p = 1$ and $t_{\text{final}} = 0.1$ using the backward Euler time integration method. A reference line is included to denote the number of time steps needed for an explicit method with $\Delta t = \Delta t_{\text{CFL}}$

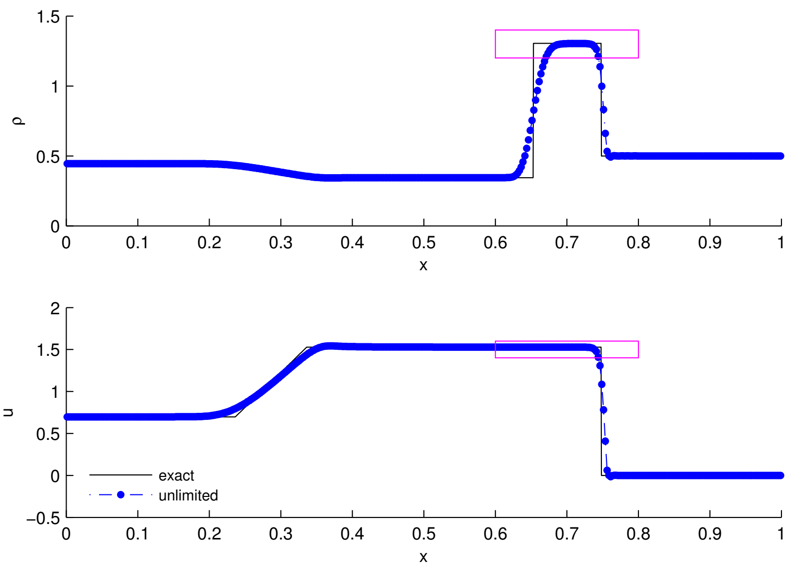


Figure 6.3: The solution of the Harten-Lax problem showing density and velocity on a *uniform* grid ($N = 200$) with $p = 1$, $\Delta t = 2 = \Delta t_{\text{CFL}}$ at $t_{\text{final}} = 1.5$ using the backward Euler time integration method. The rectangle denotes the region shown in Figures 6.4 and 6.5.

A second important observation is that the WBAP-L1 limiter and the alternative WBAP-L2 limiter have better convergence properties than the original WBAP limiter and the WBAP-L2 limiter. However, the former limiters are more discontinuous as they limit the gradient in an element to zero when the signs of the arguments are not equal with all the compared gradients. Application of the troubled cell method does not seem to give a significant improvement in the convergence properties of the WBAP-L1 limiter.

6.2.2 Limiter accuracy

In order to compare the capabilities of the various limiters we look at solution at $t = 1.5$, zoomed in at the discontinuity where the oscillations will arise. This is the contact discontinuity and the shock wave. In Figure 6.3 the region of interest is depicted and the exact solution in the whole domain is shown together with a solution without limiter. For clarity, we only take the WBAP-L1 limiter variant of the available WBAP limiter versions. The other limiter versions produce almost identical results provided they converge. The results are shown in Figure 6.4 for a uniform grid and in Figure 6.5 for a non-uniform grid.

On a uniform grid there are no oscillations at all, even for the unlimited case. However, the troubled cell approach does affect the values of the density, which probably indicates an oversensitivity for troubled cells (M to low). This is contradictory to the non-uniform grid, where the WBAP-L1 limiter with troubled cell indicator does not remove the numerical oscillations and points to an undersensitivity for troubled cells.

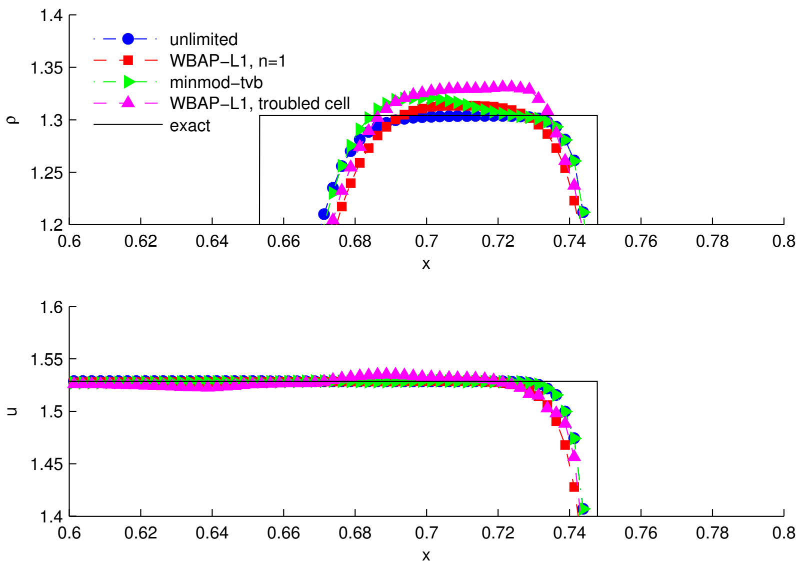


Figure 6.4: Detail of the numerical solution of the Harten-Lax problem showing density and velocity on a *uniform* grid ($N = 200$) with $p = 1$, $\Delta t = 2 = \Delta t_{\text{CFL}}$ and $t_{\text{final}} = 1.5$ using the backward Euler time integration method.

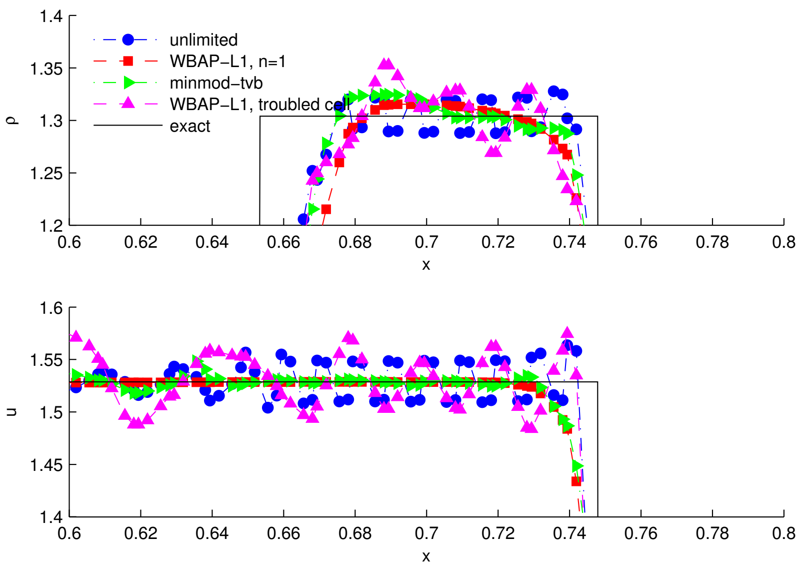


Figure 6.5: Detail of the numerical solution of the Harten-Lax problem showing density and velocity on a *non-uniform* grid ($N = 200$) with $p = 1$, $\Delta t = \Delta t_{\text{CFL}}$ and $t_{\text{final}} = 1.5$ using the backward Euler time integration method.

Compared with the minmod-TVB limiter, the WBAP limiter works good. It removes all the oscillations and has relatively small overshoots. Combined with the result of the convergence the WBAP-L1 limiter is the best limiter in this test.

6.3 Blast wave problem

The blast wave problem is also a shock tube problem just like the Harten-Lax problem, only much harder. Its initial solution is given by:

$$(\rho, u, p)^{\mathrm{T}}(x) = \begin{cases} (1, 0, 1000)^{\mathrm{T}} & \text{if } x \leq \frac{1}{2} \\ (1, 0, 0.1)^{\mathrm{T}} & \text{if } x > \frac{1}{2} \end{cases}. \tag{6.11}$$

The value for the minmod-TVB limiter is set at $M = 0.1$.

We will perform the same tests on this problem as with the Harten-Lax problem, namely the convergence properties of the Newton method as function of the time step and the limiting properties.

6.3.1 Convergence of the iterative method as a function of the time step

In Figure 6.6 the number of iterations is plotted against the relative time step for a uniform grid and in Figure 6.7 for a non-uniform grid. The results are similar as the results of the Harten-Lax problem for a non-uniform grid, i.e. the unlimited solution converges significantly better than the limited solutions and the WBAP-L1 limiter and the alternative WBAP-L2 variant are better than the other WBAP limiters and the minmod-TVB limiter. At a uniform grid we see some other results. The unlimited case has no solution for $\Delta t = \Delta t_{\mathrm{CFL}}$, and, moreover, the convergence halts for $\Delta t > 6\Delta t_{\mathrm{CFL}}$ only when combined with the WBAP-L1 it does not. Our guess is that the numerical oscillations of this problem are larger than in the previous problem and hamper the convergence when the time steps are larger.

Although it would seem that the solution on a uniform grid is easier to obtain than on a non-uniform grid and therefore this result should be the other way around, but this is not the case. A simple explanation is that the CFL condition is based on the smallest element and for the non-uniform grid this is relatively larger for half of the elements than on the uniform grid. This makes a straightforward comparison between the uniform and non-uniform grid unfair.

6.3.2 Limiting accuracy

The capabilities of the various limiters are analyzed again by comparing the solutions at $t = 1.5$ zoomed in at the discontinuity where the oscillations will arise. This are the contact discontinuity and the shock wave. In Figure 6.8 the exact solution on the whole domain is shown and our region of interest together with a numerical solution without limiter. For clarity in the figures, we again only show the WBAP-L1 limiter of the available WBAP limiter versions. The

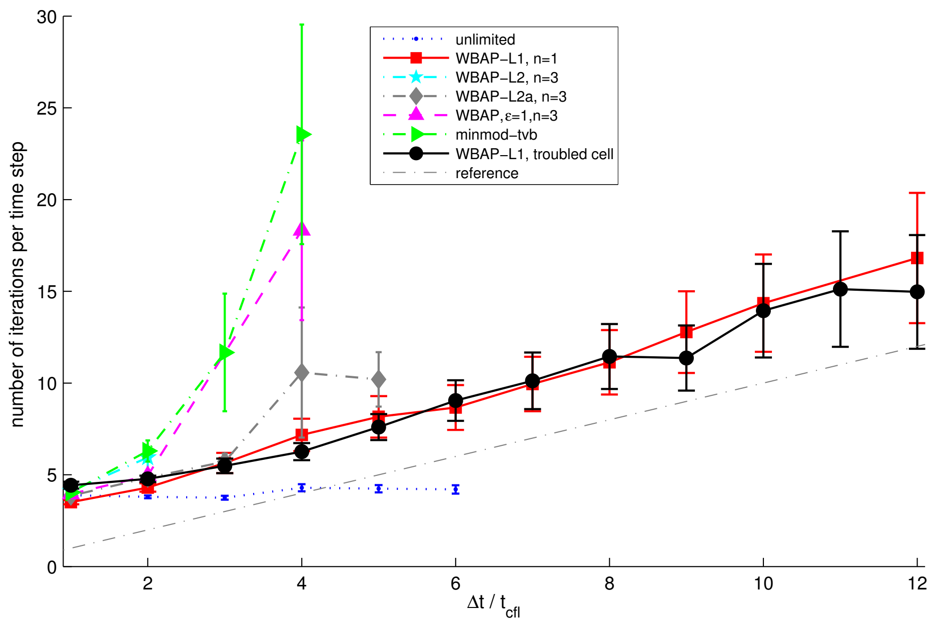


Figure 6.6: The mean number of iterations per time step with the standard deviations (divided by five) of the blast wave problem for different time step sizes compared with different limiters on a *uniform* grid ($N = 100$) with $p = 1$ and $t_{\text{final}} = 0.1$ using the backward Euler time integration method. A reference line is included to denote the number of time steps needed for an explicit method with $\Delta t = \Delta t_{\text{CFL}}$

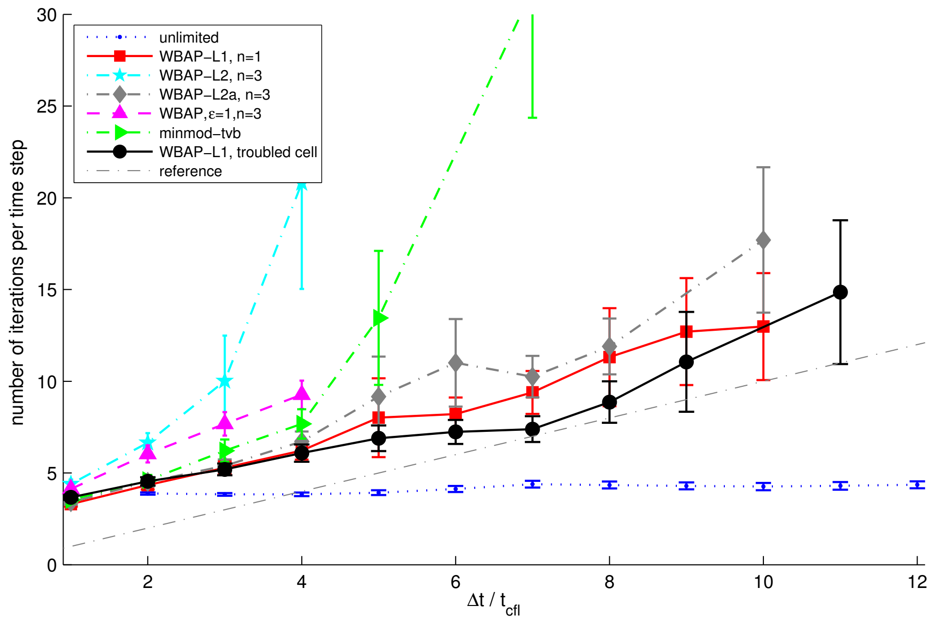


Figure 6.7: The mean number of iterations per time step with the standard deviations (divided by five) of the blast wave problem for different time step sizes compared with different limiters on a *non-uniform* grid ($N = 100$) with $p = 1$ and $t_{\text{final}} = 0.1$ using the backward Euler time integration method. A reference line is included to denote the number of time steps needed for an explicit method with $\Delta t = \Delta t_{\text{CFL}}$

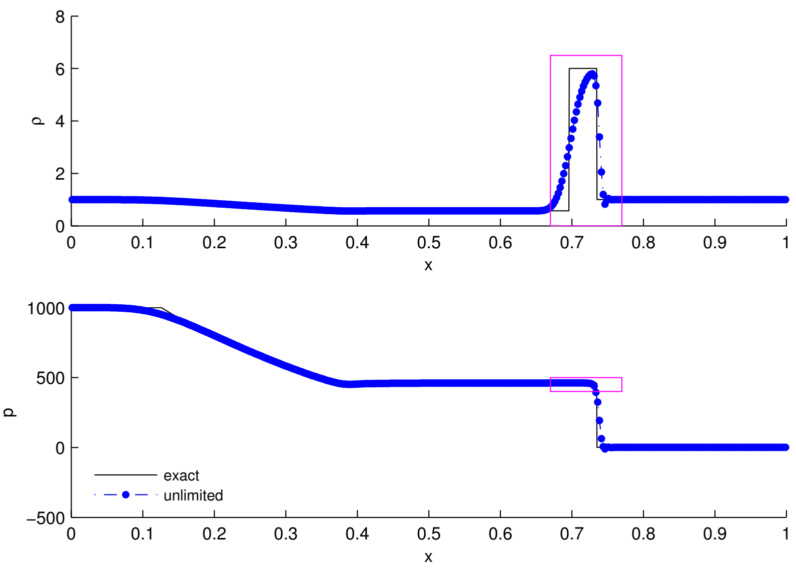


Figure 6.8: The numerical solution of the blast wave problem showing density and pressure on a *uniform* grid ($N = 200$) with $p = 1$, $\Delta t = 2 = \Delta t_{\text{CFL}}$ at $t_{\text{final}} = 1.5$ using the backward Euler time integration method. The rectangle denotes the region shown in Figures 6.9 and 6.10.

other versions produce almost identical results provided they converge. The results are shown in Figure 6.9 for the uniform grid and in Figure 6.10 for the non-uniform grid.

The first thing that can be noticed is that the method has far more difficulty to get the shape of the solution right. This is due to the backward Euler method which reduces the order of accuracy to first order. With limiters the numerical dissipation causes the density to reduce with about 10%. The results with the limiter are the same as for the Harten-Lax problem, only amplified. The minmod-TVB limiter does not remove the oscillations on the non-uniform mesh, neither does the WBAP-L1 limiter with the troubled cell indicator. The WBAP-L1 limier computes the height of the pressure slightly better on the non-uniform grid than the minmod-TVB limiter, but it is the other way around on the uniform grid.

6.4 Conclusions

The main conclusion of this chapter is that the WBAP limiters can be combined with the discontinuous Galerkin method in combination with a backward Euler or Crank-Nicolson time integration method. The L_1 error is second order as expected and the L_∞ error half an order lower. This error is slightly better than obtained with the minmod-TVB on non-uniform grids, but the absolute error is of the same order.

When applied to discontinuous problems we see that the use of a limiter results in significantly more iterations in the Newton method than without a limiter. The specific increase

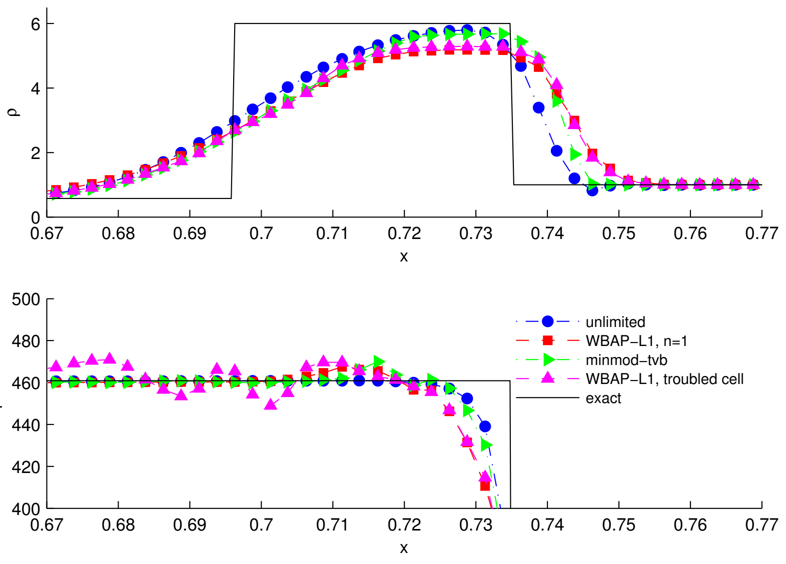


Figure 6.9: Detail of the numerical solution of the blast wave problem showing density and pressure on a *uniform* grid ($N = 200$) with $p = 1$, $\Delta t = 2 = \Delta t_{\text{CFL}}$ at $t_{\text{final}} = 1.5$ using the backward Euler time integration method.

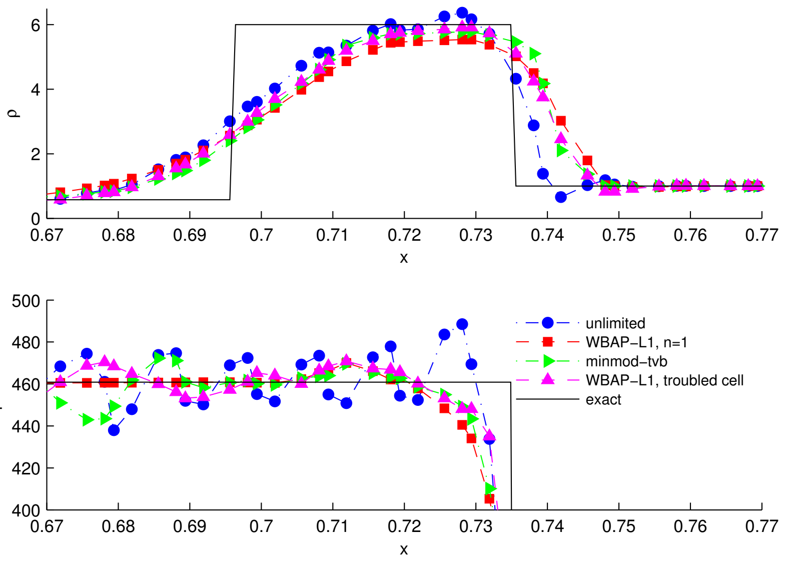


Figure 6.10: Detail of the numerical solution of the blast wave problem showing density and pressure on a *non-uniform* grid ($N = 200$) with $p = 1$, $\Delta t = \Delta t_{\text{CFL}}$ at $t_{\text{final}} = 1.5$ using the backward Euler time integration method.

differs significantly between the versions of the WBAP limiters. The WBAP-L1 limiter and the alternative WBAP-L2 limiter are better than the other versions. The difference between those two group of limiters is that the WBAP-L1 limiter and the alternative WBAP-L2 limiter reduce the gradient to zero if not all gradients are of the same sign. Compared to the minmod-TVB limiter the convergence in the Newton method of the WBAP-L1 limiter are better. The qualify of the limited solutions are comparable. In problems with relatively large numerical oscillations the use of the WBAP-L1 limiter improves the convergence rate of the Newton method for large time steps, since the unlimited case halts and does not converge at some point during the iterations.

Chapter 7

Steady state solution of the Burgers equation

In this chapter we will investigate the limiter for steady state problems. Steady state problems arise when the input boundaries conditions are fixed in time and not periodic. Usually it comes together with a source term which compensates the flux term when the solution has reached steady state. The use of an implicit time integration method is advantageous because the time step can be increased when the solution is close to steady state. This results in exponential convergence of the solution to steady state and reduces the number of time steps dramatically.

We will investigate two different steady state problems for the inviscid Burgers equation. The problems are taken from Chou and Shu [7]. The first problem has a source term which is independent of the solution and the second one does depend on the solution. In both problems we investigate the convergence of the Newton method as a function of the time step. This is largely the same as in the previous chapter.

Also, we will study how the limiter reacts to a time step increase with respect to the convergence of the steady state solution. This time step increase will be done in a simple way by multiplying it with a fixed multiplier from an offset. This method is very simple, difficult to tweak and error prone. For future research it would be better to replace it with a function that modifies the time step based on information on the convergence improvement and the current time step.

The grids used in this chapter are the same as in the previous chapter. In the uniform grid the elements all have the same width, Δx and is given by (6.7) for the non-uniform grid.

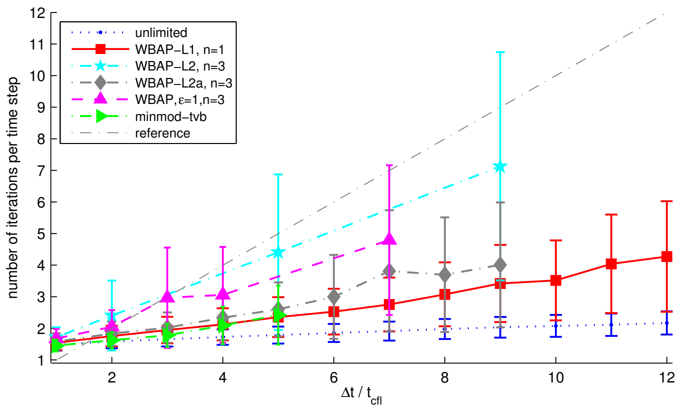


Figure 7.1: The mean number of iterations per time step with the standard deviations (divided by four) of the Burgers equation with a source term *independent* of the solution for different time step sizes compared with different limiters on a *uniform* grid ($N = 100$) with $p = 1$ at $t_{\text{final}} = 22$ using the backward Euler time integration method. A reference line is included to denote the number of time steps needed for an explicit method with $\Delta t = \Delta t_{\text{CFL}}$

7.1 Burgers equation with source term independent of the solution

The first problem is given by: [7]

$$u_t + \left(\frac{u^2}{2}\right)_x = \sin x \cos x, \qquad x \in [0, \pi], t \geq 0 \,, \tag{7.1}$$

with homogeneous Dirichlet boundary conditions $u(0, t) = 0 = u(\pi, t)$ and initial condition:

$$u(x, 0) = \beta \sin x \,. \tag{7.2}$$

The solutions for this problem are $\sin x$ and $-\sin x$, but the entropy solution is a combination of both connected with a shock when $|\beta| < 1$. We take $\beta = 0.5$. The shock is then located at

$$\pi - \arcsin\sqrt{1 - \beta^2} = \frac{2}{3}\pi \approx 2.0944 \,.$$

In Figures 7.1 and 7.2 the number of iteration steps is plotted as a function of the time steps for a uniform, respectively, non-uniform grid. For the uniform grid the WBAP-L1 limiter is the best limiter as it is the only limiter working for all time steps. The alternative WBAP-L2 limiter is a good second and the other limiters are unreliable as they do not give a result for all time steps. Minmod-TVB does not converge in the Newton method for $\Delta t > 5\Delta t_{\text{CFL}}$. However, for non-uniform grids it is the only reasonable limiter.

In Figure 7.3 the convergence to the steady state solution of the different methods for $\Delta t = 2\Delta t_{\text{CFL}}$ is shown for uniform and non-uniform grids. The convergence rates are considerably different. For the uniform grid the convergence is the same for all limiters and reaches the numerical limit of double floating point precision. For the non-uniform grid convergence is different. All methods do not converge to the numerical limit of double floating point precision,

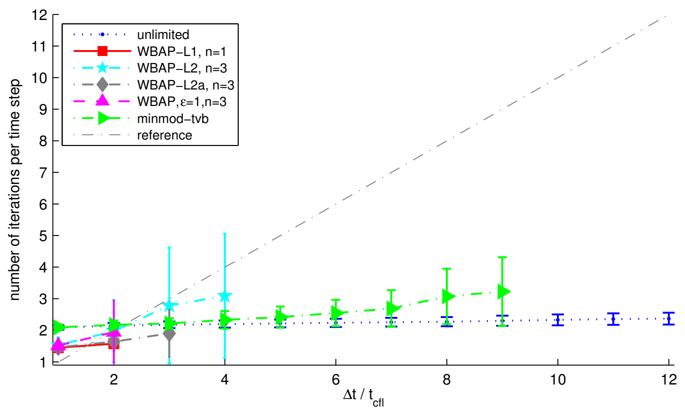
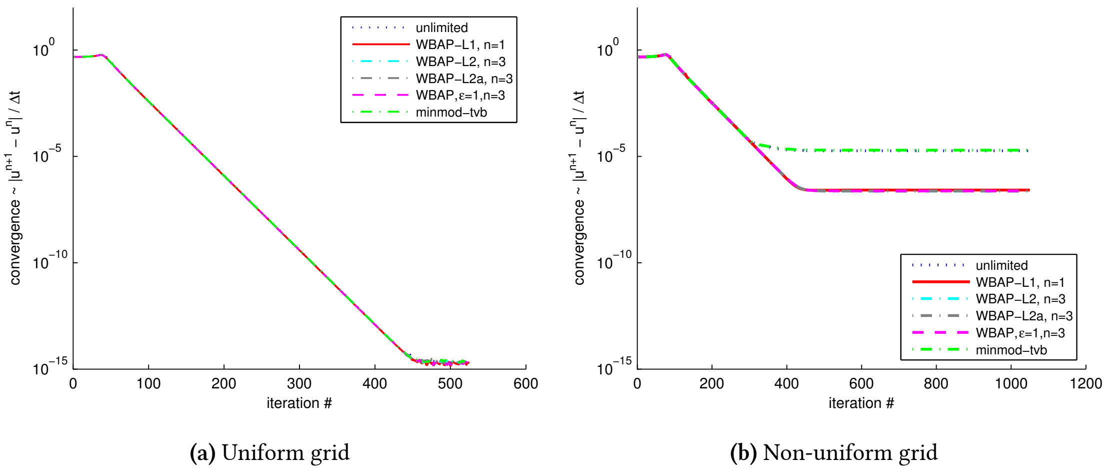
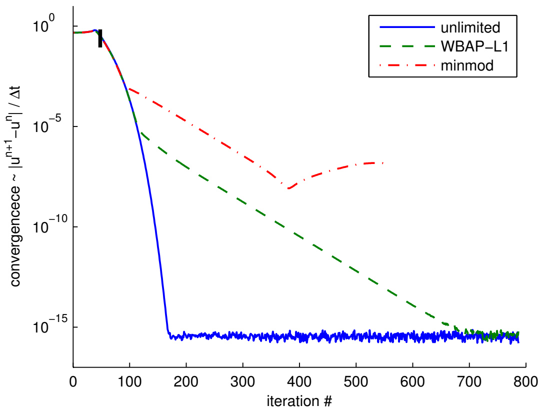


Figure 7.2: The mean number of iterations per time step with the standard deviations (divided by four) of the Burgers equation with a source term *independent* of the solution for different time step sizes compared with different limiters on a *non-uniform* grid ($N = 100$) with $p = 1$ at $t_{\text{final}} = 22$ using the backward Euler time integration method. A reference line is included to denote the number of time steps needed for an explicit method with $\Delta t = \Delta t_{\text{CFL}}$

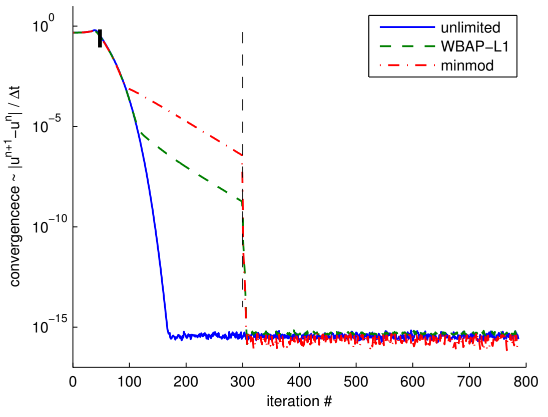


(a) Uniform grid **(b)** Non-uniform grid

Figure 7.3: Convergence of the steady state solution of the Burgers equation with a source term *independent* of the solution for a uniform and non-uniform grid ($N = 100$) with $p = 1$, $\Delta t = 2\Delta t_{\text{CFL}}$ at $t_{\text{final}} = 22$ using the backward Euler time integration method.



(a) Unlimited growth of Δt



(b) Limited growth of Δt

Figure 7.4: Convergence of the steady state solution of the Burgers equation with a source term *independent* of the solution for a uniform grid ($N = 100$) with $p = 1$, $\Delta t = 2\Delta t_{\text{CFL}}$ multiplied with 1.02 from $t = 2.0$ (black bar) onwards (a) and till 300 iterations (b, designated with dashed line). Time integration is done via the backward Euler method time integration method.

but the WBAP limiters converge two orders more. This explains why the minmod-TVB limiter and the unlimited case have better convergence properties in the Newton method for the non-uniform grid as shown in Figure 7.2.

As a last test case we increase the time step with a constant factor (1.02) in every iteration after some time. In Figure 7.4 the convergence to the steady state solution for the unlimited and limited solutions is shown for $N = 100$ and $\Delta t = 2\Delta t_{\text{CFL}}$ on a uniform grid. We see exponential convergence for the unlimited case. For the WBAP-L1 and minmod-TVB limiter it starts with exponential convergence but stops at some point and reduces to a second order convergence. If we stop the increase in time step, the convergence increases again to the same exponential rate (only requiring 5 iterations to obtain machine precision convergence) as the unlimited case.

In Figure 7.5 the unlimited and limited solutions using the minmod-TVB and WBAP limiters are shown. All methods have the jump location at the right position and both limiters remove the oscillations. The minmod-TVB and WBAP limiter obtain almost similar results. The WBAP limiter is slightly less dissipative.

7.2 Burgers equation with source term depending on the solution

The second problem is given by: [7]

$$u_t + \left(\frac{u^2}{2}\right)_x = -\pi\cos(\pi x)\,u, \qquad x \in [0,1], t \geq 0\ , \tag{7.3}$$

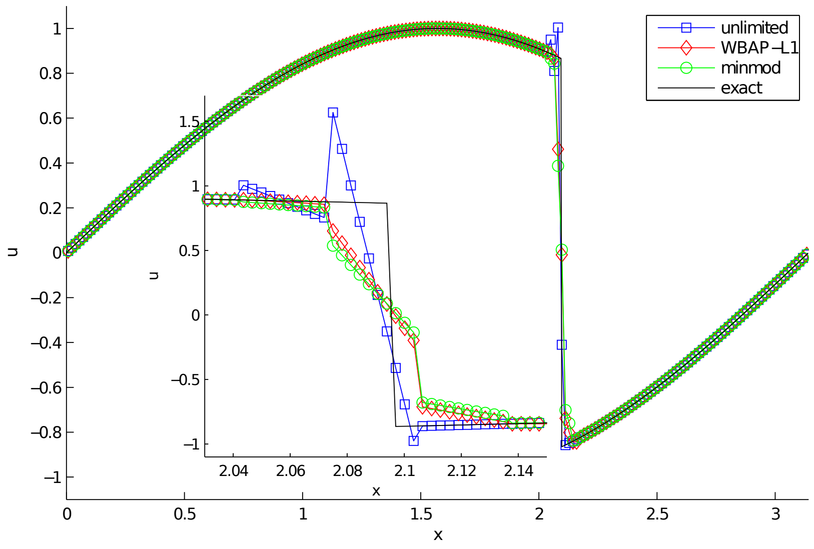


Figure 7.5: The steady state solution of the Burgers equation with source term *independent* of the solution compared with different limiters on a *uniform* grid ($N = 100$) with $p = 1$ using the backward Euler time integration method. The solution is obtained by increasing the time step for a limited number of time steps.

with Dirichlet boundary conditions $u(0,t) = 1$ and $u(1,t) = -1/10$. This problem has two steady state solutions with shocks:

$$u(x,\infty) = \begin{cases} 1 - \sin \pi x & \text{if } 0 \le x < x_s \\ -\frac{1}{10} - \sin \pi x & \text{if } x_s \le x < 1 \end{cases}, \tag{7.4}$$

where $x_s = 0.1486$ or $x_s = 0.8514$. Both solutions are entropy solutions and satisfy the Rankine-Hugoniot jump conditions, but only the solution with the shock at 0.1486 is stable for small perturbations. We will use the same initial condition as in [7]:

$$u(x,0) = \begin{cases} 1 & \text{if } 0 \le x < \frac{1}{2}, \\ -\frac{1}{10} & \text{if } \frac{1}{2} \le x < 1, \end{cases} \tag{7.5}$$

where the initial jump is located in the middle of the positions of the shock in the two admissible steady state solutions.

In Figures 7.6 and 7.7 the number of iteration steps are plotted as a function of the time step for uniform, respectively, non-uniform grids. The WBAP-L1 limiter is the best for the uniform case with the alternative WBAP-L2 limiter as a closer second than the previous test. The WBAP-L1 limiter actually improves the time step size compared to the unlimited case. For the non-uniform grid we see again that the minmod-tvb limiter is a slightly better candidate than any of the WBAP limiters.

The convergence to the steady state solution of the methods for $\Delta t = 2\Delta t_{\text{CFL}}$, as plotted in Figure 7.3, is almost the same. Both reach the numerical limit of double floating point precision. A small difference is that for the non-uniform grid the steady state oscillates more.

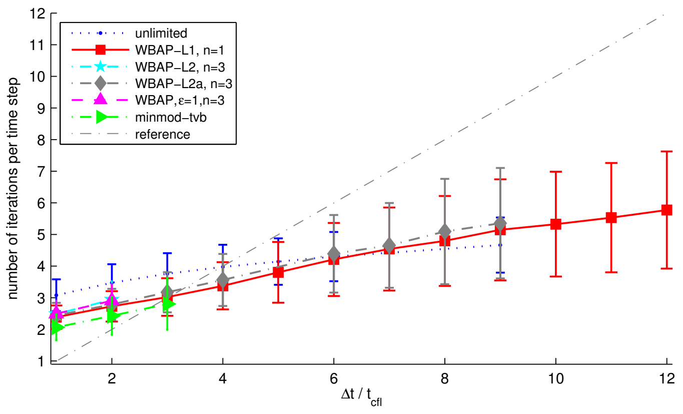


Figure 7.6: The mean number of iterations per time step with the standard deviations (divided by four) of the Burgers equation with a source term *depending* on the solution for different time step sizes compared with different limiters on a *uniform* grid ($N = 100$) with $p = 1$ at $t_{\text{final}} = 22$ using the backward Euler time integration method. A reference line is included to denote the number of time steps needed for an explicit method with $\Delta t = \Delta t_{\text{CFL}}$

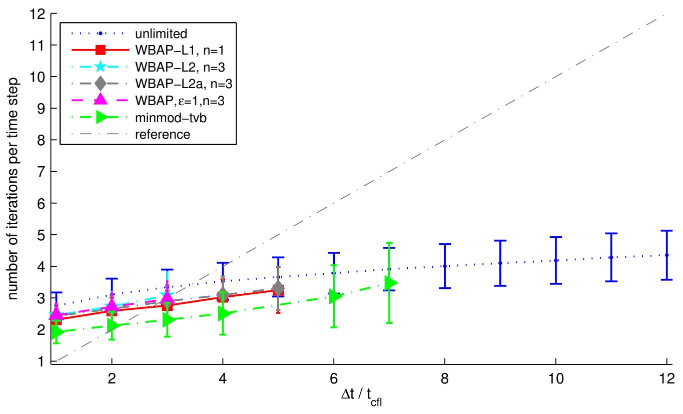
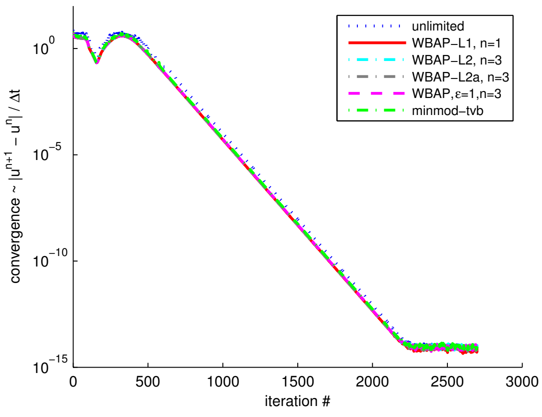
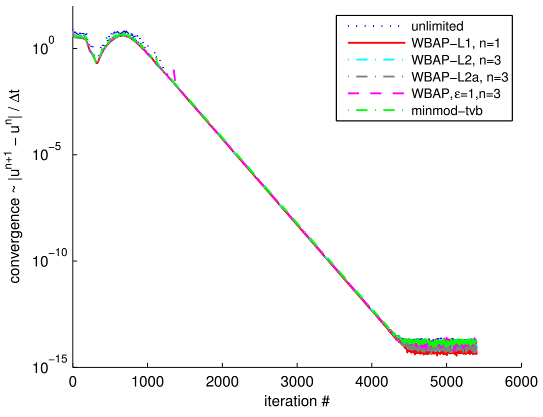


Figure 7.7: The mean number of iterations per time step with the standard deviations (divided by four) of the Burgers equation with a source term *depending* on the solution for different time step sizes compared with different limiters on a *non-uniform* grid ($N = 100$) with $p = 1$ at $t_{\text{final}} = 22$ using the backward Euler time integration method. A reference line is included to denote the number of time steps needed for an explicit method with $\Delta t = \Delta t_{\text{CFL}}$

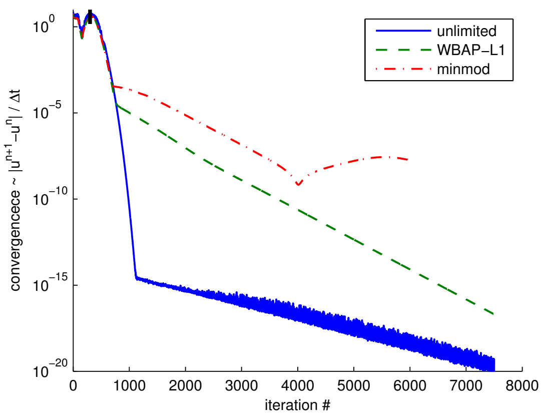


(a) Uniform grid

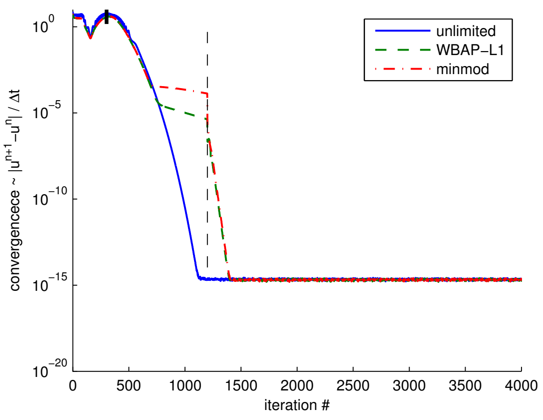


(b) Non-uniform grid

Figure 7.8: Convergence of the steady state solution of the Burgers equation with a source term *depending* on the solution for a uniform and non-uniform grid ($N = 100$) with $p = 1$, $\Delta t = 2\Delta t_{\text{CFL}}$ at $t_{\text{final}} = 22$ using the backward Euler time integration method.



(a) Unlimited growth of Δt



(b) Limited growth of Δt

Figure 7.9: Convergence of the steady state solution of the Burgers equation with a source term *depending* on the solution for a uniform grid ($N = 100$) with $p = 1$, $\Delta t = 2\Delta t_{\text{CFL}}$ multiplied with 1.002 from $t = 2.0$ (black bar) onwards (a) and till 1200 iterations (b, designated with dashed line). Time integration is done via the backward Euler time integration method.

If we let the time step grow again we see the same features as with a source term independent of the source. The convergence to steady state is shown for this test case in Figure 7.9. For unlimited growth of the time step, the minmod-TVB and WBAP-L1 limiters reduce the exponential convergence to a second order convergence after a certain time step size. When the growth is limited they achieve exponential convergence till machine precision – just for the unlimited case – after the fixation of the time step.

In Figure 7.10 the unlimited and limited solutions using the minmod-TVB and WBAP limiters are shown. All methods converge to the stable steady state and both limiters remove the

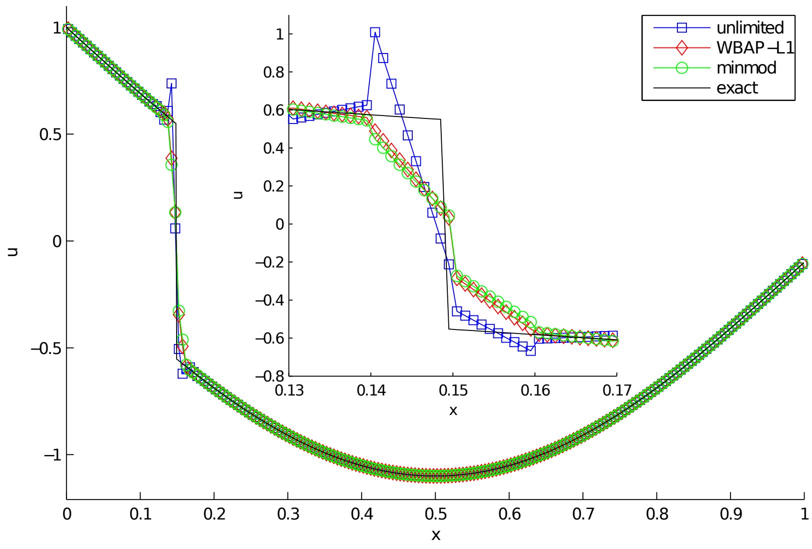


Figure 7.10: The steady state solution of the Burgers equation with source term *depending* on the solution compared with different limiters on a *uniform* grid ($N = 100$) with $p = 1$ at $t_{\text{final}} = 22$ using the backward Euler time integration method. The solution is obtained by increase the time step for a limited number of time steps.

numerical oscillations. Again the WBAP-L1 limiter is slightly less dissipative.

7.3 Conclusion

In line with the previous chapter we conclude that the WBAP-L1 limiter and the alternative WBAP-L2 limiter show improved convergence rates in the Newton method than the other limiter variants. The convergence rate is often better than for the minmod-TVB limiter and when this is not the case, either the end result differs (first problem) or the difference is small (second problem).

Both the minmod-TVB limiter and the WBAP limiter have the same limitation regarding the time step. At a certain time step size the convergence towards steady state is second order until we fix the time and very fast convergence to machine precision is obtained. This time step size is larger for the WBAP-L1 limiter than the minmod-TVB limiter. All solutions converge to the stable steady state, but this could be a problem for solutions that have multiple steady states closer to each other.

Chapter 8

Multiple steady states in a channel contraction

8.1 Introduction

This chapter deals with a more realistic experiment considering two dimensional shallow water flows. In many channels one can find a contraction. In this case the domain consists of a uniform domain followed by a linear contraction into a nozzle where the width is smaller than at the point of entry. In the case of supercritical flow the contraction will result in one or more oblique hydraulic jumps. From [2] we know that for certain inflow condition two steady state solutions are possible and by including a drag coefficient another one.

If the inflow is kept constant, the system will converge to a steady state. The challenge of this problem is to include a limiter which converges to one of the multiple steady states and not switch between the different solutions when nothing changes at the inflow.

8.1.1 Model

The channel that is being modeled has a top overview as displayed in Figure 8.1. It is inclined with ζ, the inclination angle. After the contraction we broaden the channel again to ensure that the water pours out and does not affect our solution in the contraction. The velocity is denoted as $\mathbf{u} = (u, v)$, the height with h, g is the gravity constant and t time. The two dimensional shallow water equations read:

$$h_t + \big(hu\big)_x + \big(hv\big)_y = 0, \tag{8.1a}$$

$$\big(hu\big)_t + \big(hu^2 + \tfrac{1}{2}h^2 g\cos\zeta\big)_x + \big(huv\big)_y = C_d u^2 + hg\sin\zeta, \tag{8.1b}$$

$$\big(hv\big)_t + \big(huv\big)_x + \big(hv^2 + \tfrac{1}{2}h^2 g\cos\zeta\big)_y = C_d v^2\ . \tag{8.1c}$$

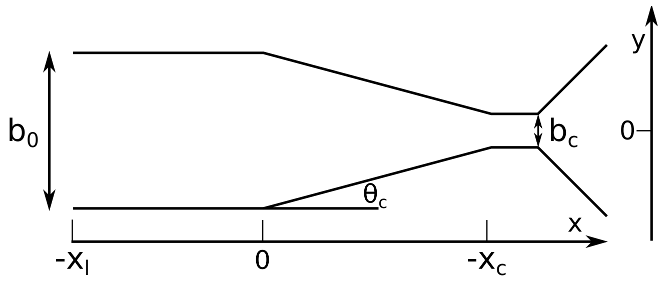


Figure 8.1: Model of the channel contraction

To reduce the number of constants we non-dimensionalize the system using the following dimensionless variables (with a hat)

$$h = h_l\hat{h}, \qquad (x, y) = b_0(\hat{x}, \hat{y}), \qquad (u, v) = \sqrt{gb_0}(\hat{u}, \hat{v}), \qquad t = \sqrt{\frac{b_0}{g}}\hat{t}\,. \tag{8.2}$$

This leads to the following dimensionless equations (dropping the hat):

$$h_t + \big(hu\big)_x + \big(hv\big)_y = 0, \tag{8.3a}$$

$$\big(hu\big)_t + \big(hu^2 + \tfrac{1}{2}\epsilon h^2\cos\zeta\big)_x + \big(huv\big)_y = \frac{C_d}{\epsilon}u^2 + h\sin\zeta, \tag{8.3b}$$

$$\big(hv\big)_t + \big(huv\big)_x + \big(hv^2 + \tfrac{1}{2}\epsilon h^2\cos\zeta\big)_y = \frac{C_d}{\epsilon}v^2\,. \tag{8.3c}$$

with $\epsilon = {}^{h_l}/_{b_0} \ll 1$. The characteristic speed is $c = \sqrt{\epsilon h\cos\zeta}$ and the Froude number is defined to be:

$$Fr = \frac{|\mathbf{u}|}{c} = \frac{|\mathbf{u}|}{\sqrt{\epsilon h\cos\zeta}}\,. \tag{8.4}$$

The Froude number is important because it relates the average water velocity to the characteristic wave speed. In gas dynamics the Mach number plays a similar role. For flows with a Froude number higher than 1, i.e. supercritical, the information travels slower than the water itself resulting in hydraulic jumps to discharge the water flow. The Froude number shows up in the momenta equations of (8.3) in the following way:

$$h_t + \big(hu\big)_x + \big(hv\big)_y = 0, \tag{8.5a}$$

$$\big(hu\big)_t + \left(hu^2 + \frac{|\mathbf{u}|}{2Fr^2}\right)_x + \big(huv\big)_y = \frac{C_d}{\epsilon}u^2 + h\sin\zeta, \tag{8.5b}$$

$$\big(hv\big)_t + \big(huv\big)_x + \left(hv^2 + \frac{|\mathbf{u}|}{2Fr^2}\right)_y = \frac{C_d}{\epsilon}v^2\,. \tag{8.5c}$$

The boundary conditions are easy for supercritical flow since we can just prescribe the values at the inflow as:

$$h = h_l, \qquad u = u_l, \tag{8.6}$$

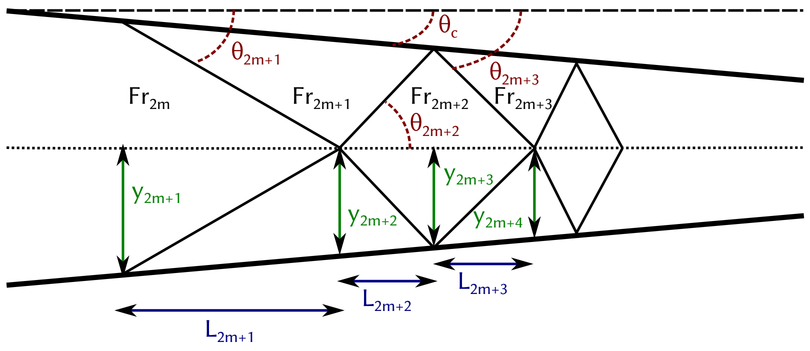


Figure 8.2: Sketch of the oblique hydraulic jumps (thin solid lines) within the contraction and the definition of some variables involved. The centerline of the channel is dashed. Channel walls are the thick lines. This is the corrected Figure 9 from Akers and Bokhove [2].

and at the outflow we just apply Neumann boundary conditions as transparent boundary conditions. For subcritical flow these boundary conditions will not work as the wave is reflected back at the inflow from the shock and the water height and momenta will blow up at the outflow. Along the wall we apply the standard no normal flow boundary condition:

$$h\mathbf{u}\cdot\hat{\mathbf{n}} = 0, \tag{8.7}$$

8.2 Oblique hydraulic jumps

In Figure 8.2 a is sketch shown with a schematic view of the oblique hydraulic jumps and the variables involved. If we assume an inviscid flow upstream with constant Froude number Fr_0, depth h_0 and speed $\mathbf{u} = U_0(1,0)$ we will observe a hydraulic jump due to the collision on the slanted walls. The oblique jumps meet symmetrically at the center and generates two new oblique jumps which hit the slanted walls and so forth. In this pattern we distinguish the odd jumps (from the walls to the center) and the even jumps (from the center to the wall). We will denote the odd jumps by $2m+1$ and the even jumps by $2m+2$.

Classical 2D theory for oblique hydraulic jumps [20] gives the following relations between the angles θ, height h, width y, velocity U and Froude number Fr assuming no inclination and no drag.

$$\frac{h_{2m+1}}{h_{2m}} = -\frac{1}{2} + \frac{1}{2}\sqrt{1 + 8Fr_{2m}^2\,\sin^2\theta_{2m+1}} = \frac{\tan\theta_{2m+1}}{\tan(\theta_{2m+1}-\theta_c)}, \tag{8.8a}$$

$$\frac{U_{2m+1}}{U_{2m}} = \frac{\cos\theta_{2m+1}}{\cos(\theta_{2m+1}-\theta_c)}, \tag{8.8b}$$

$$Fr_{2m+1}^2 = Fr_{2m}^2\frac{\cos^3\theta_{2m+1}\,\sin(\theta_{2m+1}-\theta_c)}{\cos^3(\theta_{2m+1}-\theta_c)\,\sin\theta_{2m+1}} \tag{8.8c}$$

and

$$\frac{h_{2m+2}}{h_{2m+1}} = -\frac{1}{2} + \frac{1}{2}\sqrt{1 + 8Fr^2_{2m+1}\sin^2(\theta_{2m+2} + \theta_c)} = \frac{\tan(\theta_{2m+2} + \theta_c)}{\tan\theta_{2m+2}}, \tag{8.9a}$$

$$\frac{U_{2m+2}}{U_{2m+1}} = \frac{\cos(\theta_{2m+2} + \theta_c)}{\cos\theta_{2m+2}}, \tag{8.9b}$$

$$Fr^2_{2m+2} = Fr^2_{2m+1}\frac{\cos^3(\theta_{2m+2} + \theta_c)\,\sin\theta_{2m+2}}{\cos^3\theta_{2m+2}\,\sin(\theta_{2m+2} - \theta_c)}\ . \tag{8.9c}$$

Note that (8.9) equals (8.8) by replacing $\theta_{2m+2} + \theta_c$ by θ_{2m+1}. To find the angles we have to solve (8.8a) and (8.9a). This is the most complicated part which will be explained first. Next we will describe the full algorithm to find all variables.

Using the following trigonometric identities and definitions

$$\cos\theta = \sqrt{1 - \sin^2\theta}, \quad \tan\theta = \sqrt{\frac{\sin^2\theta}{1 - \sin^2\theta}}, \quad \tan(\theta_1 + \theta_2) = \frac{\tan\theta_1 + \tan\theta_2}{1 - \tan\theta_1\tan\theta_2}, \tag{8.10}$$

$$F = 8Fr, \qquad S = \tan\theta_c, \qquad Z = \sin^2\theta_{2m+1}, \tag{8.11}$$

we can rewrite[1] (8.8a) such that the solution of the following polynomial

$$\begin{aligned}&\big[1+S^2\big]Z^5 + \big[-5(1+S^2)\big]Z^4 + \big[(1+2S^2)(1-F)+1)\big]Z^3 + \big[(1+S^2)(14+6F) - F\big]Z^2 \\ &+ \big[FS^2(F+2) - 3(S^2+F+1)\big]Z + \big[-FS^2(F+6) - 9(S^2+F+1)\big] = 0\ ,\end{aligned} \tag{8.12}$$

leads to the solution of the angle:

$$\theta_{2m+1} = \arcsin\sqrt{\frac{Z^2-1}{F}}\ . \tag{8.13}$$

The polynomial equation (8.12) can be solved by an algorithm that find the roots of a polynomial, which is available in many numerical libraries. We have used the `roots` function of Matlab. If the solution to the polynomial root problem has one or more real solutions, we have a solution to the actual problem. Likewise for the even equation (8.9a) we can solve the same polynomial equation (8.12) only now with $Z = \sin^2(\theta_{2m+2} + \theta_c)$ and obtain the new angle by:

$$\theta_{2m+2} = \arcsin\sqrt{\frac{Z^2-1}{F}} - \theta_c\ . \tag{8.14}$$

The algorithm to find the variables for the oblique hydraulic jumps can be described in the following way. We start with an upstream Fr_0 and the known half-channel width $y_1 = {}^{b_0}/_2$. So we start with the even 'shock' as known, i.e. Fr_{2m} and y_{2m+1}. Solving θ_{2m+1} using (8.12) and (8.13), we can determine L_{2m+1} and y_{2m+2} by:

$$L_{2m+1} = \frac{y_{2m+1}}{\tan\theta_{2m+1}}. \qquad y_{2m+2} = L_{2m+1}\big(\tan\theta_{2m+1} - \tan\theta_c\big)\ . \tag{8.15}$$

[1]With a little help from Maple.

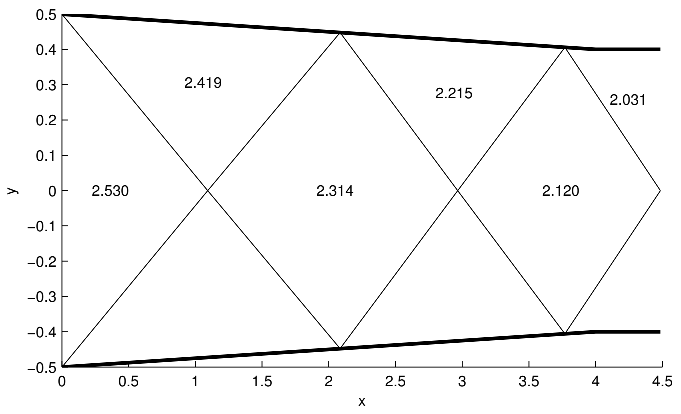


Figure 8.3: The two dimensional analytical steady state solution for the channel flow in terms of Froude numbers. Thin lines denote the oblique hydraulic jumps and the thick lines are the walls. The numbers in the regions are the local Froude number of the flow.

The Froude number Fr_{2m+2} can be calculated via (8.8c). Next we solve the even angle θ_{2m+2} using (8.12) and (8.14) and determine L_{2m+2} and y_{2m+2} by:

$$L_{2m+2} = \frac{y_{2m+1}}{\tan\theta_{2m+1} + \tan\theta_c}. \qquad y_{2m+3} = L_{2m+2}\tan\theta_{2m+2}\,. \tag{8.16}$$

The Froude number Fr_{2m+2} can be calculated via (8.9c). We have enough information to calculate the odd shock again, and are at the beginning of the loop again. This has to be continued until the total length is larger than the contraction length or the polynomial equation (8.12) has no real solution, i.e. there is no angle. This happens when the Froude number at the left side of the hydraulic jump is close to 1.

8.3 Simulations

In this section we will look at two different things. First, we will compare the WBAP limiter to other known limiters to see how it performs in a two dimensional DG setup. We will use the WBAP-L1 limiter since it is the most promising limiter from the one dimensional results. Second, we investigate for one mesh how many iterations in the Newton process is needed with or without the limiter for different time steps.

8.3.1 Setup

In all our simulations we will use the following parameters:

$$C_d = 0, \qquad \epsilon = 0.1, \qquad h_l = 1.0, \qquad u_l = 0.8, \qquad \zeta = 0\,. \tag{8.17}$$

Using (8.4) we come to a Froude number of 2.530 at the inflow. The oblique hydraulic jumps in the steady state solution for these input variables are shown in Figure 8.3.

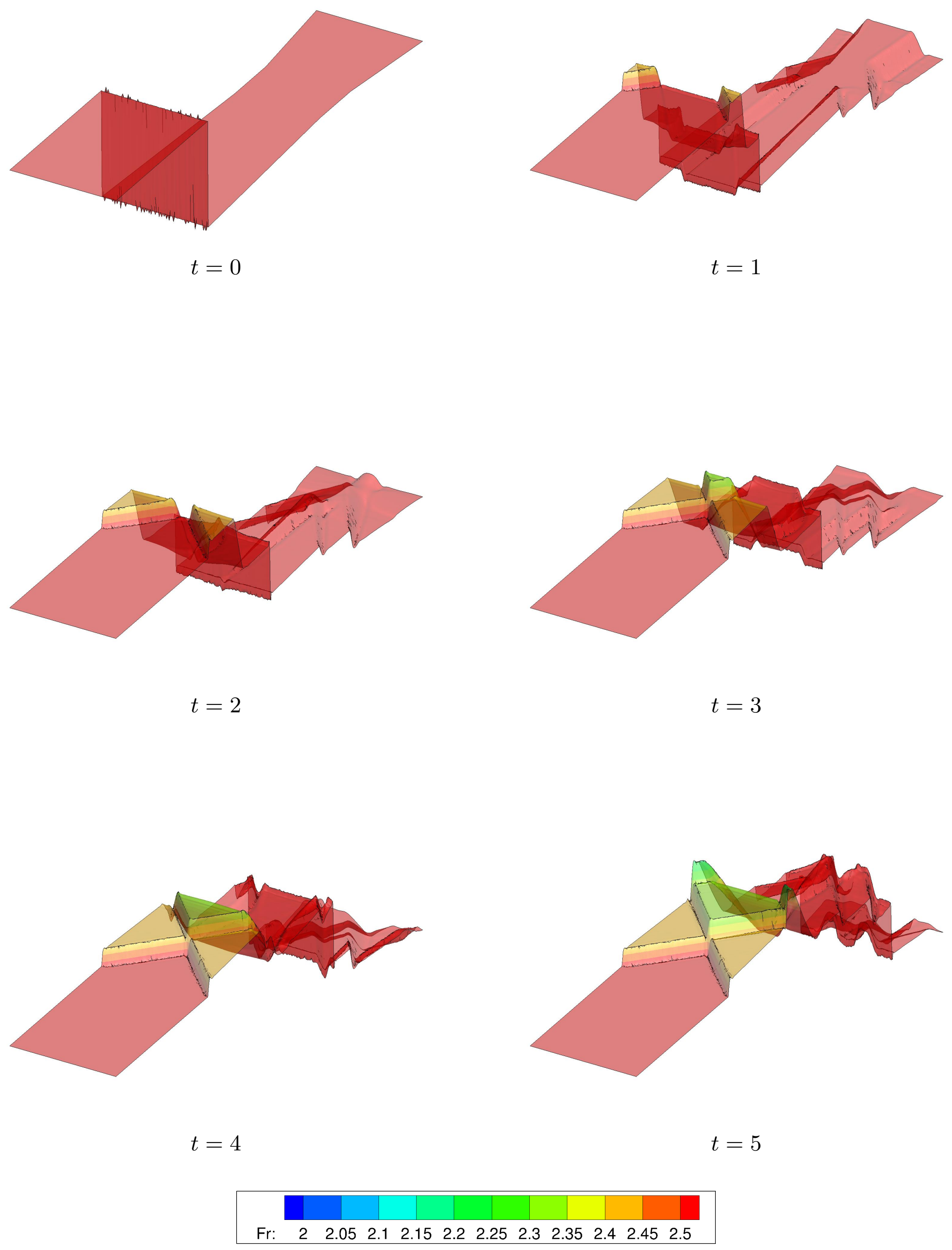


Figure 8.4: Snapshots of the height and Froude number (in color) of a simulation using the explicit Runge-Kutta time integration method, $p = 1$, $N_{\text{el}} = 44138$ and no limiting. Part 1 of 2. See Figure 8.5 for $t > 5$.

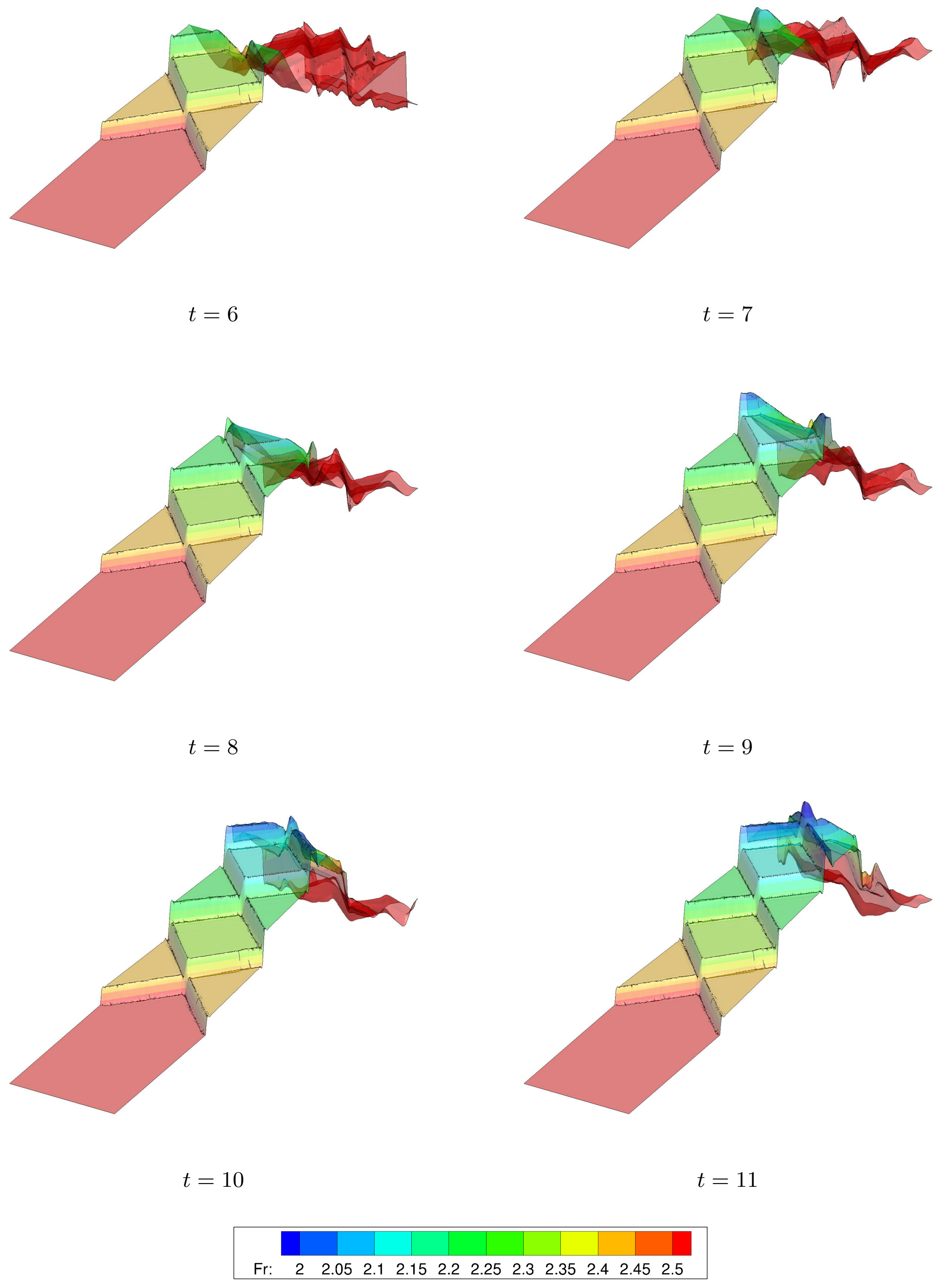


Figure 8.5: Snapshots of the height and Froude number (in color) of a simulation using the explicit Runge-Kutta time integration method, $p = 1$, $N_{\text{el}} = 44138$ and no limiting. Part 2 of 2. See Figure 8.4 for $t \leq 5$.

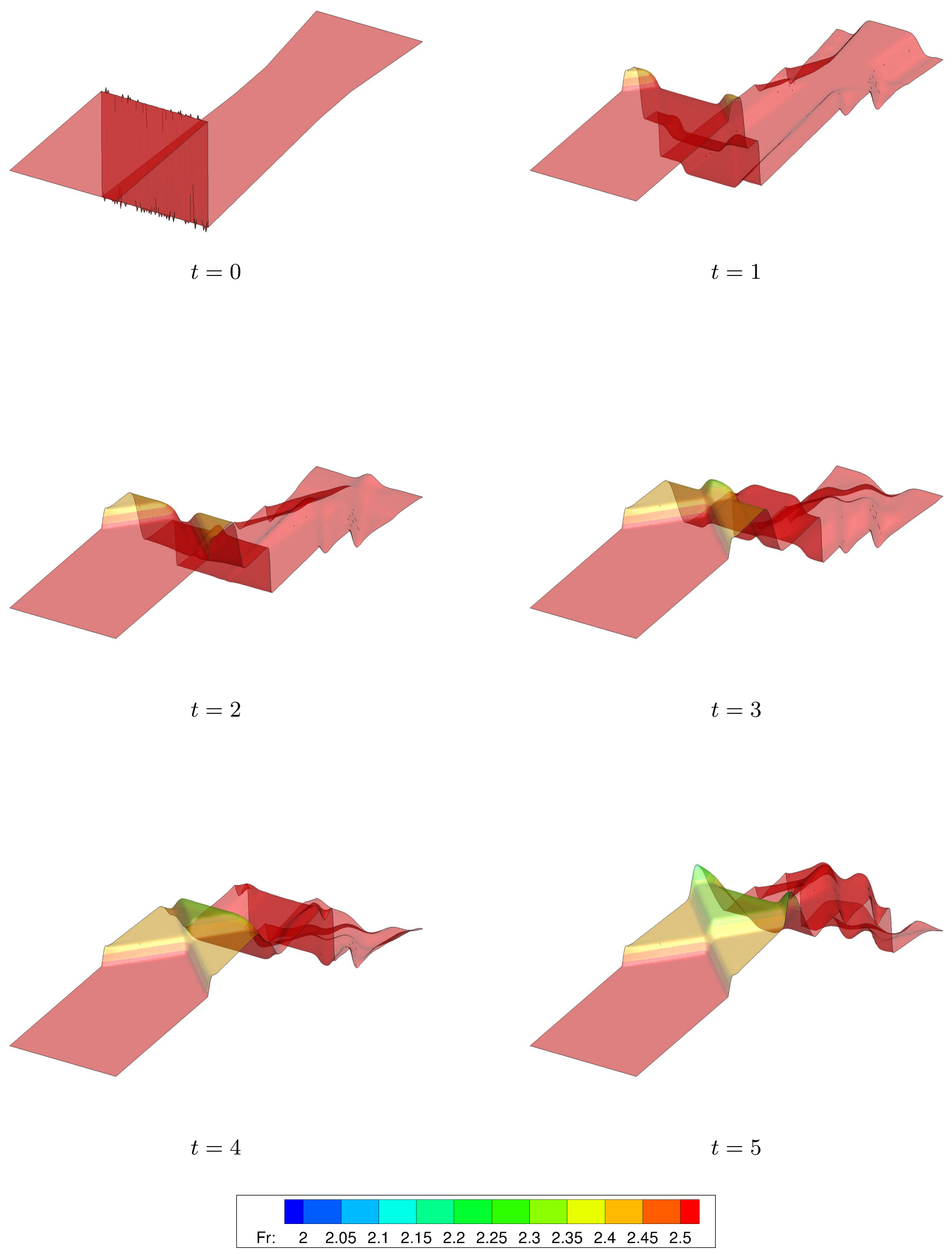


Figure 8.6: Snapshots of the height and Froude number (in color) of a simulation using the explicit Runge-Kutta time integration method, $p = 1$, $N_{\text{el}} = 44138$ and with the WBAP-L1 limiter. Part 1 of 2. See Figure 8.7 for $t > 5$.

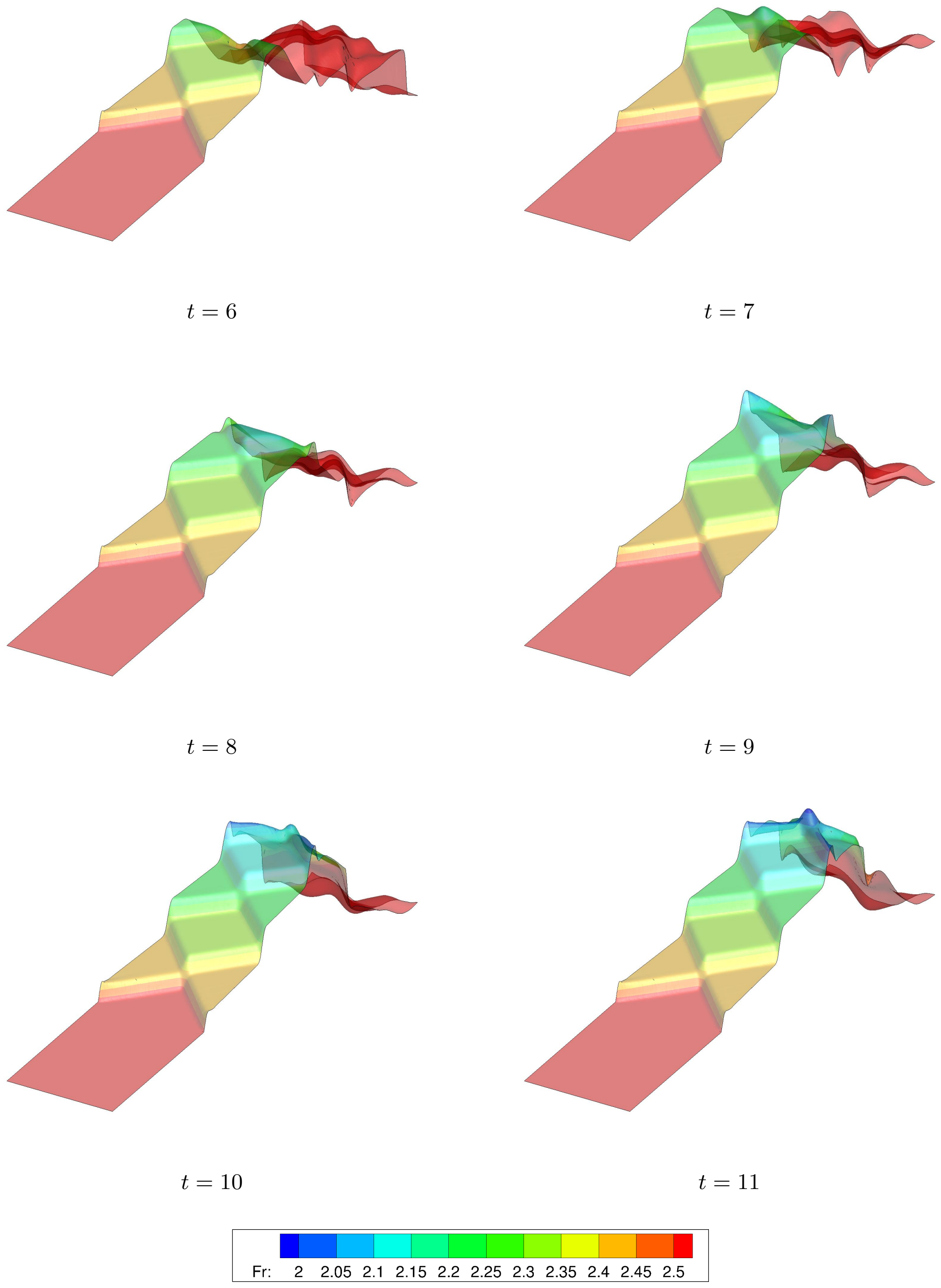


Figure 8.7: Snapshots of the height and Froude number (in color) of a simulation using the explicit Runge-Kutta time integration method, $p = 1$, $N_{\text{el}} = 44138$ and with the WBAP-L1 limiter. Part 2 of 2. See Figure 8.6 for $t \leq 5$.

As initial condition we take

$$h_0(x,y) = \begin{cases} h_l & \text{if } x < 0, \\ \frac{1}{2}h_l & \text{if } x \geq 0, \end{cases} \qquad u_0(x,y) = u_l, \qquad v_0(x,y) = 0\,. \tag{8.18}$$

We cannot take the inital water height or x-velocity to be zero at the contraction, since the code will divide by zero at some point or takes the square root of a negative number. The initial DG coefficients can be calculated via the inner product of our trial functions, or just by using the corresponding values of the variables at the midpoints and vertices of the elements.

In Figures 8.4 and 8.5 an example run with $N_{\text{el}} = 44138$ using the third order SSP Runge-Kutta method without any limiter is shown for several points of time up to $t = 12$ after which not much development is visible. In Figures 8.6 and 8.7 the same example run is displayed, but now with the WBAP-L1 limiter. One can see that no limiter is used resulting in rough edges on the hydraulic jumps which are not present in the results with the WBAP-L1 limiter. However, the hydraulic jumps in the results with the WBAP-L1 limiter contain more dissipation.

8.3.2 Comparison with other limiters

We compare the WBAP-L1 limiter with the limiters of Barth and Jespersen [4], Venkatakrishnan [33] and Michalak and Ollivier-Gooch [25]. To shorten the computational time and because we didn't calculate the Jacobian of these other limiters, we have used the explicit third order SSP Runge Kutta scheme (2.31). The results are displayed in Figure 8.8 (cross section $x = 1.5$), Figure 8.9 (cross section $y = 0$) and Figure 8.10 (cross section $y = 0.2$).

In all cross sections the WBAP limiter follows the Barth-Jespersen limiter closely, which outperforms the limiters of Venkatakrishnan and Michalak in terms of dissipation and accuracy. Also, the limiters of Venkatakrishnan and Michalak require significantly more computing time.

In the $x = 1.5$ cross section the WBAP limiter gives clearly the best result since it attains the correct value of the shock in contrast to the other limiters. Note that this limiter is not positivity preserving. In the $y = 0$ cross sections one can see a small undershoot of the WBAP limiter.

If we look at the convergence in Figure 8.11, we see that that in terms of convergence rate the WBAP limiter does a better job than the Barth-Jespersen limiter. The sudden increase at the end of the domain can be contributed to the fact that our code adapts the last time step, such that the total time is exactly the desired final time. It is the same phenomena that was also seen in Chapter 7.

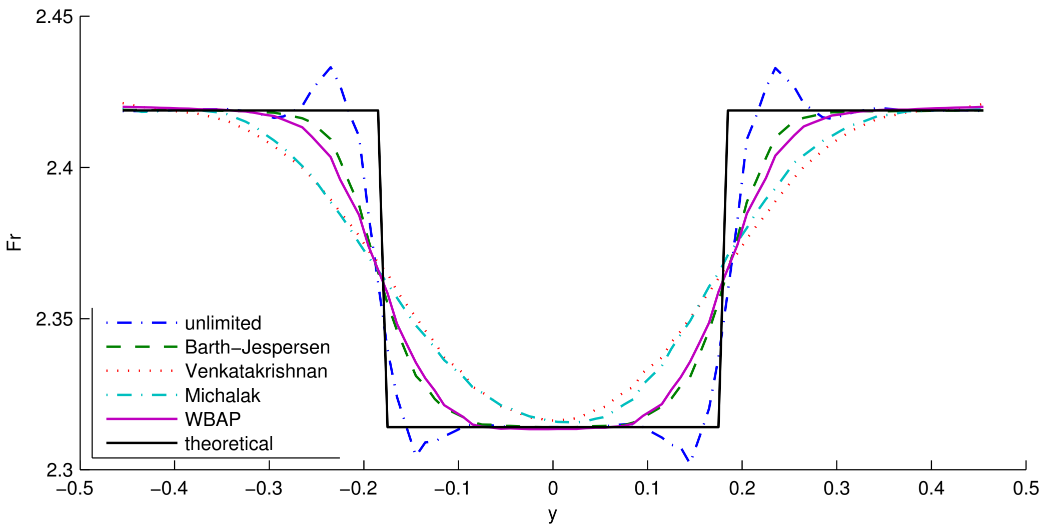


Figure 8.8: Cross section of the final solution on a unstructured regular grid ($N_{\text{el}} = 22726$) with $p = 1$, $\Delta t = 0.2\Delta t_{\text{CFL}}$ and $t_{\text{final}} = 20$ using the third order SSP Runge Kutta scheme for different limiters and the analytic solution at $x = 1.5$.

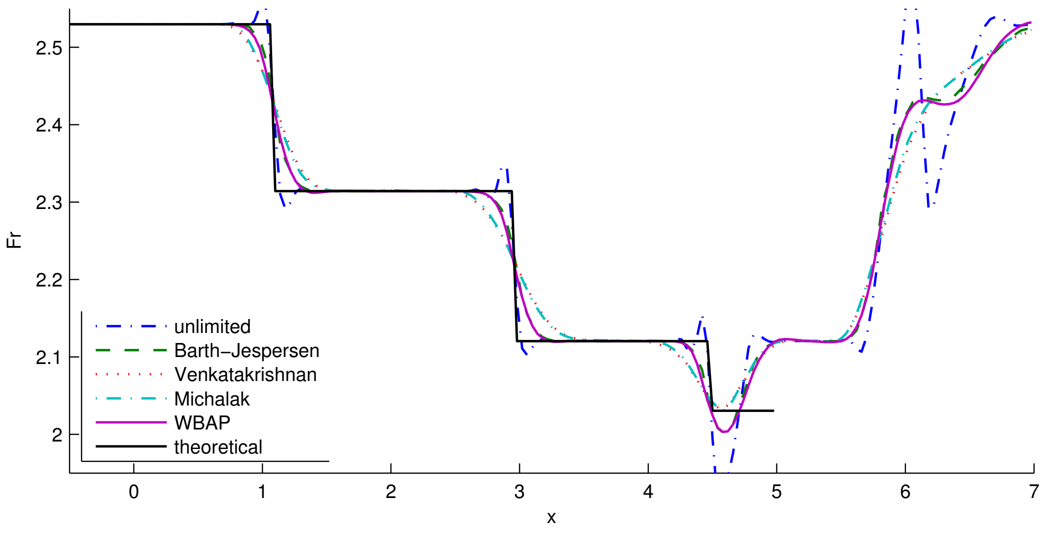


Figure 8.9: Cross section of the final solution on a unstructured regular grid ($N_{\text{el}} = 22726$) with $p = 1$, $\Delta t = 0.2\Delta t_{\text{CFL}}$ and $t_{\text{final}} = 20$ using the third order SSP Runge Kutta scheme for different limiters and the analytic solution at $y = 0$.

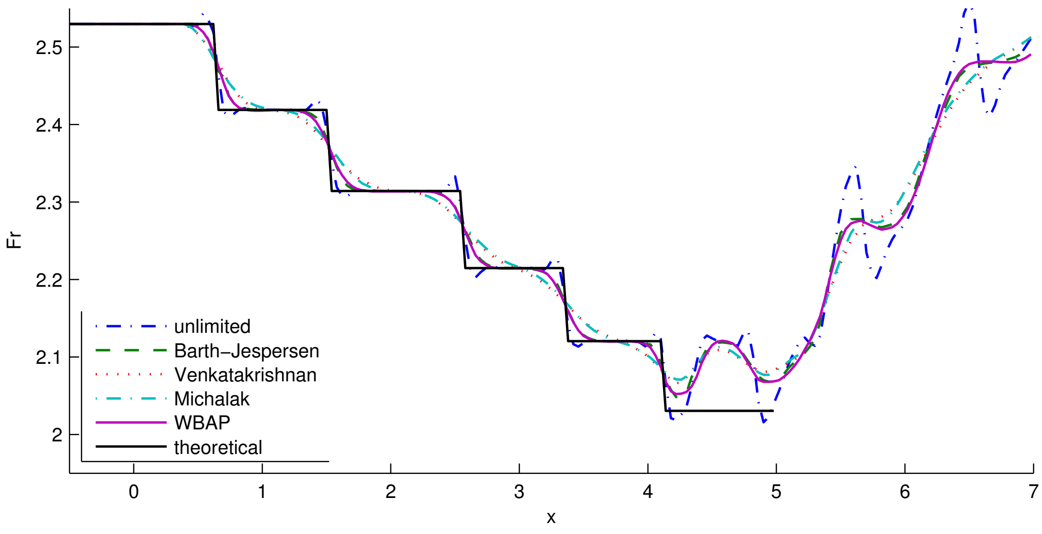


Figure 8.10: Cross section of the final solution on a unstructured regular grid ($N_{\text{el}} = 22726$) with $p = 1$, $\Delta t = 0.2\Delta t_{\text{CFL}}$ and $t_{\text{final}} = 20$ using the third order SSP Runge Kutta scheme for different limiters and the analytic solution at $y = 0.2$.

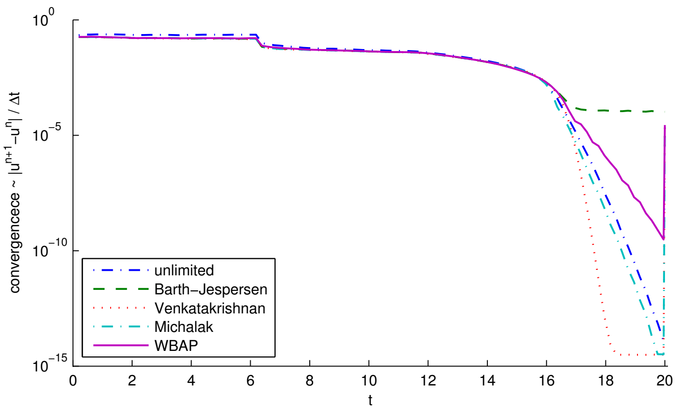


Figure 8.11: Convergence to the steady state solution on a unstructured regular grid ($N_{\text{el}} = 22726$) with $p = 1$, $\Delta t = 0.2\Delta t_{\text{CFL}}$ and $t_{\text{final}} = 20$ using the third order SSP Runge Kutta scheme for different limiters and the analytic solution.

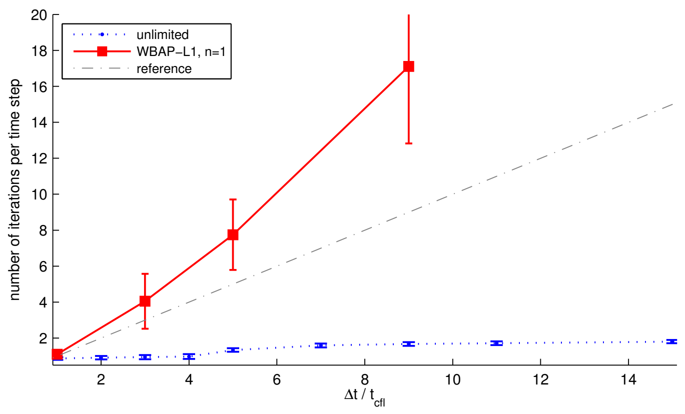


Figure 8.12: The mean number of iterations per time step with the standard deviations (divided by five) for the time steps $1, 2, 3, 5, 7, 9, 11, 13, 15$ times Δt_{CFL}, without and with the WBAP-L1 limiter on a unstructured regular grid ($N_{\text{el}} = 986$) with $p = 1$ and $t_{\text{final}} = 20$ using the backward Euler time integration method. A reference line is included to denote the number of time steps needed for an explicit method with $\Delta t = \Delta t_{\text{CFL}}$

8.3.3 Convergence in the iterative method as function of the time step

In Figure 8.12 the number of iterations per time step is plotted as a function of the time step once more. Due to time constraint this is only done for the WBAP-L1 limiter and the unlimited case, although the implementation could be easily changed to other variants. The results are worse than for the Euler equations and the steady state problems with the Burgers equation. For the unlimited case the simulation with $\Delta t = 13\Delta t_{\text{CFL}}$ halted and could not be completed. For the WBAP-L1 limiter only four of the ten different time steps succeeded.

The convergence to the steady state is plotted in Figure 8.13 for three different time steps. For $\Delta t = 3\Delta t_{\text{CFL}}$ the convergence to steady state is different with the WBAP-L1 limiter, but for larger time steps the convergence is the same with or without the WBAP-L1 limiter. Although the convergence towards steady state for the larger time step is only 10^{-3}, the results, whether or not the convergence in the Newton methods halts, will not change by a longer final time. The difficulty in convergence in the Newton method is in that time region – i.e. for convergence smaller than 10^{-3} – not present anymore.

8.4 Conclusions

This chapter could use some more simulations. One particular object of interest is how the convergence to the analytic solution will be when the number of elements is increased. A conclusion that can be made is that the WBAP-L1 limiter is comparable to the Barth-Jespersen limiter in dissipation, but better in converging to the steady state solution. For use in a implicit time integration method it is less reliable than for the one dimensional equation as the problem is harder.

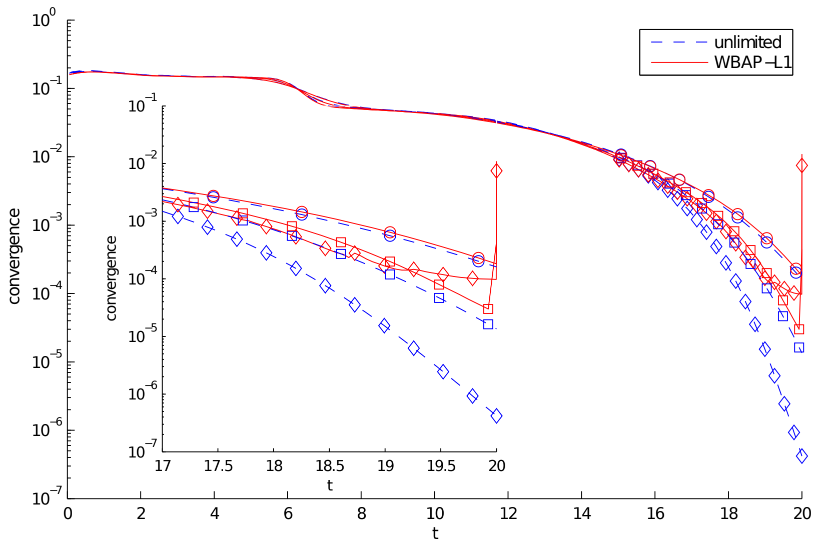


Figure 8.13: The convergence towards the steady state method for the time steps 3.0 (diamond), 5.0 (square) and 7.0 (circle) times Δt_{CFL} without and with the WBAP-L1 limiter on a unstructured regular grid ($N_{\mathrm{el}} = 986$) with $p = 1$ and $t_{\mathrm{final}} = 20$ using the backward Euler time integration method.

Chapter 9

Discussion & Conclusions

In this thesis we investigated the WBAP limiter and different variants of it. We were interested in the WBAP limiter because it is a smooth limiter in contrast to most known limiters which contain non-smooth switches. This property should be beneficial in iterative methods that are used to solve the non-linear algebraic equations resulting from an implicit time integration method.

We have successfully extended the WBAP limiter developed for finite volume methods to the DG method in one and two dimensions. All variants performed comparable to standard limiters known from the literature. in the limiting of numerical oscillations In the one dimensional setting this was the minmod-TVB limiter, and in the two dimensional setting this is the Barth-Jespersen limiter. In the accuracy test the order of magnitude of the errors from the different versions of WBAP limiter were comparable to the minmod-TVB limiter for the non-uniform grid and in absolute value they performed better. The WBAP limiter also performed better in converging to the steady state solution in the two dimensional channel contraction than the Barth-Jespersen limiter.

Although the different WBAP limiter variants perform comparable in accuracy or limiting capabilities, they do not with respect to the convergence rate of the Newton method used to solve the non-linear algebraic equations. There is a clear indication that the WBAP-L1 limiter is the preferred limiter to be used. Due to the fact that the alternative WBAP-L2 limiter is also better than the other variants we have to attribute this to the condition that those limiters set the gradient to zero when not all arguments have the same sign. In Figure 4.1 those two limiters are in the Sweby region.

Our hypothesis that smoothness would be beneficial for the iterative methods is therefore not confirmed. The limiters significantly reduce the convergence rate of the Newton method. The only exception is when the numerical oscillations and time step are large enough to prevent the Newton method without limiter to converge. It is difficult to determine the origin of this problem, but it might be attributed it to the extra complexity of the non-linear algebraic equations resulting from the limiters.

9.1 Future work

Due to time constraints on the project not everything in the two dimensional problem could be investigated. One thing to look closer into is how the limiter affects the accuracy in comparison with the analytic results when the number of elements is increased. Some parts of the numerical implementation could be optimized by using an existing numerical library. Especially the custom implementation for the matrix multiplication of two sparse matrices was slow compared to the Matlab implementation and is the main bottleneck in the simulations.

The biggest problem in this research was the convergence of the damped Newton method when a limiter was used. Since smoothness was not found to have a large impact one should look for other Newton methods. In the current setup the limiter is being implemented as an $\mathbb{R}^n \to \mathbb{R}^n$ function applied to the DG formulation. Without the limiter the convergence properties are much better. This leads to the idea of using a semi-smooth Newton method [31] instead and use the limiter as a constraint in solving the algebraic equations from the DG operator without a limiter.

In the convergence of the steady state problems we multiplied the time step by a fixed factor. The convergence in the Newton method is very sensitive to the size of this multiplier when a limiter is used and one could look into the literature to see how the limiter reacts to more sophisticated algorithms. In particular, it would be interesting if such algorithms achieve the same exponential convergence as the Newton method without limiter and do not reduce to second order convergence.

Bibliography

[1] M. Abramowitz and I. A. Stegun. *Handbook of mathematical functions with formulas, graphs, and mathematical tables.* U.S. Department of Commerce, Dec. **1972**. ISBN 978-0318117300.

[2] B. Akers and O. Bokhove. Hydraulic flow through a channel contraction: multiple steady states. *Physics of fluids*, 20(5):056601, **2008**. ISSN 1070-6631. doi:10.1063/1.2909659.

[3] D. Arnold, F. Brezzi, B. Cockburn, and L. Marini. Unified analysis of discontinuous Galerkin methods for elliptic problems. *SIAM Journal on Numerical Analysis*, 39(5):1749–1779, **2002**. doi:10.1137/S0036142901384162.

[4] T. J. Barth and D. C. Jespersen. The design and application of upwind schemes on unstructured meshes. In *Proceedings of the 27th Aerospace Sciences Meeting*, **2002**.

[5] J. C. Butcher. *Numerical methods for Ordinary Differential Equations.* John Wiley & Sons, **2003**. ISBN 0-471-96758-0.

[6] H. Choi and J.-G. Liu. The reconstruction of upwind fluxes for conservation laws: Its behavior in dynamic and steady state calculations. *Journal of Computational Physics*, 144 (2):237–256, **1998**. ISSN 0021-9991. doi:10.1006/jcph.1998.5970.

[7] C.-S. Chou and C.-W. Shu. High order residual distribution conservative finite difference WENO schemes for steady state problems on non-smooth meshes. *Journal of Computational Physics*, 214(2):698–724, **2006**. ISSN 0021-9991. doi:10.1016/j.jcp.2005.10.007.

[8] B. Cockburn and C.-W. Shu. TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws II: General framework. *Mathematics of Computation*, 52(186):411–435, **1989**. ISSN 00255718. doi:10.1090/S0025-5718-1989-0983311-4.

[9] B. Cockburn and C.-W. Shu. Runge–Kutta discontinuous Galerkin methods for convection-dominated problems. *Journal of Scientific Computing*, 16(3):173–261, **2001**. ISSN 0885-7474. doi:10.1023/A:1012873910884.

[10] S. S. Collis. Discontinuous Galerkin methods for turbulence simulation. In *Center for Turbulence Research, Proceedings of the Summer Program*, **2002**.

[11] R. Courant, K. Friedrichs, and H. Lewy. Über die partiellen Differenzengleichungen der mathematischen Physik. *Mathematische Annalen*, 100(1):32–74, **1928**. ISSN 0025-5831. doi:10.1007/BF01448839.

[12] J. E. Dennis and R. B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. SIAM, **1996**. ISBN 978-1611971200. doi:10.1137/1.9781611971200.

[13] S. Gottlieb, C.-W. Shu, and E. Tadmor. Strong stability-preserving high-order time discretization methods. *SIAM Reviews*, 43(1):89–112, **2001**. ISSN 00361445. doi:10.1137/S003614450036757X.

[14] A. Harten. High resolution schemes for hyperbolic conservation laws. *Journal of Computational Physics*, 49(3):357–393, **1983**. ISSN 0021-9991. doi:10.1016/0021-9991(83)90136-5.

[15] A. Harten, B. Engquist, S. Osher, and S. R. Chakravarthy. Uniformly high order accurate essentially non-oscillatory schemes, III. *Journal of Computational Physics*, 71(2):231–303, **1987**. ISSN 0021-9991. doi:10.1016/0021-9991(87)90031-3.

[16] T. J. Hughes and G. M. Hulbert. Space-time finite element methods for elastodynamics: Formulations and error estimates. *Computer Methods in Applied Mechanics and Engineering*, 66(3):339–363, **1988**. ISSN 0045-7825. doi:10.1016/0045-7825(88)90006-0.

[17] G.-S. Jiang and C.-W. Shu. On a cell entropy inequality for discontinuous Galerkin methods. *Mathematics of Computation*, 62(206):531–538, **1994**. ISSN 00255718. doi:10.1090/S0025-5718-1994-1223232-7.

[18] G.-S. Jiang and C.-W. Shu. Efficient implementation of weighted eno schemes. *Journal of Computational Physics*, 126(1):202–228, **1996**. ISSN 0021-9991. doi:10.1006/jcph.1996.0130.

[19] C. M. Klaij, J. J. W. van der Vegt, and H. van der Ven. Space-time discontinuous Galerkin method for the compressible Navier-Stokes equations. *Journal of Computational Physics*, 217(2):589–611, **2006**. ISSN 0021-9991. doi:10.1016/j.jcp.2006.01.018.

[20] P. K. Kundu and I. M. Cohen. *Fluid Mechanics*. Academic Press, Feb. **2010**. ISBN 978-0123813992.

[21] W. Li and Y.-X. Ren. The multi-dimensional limiters for solving hyperbolic conservation laws on unstructured grids II: Extension to high order finite volume schemes. *Journal of Computational Physics*, 231(11):4053–4077, **2012**. ISSN 0021-9991. doi:10.1016/j.jcp.2012.01.029.

[22] W. Li, Y.-X. Ren, G. Lei, and H. Luo. The multi-dimensional limiters for solving hyperbolic conservation laws on unstructured grids. *Journal of Computational Physics*, 230(21):7775–7795, **2011**. ISSN 0021-9991. doi:10.1016/j.jcp.2011.06.018.

[23] X.-D. Liu, S. Osher, and T. Chan. Weighted essentially non-oscillatory schemes. *Journal of Computational Physics*, 115(1):200–212, **1994**. ISSN 0021-9991. doi:10.1006/jcph.1994.1187.

[24] D. W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the Society for Industrial and Applied Mathematics*, 11(2):431–441, **1963**. doi:10.1137/0111030.

[25] K. Michalak and C. F. Ollivier-Gooch. Differentiability of slope limiters on unstructured grids. In *Proceedings of the 14th Annual Conference of the Computational Fluid Dynamics Society of Canada*, **2006**.

[26] L. Qi. Convergence analysis of some algorithms for solving nonsmooth equations. *Mathematics of Operations Research*, 18(1):227–244, **1993**. doi:10.1287/moor.18.1.227.

[27] W. Reed and T. Hill. Triangular mesh methods for the neutron transport equation. Technical Report LA-UR–73-479, Los Alamos Scientific Lab, Oct. **1973**.

[28] P. Solin, K. Segeth, and I. Dolezel. *Higher-Order Finite Element Methods*. Chapman and Hall/CRC, **2004**. ISBN 978-1584884385.

[29] P. K. Sweby. High resolution schemes using flux limiters for hyperbolic conservation laws. *SIAM Journal on Numerical Analysis*, 21(5):995–1011, **1984**. doi:10.1137/0721062.

[30] E. F. Toro. *Riemann Solvers and Numerical Methods for Fluid Dynamics*. Springer, **2009**. ISBN 978-3-540-49834-6. doi:10.1007/b79761.

[31] M. Ulbrich. *Semismooth Newton Methods for Variational Inequalities and Constrained Optimization Problems in Function Spaces*. SIAM, **2011**. ISBN 978-1611970692. doi:10.1137/1.9781611970692.

[32] B. van Leer. Towards the ultimate conservative difference scheme. v. a second-order sequel to Godunov's method. *Journal of Computational Physics*, 32(1):101–136, **1979**. ISSN 0021-9991. doi:10.1016/0021-9991(79)90145-1.

[33] V. Venkatakrishnan. Convergence to steady state solutions of the Euler equations on unstructured grids with limiters. *Journal of Computational Physics*, 118(1):120–130, **1995**. ISSN 0021-9991. doi:10.1006/jcph.1995.1084.

Acknowledgment

After seven years my study has come to an end. In this last year I worked full time on this research and learned a lot about numerical methods for PDE's, in particular the discontinuous Galerkin method and the use of limiters, but also additional techniques on how to convert the mathematics into code. It is fair to say that this was a great learning experience, and valuable addition to the classroom theory acquired from earlier years.

Half of the research was conducted at Brown University in the United States. Studying abroad was something on my checklist and my experience in and around Providence was very pleasant. During my stay at Brown I had the pleasure to have Chi-Wang Shu as an advisor. Despite being a highly cited author and popular speaker, he shows interest in his students and his open office policy is encouraging to show results and discuss them. The classes I attended were of a high quality and given by enthusiastic professors. They broadened and deepened my understanding of numerical methods for PDE's. Together with the seminars with researchers from different applications I enjoyed the glimpse into the wonderful world of applied mathematics.

In the Netherlands I finished the research under supervision of Jaap van der Vegt and Onno Bokhove. I enjoyed the weekly meetings with Jaap and am thankful for the helpful pointers and articles. It was good to see how the more theoretical mathematics from Brown can be applied in more application focused research in Twente.

Most work in this research was done individually with the help of my supervisors. However, I would like to thank Sirui Tan for easing the path in the introduction to DG and limiters and the pointers on how to find the source of the inevitable bugs. A last thanks to the Dutch government, which provided me with a large scholarship. Large enough to not only provide in my means of support, but which also supported my exploration of the land of the free. The oppurtunity to live in a foreign culture was an addition to my cultural baggage, which in my opinion is a necessary complement to the academic skills learned in college for every academic student.