Diploma Thesis

Institute for Numerical Simulation, TUHH

On the Comparison of the Finite Volume and Discontinuous Galerkin Methods

Corrected version

Katja Baumbach

August 17, 2006

Supervisor: Prof. Dr. M. Lukáčová-Medvid'ová

Technische Universität Hamburg-Harburg

Acknowledgements

I would like to thank Prof. Dr. M. Lukáčová-Medvid'ová for promoting my interest in computational fluid dynamics and for devoting a lot of time and effort into my work. For the contribution of numerical results obtained with the FVEG scheme my thanks go to Marcus Kraft.
I am sincerly grateful to my family and to Steffen Buller for their invaluable support, help and encouragement throughout the whole of my studies.

Contents

Introduction

Hyperbolic conservation laws are of vast importance in applied mathematics and engineering science. Among the many physical and technical applications which can be mathematically modeled by them, phenomena in fluid dynamics, elastodynamics, biomechanics, astrophysics and traffic systems should be mentioned.
In the general case it will not be possible to find an analytical solution of the hyperbolic differential equation system, so that it is more important to look for numerical solutions. The exact solution of hyperbolic equations is often characterized by discontinuities and shocks. With the popular class of finite volume methods, discontinuous approximate solutions of the hyperbolic system can be computed. For multidimensional hyperbolic problems, M. Lukáčová-Medvid'ová, K. W. Morton, and G. Warnecke obtained good numerical solutions with the genuinely multidimensional finite volume evolution Galerkin method, cf. [7].
Another important class of schemes for the modeling of partial differential equations is given by the finite element methods. In the usual case, these methods produce numerical solutions that are piecewise polynomial and continuous. As a result shocks are often smeared out. A possibility of combining the advantages of the finite element methods with those of the finite volume schemes is given by the discontinuous Galerkin method (DG method). This finite element method allows the construction of discontinuous numerical solutions. Being a finite element method, the DG scheme can easily accommodate complex geometries. The domain is subdivided into a grid of a finite number of elements. Grids of arbitrarily shaped elements can be constructed and are easily refined, where necessary. A piecewise polynomial numerical solution which can be discontinuous on cell interfaces is constructed. Interface fluxes are computed with approaches from the finite volume schemes. This bears the advantage that the numerical solution will reflect the conservation property which is characteristic for conservation laws. Finite volume solutions always reflect this property.
In this work we will apply the DG scheme to two hyperbolic problems, namely the Burgers equation and the shallow water equations, and compare the numerical solution with that obtained with a given implementation of the FVEG scheme and with solutions obtained with standard FV schemes.

Chapter 1

Discontinuous Galerkin Method

1.1 Standard Methods for Hyperbolic Conservation Laws

Hyperbolic Conservation Laws

A conservation law in two space dimensions has the following form

$$\frac{\partial}{\partial t}\boldsymbol{w}\left(\boldsymbol{x},t\right)+\frac{\partial}{\partial x_1}\boldsymbol{f}_1\left(\boldsymbol{w}\left(\boldsymbol{x},t\right)\right)+\frac{\partial}{\partial x_2}\boldsymbol{f}_2\left(\boldsymbol{w}\left(\boldsymbol{x},t\right)\right)=0, \quad \text{in } \Omega\times(0,T), \tag{1.1}$$

where $\boldsymbol{w}\left(\boldsymbol{x},t\right)\in\mathbb{R}^m$, $\boldsymbol{x}=(x_1,x_2)$. By m we denote the dimension of the system of equations. The functions $\boldsymbol{f}_1,\boldsymbol{f}_2\in C^1\left(\mathbb{R}^m;\mathbb{R}^m\right)$ are the fluxes of the conserved quantity $\boldsymbol{w}$ in x_1 and x_2 direction, respectively. If their Jacobians $\mathbb{A}_s\in C\left(\mathbb{R}^m;\mathbb{R}^{m\times m}\right)$, $s=1,2$,

$$\begin{aligned}\mathbb{A}_1&=\frac{d\boldsymbol{f_1}}{d\boldsymbol{w}},\\ \mathbb{A}_2&=\frac{d\boldsymbol{f_2}}{d\boldsymbol{w}},\end{aligned} \tag{1.2}$$

and all their linear combinations are diagonalizable with real eigenvalues, (1.1) is called *hyperbolic.*

The Finite Volume Method

When integrating (1.1) over a domain Ω and applying Green's theorem

$$\int_\Omega \operatorname{div}\,\boldsymbol{w}\,dx=\int_{\partial\Omega}\boldsymbol{w}\cdot\boldsymbol{n}\,dS, \tag{1.3}$$

we can derive an integral formulation of the conservation law:

$$\frac{d}{dt}\int_\Omega \boldsymbol{w}\left(\boldsymbol{x},t\right)\,d\boldsymbol{x}=\int_{\partial\Omega}\boldsymbol{f}_1 n_1+\boldsymbol{f}_2 n_2\,dS, \tag{1.4}$$

where $\boldsymbol{n} = (n_1, n_2)^T$ denotes the outer normal to the boundary $\partial\Omega$. This integral formulation of (1.1) expresses that changes to the integral of $\boldsymbol{w}$ over Ω occur only due to a flux through the boundary $\partial\Omega$. We say that $\boldsymbol{w}$ is conserved in the spatial domain Ω. This is the characteristic *conservation property* of conservation laws. In order to find a discrete formulation of (1.1) we integrate (1.1) over a time interval $[t^n, t^{n+1}]$ and get

$$\int_\Omega \boldsymbol{w}^{\boldsymbol{n+1}} \, d\boldsymbol{x} - \int_\Omega \boldsymbol{w}^{\boldsymbol{n}} \, d\boldsymbol{x} = -\int_{t^n}^{t^{n+1}} \int_{\partial\Omega} \boldsymbol{f}_1 n_1 + \boldsymbol{f}_2 n_2 \, dS. \tag{1.5}$$

Now the domain Ω is subdivided into a finite number of cells. We consider a mesh discretization $\mathcal{T}_h$ of a polygonal approximation Ω_h of the computational domain Ω. The elements K_i of the mesh discretization are numbered with indices $i \in I$, where I is a suitable index set. We thus have $\mathcal{T}_h = \{K_i\}_{i\in I}$. By Γ_{ij} we denote the interface between two neighbouring elements K_i, K_j and by $S(i) \subseteq I$ an index set with $j \in S(i)$ if K_i and K_j are neighbours. We reformulate (1.5) for each cell K_i and get

$$\begin{aligned} \frac{1}{|K_i|}\int_{K_i} \boldsymbol{w}^{\boldsymbol{n+1}} \, d\boldsymbol{x} - \frac{1}{|K_i|}\int_{K_i} \boldsymbol{w}^{\boldsymbol{n}} \, d\boldsymbol{x} &= \\ -\int_{t^n}^{t^{n+1}} \frac{1}{|K_i|} \sum_{j\in S(i)} \int_{\Gamma_{ij}} \boldsymbol{f}_1 n_1 + \boldsymbol{f}_2 n_2 \, dS. \end{aligned} \tag{1.6}$$

We approximate time integrals in (1.6) by the rectangle rule at old time levels. Due to possibly discontinuous solutions $\boldsymbol{w}$ it will be necessary to approximate flux integrals by a *numerical flux* $\boldsymbol{H}\left(\boldsymbol{w}_i^n, \boldsymbol{w}_j^n, \boldsymbol{n}_{ij}\right)$:

$$\boldsymbol{H}\left(\boldsymbol{w}_i^n, \boldsymbol{w}_j^n, \boldsymbol{n}_{ij}\right) \approx \boldsymbol{f}_1(\boldsymbol{w}^n) n_1 + \boldsymbol{f}_2(\boldsymbol{w}^n) n_2 \, dS. \tag{1.7}$$

Assuming a piecewise constant numerical solution, $\boldsymbol{H}\left(\boldsymbol{w}_i^n, \boldsymbol{w}_j^n, \boldsymbol{n}_{ij}\right)$ will be constant on cell interfaces and we have

$$\boldsymbol{H}\left(\boldsymbol{w}_i^n, \boldsymbol{w}_j^n, \boldsymbol{n}_{ij}\right) \approx \frac{1}{|\Gamma_{ij}|}\int_{\Gamma_{ij}} \boldsymbol{f}_1(\boldsymbol{w}^n) n_1 + \boldsymbol{f}_2(\boldsymbol{w}^n) n_2 \, dS. \tag{1.8}$$

The equation (1.6) then yields at each time level t^{n+1} an approximation $\boldsymbol{W}_i^{n+1}$ of the piecewise constant cell averaged values $\frac{1}{|K_i|}\int_{K_i} \boldsymbol{w}^{n+1} d\boldsymbol{x}$, which will be the numerical solution of (1.1)

$$\boldsymbol{W}_i^{n+1} \approx \frac{1}{|K_i|}\int_{K_i} \boldsymbol{w}^{\boldsymbol{n+1}} \, d\boldsymbol{x}. \tag{1.9}$$

Applying (1.6), (1.8), (1.9), we obtain the finite volume formulation of a general hyperbolic conservation law

$$\boldsymbol{W}_i^{n+1} = \boldsymbol{W}_i^n - \frac{\Delta t}{|K_i|} \sum_{j\in S(i)} \boldsymbol{H}\left(\boldsymbol{w}_i^n, \boldsymbol{w}_j^n, \boldsymbol{n}_{ij}\right) |\Gamma_{ij}|. \tag{1.10}$$

Consider now a rectangular grid with cells, which are aligned to the axes of the cartesian coordinate system, and denote by Δx_1 and Δx_2 the lengths of the cell interfaces. Assume that Γ_{ij} denotes either the right or left interface of a cell K_i, and Γ_{ik} the upper or lower cell interface, i.e. $j, k \in S(i)$. In horizontal direction we can then set

$$\boldsymbol{F}_{1,\Gamma_{ij}}(\boldsymbol{w}^n) \approx \frac{1}{\Delta x_2} \int_{\Gamma_{ij}} \boldsymbol{f}_1 \, dS. \tag{1.11}$$

and in vertical direction we set

$$\boldsymbol{F}_{2,\Gamma_{ik}}(\boldsymbol{w}^n) \approx \frac{1}{\Delta x_1} \int_{\Gamma_{ik}} \boldsymbol{f}_2 \, dS. \tag{1.12}$$

Then we can formulate the finite volume discretization of the conservation law (1.4) in the following way

$$\boldsymbol{W}_i^{n+1} = \boldsymbol{W}_i^n - \frac{\Delta t}{\Delta x_1}\left[\boldsymbol{F}^n_{1,\Gamma_{i,Right}} - \boldsymbol{F}^n_{1,\Gamma_{i,Left}}\right] - \frac{\Delta t}{\Delta x_2}\left[\boldsymbol{F}^n_{2,\Gamma_{i,Up}} - \boldsymbol{F}^n_{2,\Gamma_{i,Down}}\right]. \tag{1.13}$$

In literature, a wide variety of possible approximations of numerical flux functions can be found, cf. e.g. [3].

Approximate Riemann solvers

In this work we have implemented several approaches for the approximation of the integral fluxes, which belong all to the class of *approximate Riemann solvers*, also called *methods of Godunov type*. Their derivation follows from the analysis of the following one-dimensional linear *Riemann problem*:

$$\frac{\partial \boldsymbol{w}}{\partial t} + \mathbb{A}\frac{\partial \boldsymbol{w}}{\partial x} = 0, \quad (x,t) \in \mathbb{R} \times (0,\infty), \tag{1.14}$$

with the following initial condition

$$\boldsymbol{w}(\boldsymbol{x},0) = \boldsymbol{w}^0 = \begin{cases} \boldsymbol{u}, & x < 0, \\ \boldsymbol{v}, & x > 0. \end{cases} \tag{1.15}$$

We again assume that $\mathbb{A}$ is diagonalizable with real eigenvalues λ_i. The exact solution $\boldsymbol{w}$ of (1.14), (1.15) is given by

$$\boldsymbol{w}(x,t) = \sum_{i=1}^{m} \left[\beta_i H(x - \lambda_i t) + \alpha_i (1 - H(x - \lambda_i t))\right] \boldsymbol{r}_i, \tag{1.16}$$

where $\boldsymbol{r}_i$ are the right eigenvectors of $\mathbb{A}$, the scalars α_i, β_i are the coefficients in the following linear combinations

$$\boldsymbol{u} = \sum_{i=1}^{m} \alpha_i \boldsymbol{r}_i, \quad \boldsymbol{v} = \sum_{i=1}^{m} \beta_i \boldsymbol{r}_i, \tag{1.17}$$

and $H(x)$ is the Heavyside function defined as follows

$$H(x) = \begin{cases} 1, & x > 0, \\ 0, & x < 0. \end{cases} \tag{1.18}$$

If we compute a finite volume solution of a linear one-dimensional hyperbolic problem

$$\begin{aligned} &\frac{\partial \boldsymbol{w}}{\partial t} + \mathbb{A}\frac{\partial \boldsymbol{w}}{\partial x} = 0, \quad (x,t) \in \mathbb{R} \times (0,\infty), \\ &\boldsymbol{w}(x,t) = \boldsymbol{w}^0, \quad x \in \mathbb{R}, \end{aligned} \tag{1.19}$$

with arbitrary initial data $\boldsymbol{w}^0$, according to (1.10), the computation of the approximate solution on an interface Γ_{ij} from the data $\boldsymbol{w}|_{K_i}$ and $\boldsymbol{w}|_{K_j}$ in two adjacent cells K_i, K_j can be understood as just such a linear one-dimensional Riemann problem (1.14), (1.15) with

$$\begin{aligned} \boldsymbol{u} &= \boldsymbol{w}^n|_{K_i}, \\ \boldsymbol{v} &= \boldsymbol{w}^n|_{K_j}. \end{aligned} \tag{1.20}$$

The numerical fluxes in (1.10) can then be computed by inserting the exact solution of the arising local linear Riemann problem into the numerical flux functions $\boldsymbol{f}$ of the hyperbolic system.
We then get the following numerical fluxes

$$\boldsymbol{F}^n_{\Gamma_{ij}} = \mathbb{A}^+\boldsymbol{w}^n|_{K_i} + \mathbb{A}^-\boldsymbol{w}^n|_{K_j}. \tag{1.21}$$

The matrices $\mathbb{A}^+, \mathbb{A}^-$ are defined as follows

$$\begin{aligned} &\mathbb{A}^\pm = \mathbb{R}\Lambda^\pm\mathbb{L}, \\ &\Lambda^\pm = \operatorname{diag}(\lambda_1^\pm, \dots, \lambda_m^\pm), \\ &\lambda_i^+ = \max(\lambda_i, 0),\ \lambda_i^- = \min(\lambda_i, 0),\ i = 1, \dots, m. \end{aligned} \tag{1.22}$$

Here $\mathbb{R}$, $\mathbb{L}$ denote the matrices consisting of right and left eigenvectors of $\mathbb{A} = \frac{d\boldsymbol{f}}{d\boldsymbol{w}}$, respectively, and $\lambda_i,\ i = 1, \dots, m$ their eigenvalues. A derivation of (1.21) as well as further details on Riemann problems and finite volume methods for hyperbolic problems can be found in [2]. The concept of computing the numerical fluxes in (1.13) according to (1.20), (1.21) is called *exact Riemann solver.*
For a nonlinear hyperbolic problem of the form (1.1) we cannot compute in general the exact solution of the local Riemann problem, efficiently. The so called *approximate Riemann solver* proposes to set

$$\begin{aligned} \boldsymbol{F}^n_{s,\Gamma_{ij}} &= \boldsymbol{g}^R_s(\boldsymbol{u}, \boldsymbol{v}), \\ \boldsymbol{g}^R_s(\boldsymbol{u}, \boldsymbol{v}) &= \boldsymbol{f}^+_s(\boldsymbol{u}) + \boldsymbol{f}^-_s(\boldsymbol{v}), \end{aligned} \tag{1.23}$$

where

$$\begin{aligned} &\boldsymbol{f}_s(\boldsymbol{w}) = \boldsymbol{f}^+_s(\boldsymbol{w}) + \boldsymbol{f}^-_s(\boldsymbol{w}), \\ &\frac{d\boldsymbol{f}^\pm_s(\boldsymbol{w})}{d\boldsymbol{w}} = \mathbb{A}^\pm_s(\boldsymbol{w}), \quad s = 1, 2. \end{aligned} \tag{1.24}$$

For linear problems we have

$$\boldsymbol{f}_s(\boldsymbol{w}) = \mathbb{A}_s \boldsymbol{w}, \tag{1.25}$$

and can thus set

$$\begin{aligned} \boldsymbol{f}_s^+(\boldsymbol{w}) &= \mathbb{A}_s^+ \boldsymbol{w}, \\ \boldsymbol{f}_s^-(\boldsymbol{w}) &= \mathbb{A}_s^- \boldsymbol{w}. \end{aligned} \tag{1.26}$$

Therefore, the idea (1.20), (1.23) is consistent with the approach (1.20), (1.21) of the exact Riemann solver.
If the hyperbolic problem fulfills the following *homogeneity condition*

$$\boldsymbol{f}_s(\boldsymbol{w}) = \mathbb{A}_s(\boldsymbol{w})\boldsymbol{w}, \tag{1.27}$$

the vectors $\boldsymbol{f}_s^+, \boldsymbol{f}_s^-$ in (1.23) can be constructed in the following way

$$\begin{aligned} \boldsymbol{f}_s^\pm(\boldsymbol{w}) &= \mathbb{A}_s^\pm(\boldsymbol{w})\boldsymbol{w}, \\ \boldsymbol{f}_s(\boldsymbol{w}) &= \mathbb{A}_s^+(\boldsymbol{w})\boldsymbol{w} + \mathbb{A}_s^-(\boldsymbol{w})\boldsymbol{w}. \end{aligned} \tag{1.28}$$

On the basis of (1.28) we can derive the **Steger-Warming scheme**, reading

$$\boldsymbol{g}_s^{SW}(u,v) = \mathbb{A}_s^+(\boldsymbol{u})\,\boldsymbol{u} + \mathbb{A}_s^-(\boldsymbol{v})\,\boldsymbol{v}. \tag{1.29}$$

Another approach, also based on the homogeneity condition, is given by the **Vijayasundaram scheme**

$$\boldsymbol{g}_s^V(u,v) = \mathbb{A}_s^+\left(\frac{\boldsymbol{u}+\boldsymbol{v}}{2}\right)\boldsymbol{u} + \mathbb{A}_s^-\left(\frac{\boldsymbol{u}+\boldsymbol{v}}{2}\right)\boldsymbol{v}. \tag{1.30}$$

Unfortunately, not all hyperbolic problems fulfill (1.27). In this work we have set our focus on the Burgers equation (4.1) and on the shallow water equations (5.1), (5.2), presented in the Chapters 4 and 5, for both of which we have

$$\boldsymbol{f}(\boldsymbol{w}) \neq \mathbb{A}(\boldsymbol{w})\boldsymbol{w}. \tag{1.31}$$

For these equations the *Van Leer scheme*, which does not require (1.27) is an appropriate choice

$$\boldsymbol{g}_s^{VL}(u,v) = \frac{1}{2}\left\{\boldsymbol{f}_s(\boldsymbol{u}) + \boldsymbol{f}_s(\boldsymbol{v}) - \left|\mathbb{A}_s\left(\frac{\boldsymbol{u}+\boldsymbol{v}}{2}\right)\right|\right\}, \tag{1.32}$$

with

$$\begin{aligned} |\mathbb{A}_s| &= \mathbb{R}_s|\Lambda_s|\mathbb{L}_s, \\ |\Lambda_s| &= \operatorname{diag}(|\lambda_{s,1}|, \ldots, |\lambda_{s,m}|). \end{aligned} \tag{1.33}$$

The Finite Volume Evolution Galerkin Method

The FVEG method is a genuinely multidimensional finite volume scheme. Instead of approximating the average fluxes through the cell interfaces by a one-dimensional approach, as is the case for the above presented approximate Riemann solvers, the fluxes are computed with a truly multidimensional approach, the so-called *evolution operator* E_Δ. The evolution operator maps the approximate solution at time t^n to its value at an intermediate time level $t^{n+\frac{1}{2}}$. It is an approximation of the exact evolution operator of the hyperbolic system, which describes the time evolution of the solution along so-called *bicharacteristics*. With this approach, the numerical solution can be computed at the interfaces of each cell K_i and inserted into the flux functions $\boldsymbol{f}_1, \boldsymbol{f}_2$. For a regular rectangular grid the numerical fluxes can be formulated as follows

$$\begin{aligned} \boldsymbol{F}^n_{1,\Gamma_{ij}} &= \frac{1}{\Delta x_2} \int_{\Gamma_{ij}} \boldsymbol{f}_1 \left(E^{\Gamma_{ij}}_{\Delta t/2} \boldsymbol{U}^n \right) \, dS, \\ \boldsymbol{F}^n_{2,\Gamma_{ik}} &= \frac{1}{\Delta x_1} \int_{\Gamma_{ik}} \boldsymbol{f}_2 \left(E^{\Gamma_{ik}}_{\Delta t/2} \boldsymbol{U}^n \right) \, dS. \end{aligned} \tag{1.34}$$

The solution at the new time level can then be computed according to (1.13). The theoretical background as well as an extensive numerical analysis of the FVEG scheme can be found in [7], [8].

The Finite Element Method

Again we consider problem (1.1)

$$\frac{\partial}{\partial t} \boldsymbol{w}(\boldsymbol{x}, t) + \frac{\partial}{\partial x_1} \boldsymbol{f}_1 (\boldsymbol{w}(\boldsymbol{x}, t)) + \frac{\partial}{\partial x_2} \boldsymbol{f}_2 (\boldsymbol{w}(\boldsymbol{x}, t)) = 0, \quad \text{in } \Omega \times (0, T), \tag{1.35}$$

with a *Dirichlet condition* which prescribes $\boldsymbol{w}$ at the boundary $\partial\Omega$ of the domain Ω

$$\boldsymbol{w}|_{\partial\Omega} = \boldsymbol{w}_D, \tag{1.36}$$

with a given function $\boldsymbol{w}_D$. A function $\boldsymbol{w} \in C^1(\overline{\Omega} \times (0, T))$ that satisfies (1.35), (1.36) is called a *classical solution* of (1.35), (1.36).
We will now formulate an integral formulation of (1.35) which allows solutions which are not necessarily in $C^1(\overline{\Omega} \times (0, T))$. For the problems described by (1.35) this yields the so-called *variational solution.*
We introduce the Sobolev space $\boldsymbol{H}^1(\Omega) = [H^1(\Omega)]^m$,

$$H^1(\Omega) = \left\{ u \in L^2(\Omega);\; D_x u \in L^2(\Omega) \right\}, \tag{1.37}$$

where $D_x u$ is the so-called distributional derivative, defined by

$$\int_\Omega u \varphi_x \, dx = - \int_\Omega D_x u \varphi \, dx, \quad \forall \varphi \in C_0^\infty(\Omega). \tag{1.38}$$

For $u \in C^1(\Omega)$ we have $D_x u = u'_x$. Thus the concept of distributional derivatives extends the notion of derivatives for functions which are not differentiable.
There exists another definition of the Sobolev space $H^1(\Omega)$. We consider the following norm

$$||f||_1 = \left(\int_\Omega f^2 + \left(\frac{\partial f}{\partial x} \right)^2 + \left(\frac{\partial f}{\partial y} \right)^2 \right)^{\frac{1}{2}}. \tag{1.39}$$

The Sobolev space $H^1(\Omega)$ is the extension of the space $C^\infty(\Omega)$ with respect to this norm

$$H^1(\Omega) = \overline{(C^\infty(\Omega))}^{||\cdot||_1}. \tag{1.40}$$

It is possible to prove that (1.37) is equivalent to (1.40).
We further consider the space

$$H_0^1(\Omega) = \left\{ \varphi \in H^1(\Omega);\ \text{trace of } \varphi \text{ on } \partial\Omega = 0 \right\}. \tag{1.41}$$

Now we multiply (1.35) by an arbitrary function $\boldsymbol{\varphi} \in \boldsymbol{H}_0^1(\Omega) = [H_0^1(\Omega)]^m$, integrate over Ω and get

$$\frac{d}{dt} \int_\Omega \boldsymbol{w}(\boldsymbol{x}, t) \cdot \boldsymbol{\varphi} \, d\boldsymbol{x} = - \int_\Omega \left(\frac{\partial}{\partial x_1} \boldsymbol{f}_1 + \frac{\partial}{\partial x_2} \boldsymbol{f}_2 \right) \cdot \boldsymbol{\varphi} \, d\boldsymbol{x}, \quad \forall \boldsymbol{\varphi} \in \boldsymbol{H}_0^1(\Omega). \tag{1.42}$$

Application of the product rule

$$\frac{\partial}{\partial x_1} (\boldsymbol{f}_1 \cdot \boldsymbol{\varphi}) + \frac{\partial}{\partial x_2} (\boldsymbol{f}_2 \cdot \boldsymbol{\varphi}) = \frac{\partial \boldsymbol{f}_1}{\partial x_1} \cdot \boldsymbol{\varphi} + \frac{\partial \boldsymbol{f}_2}{\partial x_2} \cdot \boldsymbol{\varphi} + \frac{\partial \boldsymbol{\varphi}}{\partial x_1} \cdot \boldsymbol{f}_1 + \frac{\partial \boldsymbol{\varphi}}{\partial x_2} \cdot \boldsymbol{f}_2 \tag{1.43}$$

yields

$$\int_\Omega \left(\frac{\partial}{\partial x_1} \boldsymbol{f}_1 + \frac{\partial}{\partial x_2} \boldsymbol{f}_2 \right) \cdot \boldsymbol{\varphi} \, d\boldsymbol{x} = \int_\Omega \left(\frac{\partial}{\partial x_1} (\boldsymbol{f}_1 \cdot \boldsymbol{\varphi}) + \frac{\partial}{\partial y} (\boldsymbol{f}_2 \cdot \boldsymbol{\varphi}) - \frac{\partial \boldsymbol{\varphi}}{\partial x_1} \cdot \boldsymbol{f}_1 - \frac{\partial \boldsymbol{\varphi}}{\partial x_2} \cdot \boldsymbol{f}_2 \right) d\boldsymbol{x} \tag{1.44}$$

and thus

$$\begin{aligned} \frac{d}{dt} \int_\Omega \boldsymbol{w}(\boldsymbol{x}, t) \cdot \boldsymbol{\varphi} \, d\boldsymbol{x} = & - \int_\Omega \left(\frac{\partial}{\partial x_1} (\boldsymbol{f}_1 \cdot \boldsymbol{\varphi}) + \frac{\partial}{\partial x_2} (\boldsymbol{f}_2 \cdot \boldsymbol{\varphi}) \right) d\boldsymbol{x} + \\ & \int_\Omega \left(\frac{\partial \boldsymbol{\varphi}}{\partial x_1} \cdot \boldsymbol{f}_1 + \frac{\partial \boldsymbol{\varphi}}{\partial x_2} \cdot \boldsymbol{f}_2 \right) d\boldsymbol{x}, \quad \forall \boldsymbol{\varphi} \in \boldsymbol{H}_0^1(\Omega). \end{aligned} \tag{1.45}$$

Application of Green's theorem (1.3) yields a variational formulation of (1.35) which is the basis of the finite element method

$$\begin{aligned} \frac{d}{dt} \int_\Omega \boldsymbol{w}(\boldsymbol{x}, t) \cdot \boldsymbol{\varphi} \, d\boldsymbol{x} = & - \int_{\partial\Omega} (\boldsymbol{f}_1 \cdot \boldsymbol{\varphi} n_1 + \boldsymbol{f}_2 \cdot \boldsymbol{\varphi} n_2) \, dS + \\ & \int_\Omega \left(\frac{\partial \boldsymbol{\varphi}}{\partial x_1} \cdot \boldsymbol{f}_1 + \frac{\partial \boldsymbol{\varphi}}{\partial x_2} \cdot \boldsymbol{f}_2 \right) d\boldsymbol{x}, \quad \forall \boldsymbol{\varphi} \in \boldsymbol{H}_0^1(\Omega). \end{aligned} \tag{1.46}$$

As $\boldsymbol{\varphi} \in \boldsymbol{H}_0^1(\Omega)$, the surface integrals are equal to zero and we get

$$\frac{d}{dt}\int_{\Omega} \boldsymbol{w}\,(\boldsymbol{x},t)\cdot\boldsymbol{\varphi}\,d\boldsymbol{x} = +\int_{\Omega}\left(\frac{\partial\boldsymbol{\varphi}}{\partial x_1}\cdot\boldsymbol{f}_1+\frac{\partial\boldsymbol{\varphi}}{\partial x_2}\cdot\boldsymbol{f}_2\right)\,d\boldsymbol{x},\quad \forall\boldsymbol{\varphi}\in\boldsymbol{H}_0^1(\Omega). \tag{1.47}$$

We call (1.47) a *variational formulation* of (1.35) and any solution $\boldsymbol{w} \in \boldsymbol{L}^2((0,T);\boldsymbol{H}^1(\Omega))$ of (1.47) a *variational solution* of (1.35). It represents an extended formulation of problem (1.35), that allows also solutions which are not in $C^1(\overline{\Omega}\times(0,T))$. The variational formulation is consistent with (1.35) because any variational solution which is in $C^1(\overline{\Omega}\times(0,T))$ is also a classical solution of (1.35).
It should be pointed out that in order to derive the above variational formulation we need to assume a $\boldsymbol{H}^1(\Omega)$-regularity of the solution. For physically reasonable solutions it is however enough that the integral equation (1.5) is satisfied. Such solutions, the so-called *weak solutions*, belong to the class of $L^\infty(\Omega\times(0,T))$ for which the total variation is bounded, cf. [3].
A discretization of the domain Ω as well as of (1.47), leads to an algebraic system, from which an approximate solution of (1.47) can be computed.
To this aim we consider a polygonal approximation Ω_h of the domain Ω, and construct a grid by subdividing Ω_h into a finite number of cells K_i. In this work we will only be concerned with regular grids. On the approximate domain Ω_h we define the space $\mathbb{X}_h$ of piecewise polynomial functions

$$\begin{aligned}\mathbb{X}_h &= \left\{v_h \in L^2(\Omega_h);\ v_h|_{K_i}\in P^p(K_i)\right\},\\ \mathbb{X}_{h,0} &= \{\varphi\in\mathbb{X}_h(\Omega_h);\ \varphi|_{\partial\Omega_h}=0\},\end{aligned} \tag{1.48}$$

where $P^p(K_i)$ is the set of all polynomials of degree $\le p$ on the cell K_i. An approximate solution $\boldsymbol{w}_h$ of (1.47) lying in the space

$$\boldsymbol{X}_h = [\mathbb{X}_h]^m \tag{1.49}$$

can now be found by solving the following discretized problem

$$\frac{d}{dt}\int_{\Omega_h}\boldsymbol{w_h}\,(\boldsymbol{x},t)\cdot\boldsymbol{\varphi}_h\,d\boldsymbol{x} = \int_{\Omega_h}\left(\frac{\partial\boldsymbol{\varphi}_h}{\partial x_1}\cdot\boldsymbol{f}_1+\frac{\partial\boldsymbol{\varphi}_h}{\partial x_2}\cdot\boldsymbol{f}_2\right)\,d\boldsymbol{x},\quad\forall\boldsymbol{\varphi}_h\in\boldsymbol{X}_{h,0}. \tag{1.50}$$

In contrast to $\boldsymbol{H}_0^1(\Omega)$, the space $\boldsymbol{X}_{h,0}$ is spanned by a finite number of basis functions $\boldsymbol{\varphi}_h$. We will denote by B the set of basis functions. By claiming (1.50) for each cell K_i and for each basis function $\boldsymbol{\varphi}_h \in B$, we get the following system of equations

$$\frac{d}{dt}\int_{K_i}\boldsymbol{w_h}\,(\boldsymbol{x},t)\cdot\boldsymbol{\varphi}_h\,d\boldsymbol{x} = \int_{K_i}\left(\frac{\partial\boldsymbol{\varphi}_h}{\partial x_1}\cdot\boldsymbol{f}_1+\frac{\partial\boldsymbol{\varphi}_h}{\partial x_2}\cdot\boldsymbol{f}_2\right)\,d\boldsymbol{x},\quad\forall\boldsymbol{\varphi}_h\in B. \tag{1.51}$$

Note that we can express $\boldsymbol{w}_h$ in (1.51) as a linear combination of basis functions

$$\boldsymbol{w}_h = \sum_j c_j\boldsymbol{\varphi}_{h_j}. \tag{1.52}$$

If we approximate the integrals with a suitable quadrature rule, we finally get a linear system of equations, from which we can compute the coefficients c_j and thus obtain the approximate solution $\boldsymbol{w}_h$. For other boundary conditions the choice of space for the test functions will be slightly different, but the general proceeding is analogous. Depending on how the approximate spaces for the discrete solution $\boldsymbol{w}_h$ and test functions $\boldsymbol{\varphi}_h$ are chosen, different finite element methods can be derived.

1.2 The DG Method

The discontinuous Galerkin method is based on a modified variational formulation of (1.1). We again consider (1.46), now with $\boldsymbol{\varphi} \in H^1(\Omega)$

$$\begin{aligned} \frac{d}{dt} \int_\Omega \boldsymbol{w}(\boldsymbol{x},t) \cdot \boldsymbol{\varphi} \, d\boldsymbol{x} = &- \int_{\partial\Omega} \left(\boldsymbol{f}_1 \cdot \boldsymbol{\varphi} n_1 + \boldsymbol{f}_2 \cdot \boldsymbol{\varphi} n_2\right) \, dS + \\ & \int_\Omega \left(\frac{\partial \boldsymbol{\varphi}}{\partial x_1} \cdot \boldsymbol{f}_1 + \frac{\partial \boldsymbol{\varphi}}{\partial x_2} \cdot \boldsymbol{f}_2 \right) \, d\boldsymbol{x}, \quad \forall \boldsymbol{\varphi} \in H^1(\Omega). \end{aligned} \tag{1.53}$$

Before we focus on the surface integrals, we will formulate (1.53) for each cell K_i of Ω_h

$$\begin{aligned} \frac{d}{dt} \sum_{i\in I} \int_{K_i} \boldsymbol{w}(\boldsymbol{x},t) \cdot \boldsymbol{\varphi} \, d\boldsymbol{x} = &- \sum_{i\in I} \sum_{j \in S(i)} \int_{\Gamma_{ij}} \sum_{s=1,2} \boldsymbol{f}_s \cdot \boldsymbol{\varphi} n_s \, dS + \\ & \sum_{i \in I} \int_{K_i} \left(\frac{\partial \boldsymbol{\varphi}}{\partial x_1} \cdot \boldsymbol{f}_1 + \frac{\partial \boldsymbol{\varphi}}{\partial x_2} \cdot \boldsymbol{f}_2 \right) \, d\boldsymbol{x}, \quad \forall \boldsymbol{\varphi} \in H^1(\Omega). \end{aligned} \tag{1.54}$$

In terms of the $[L^2]^m$-scalar product

$$(\boldsymbol{w}, \boldsymbol{\varphi}) = \sum_{i\in I} \int_{K_i} \boldsymbol{w}(\boldsymbol{x},t) \cdot \boldsymbol{\varphi}(\boldsymbol{x}) \, d\boldsymbol{x} \tag{1.55}$$

(1.54) leads to

$$\begin{aligned} \frac{d}{dt} (\boldsymbol{w}, \boldsymbol{\varphi}) = &- \sum_{i\in I} \sum_{j \in S(i)} \int_{\Gamma_{ij}} \sum_{s=1,2} \boldsymbol{f}_s \cdot \boldsymbol{\varphi} n_s \, dS + \\ & \sum_{i \in I} \int_{K_i} \sum_{s=1,2} \frac{\partial \boldsymbol{\varphi}}{\partial x_s} \cdot \boldsymbol{f}_s \, d\boldsymbol{x}, \quad \forall \boldsymbol{\varphi} \in H^1(\Omega). \end{aligned} \tag{1.56}$$

We choose $\boldsymbol{\varphi}_h \in \boldsymbol{X}_h$ and claim that an approximate weak solution $\boldsymbol{w}_h \in C^1([0,T]; \boldsymbol{X}_h)$ has to satisfy a discrete version of (1.56) for all $\boldsymbol{\varphi}_h \in \boldsymbol{X}_h$.
However, as described before, our objective is to construct a numerical solution which is possibly discontinuous on cell interfaces, where we have to evaluate the flux functions if we want to compute the surface integrals in (1.56). We cannot formulate these surface integrals without defining the fluxes of $\boldsymbol{w}_h$ on the cell interfaces.

The DG method proposes to use the concept of numerical fluxes $H\left(\boldsymbol{w}_h|_{\Gamma_{ij}}, \boldsymbol{w}_h|_{\Gamma_{ji}}, \boldsymbol{n}_{ij}\right)$, cf. (1.7), derived from the finite volume methods

$$\int_{\Gamma_{ij}} \sum_{s=1,2} \boldsymbol{f}_s \cdot \boldsymbol{\varphi}_h n_s \, dS \approx \int_{\Gamma_{ij}} H\left(\boldsymbol{w}_h|_{\Gamma_{ij}}, \boldsymbol{w}_h|_{\Gamma_{ji}}, \boldsymbol{n}_{ij}\right) \cdot \boldsymbol{\varphi}_h \, dS. \tag{1.57}$$

The numerical flux can be computed with approximate Riemann solvers from the values of the discontinuous conserved variable in the cells adjacent to the considered cell interface. We now require that for an arbitrary $\boldsymbol{\varphi}_h \in \boldsymbol{X}_h$ the approximate solution $\boldsymbol{w}_h \in C^1\left(\boldsymbol{X}_h; [0,T]\right)$ fulfills the following discretization of (1.56)

$$\begin{aligned} \frac{d}{dt}\left(\boldsymbol{w}_h, \boldsymbol{\varphi}_h\right) = &- \sum_{i\in I} \sum_{j\in S(i)} \int_{\Gamma_{ij}} H\left(\boldsymbol{w}_h|_{\Gamma_{ij}}, \boldsymbol{w}_h|_{\Gamma_{ji}}, \boldsymbol{n}_{ij}\right) \cdot \boldsymbol{\varphi}_h \ dS + \\ &\sum_{i\in I} \int_{K_i} \sum_{s=1,2} \frac{\partial \boldsymbol{\varphi}_h}{\partial x_s} \cdot \boldsymbol{f}_s \, d\boldsymbol{x}, \quad \forall \boldsymbol{\varphi}_h \in \boldsymbol{X}_h. \end{aligned} \tag{1.58}$$

Denoting by B the set of basis functions of $\boldsymbol{X}_h$ – the number of basis functions of the approximate space is again finite – it is sufficient to satisfy (1.58) for all $\boldsymbol{\varphi}_h \in B$

$$\begin{aligned} \frac{d}{dt}\left(\boldsymbol{w}_h, \boldsymbol{\varphi}_h\right) = &- \sum_{i\in I} \sum_{j\in S(i)} \int_{\Gamma_{ij}} H\left(\boldsymbol{w}_h|_{\Gamma_{ij}}, \boldsymbol{w}_h|_{\Gamma_{ji}}, \boldsymbol{n}_{ij}\right) \cdot \boldsymbol{\varphi}_h \, dS + \\ &\sum_{i} \int_{K_i} \sum_{s=1,2} \frac{\partial \boldsymbol{\varphi}_h}{\partial x_s} \cdot \boldsymbol{f}_s \, d\boldsymbol{x}, \quad \forall \boldsymbol{\varphi}_h \in B. \end{aligned} \tag{1.59}$$

Using the notation

$$\begin{aligned} b_h(\boldsymbol{w}_h, \boldsymbol{\varphi}_h) = &\sum_{i\in I} \sum_{j\in S(i)} \int_{\Gamma_{ij}} H\left(\boldsymbol{w}_h|_{\Gamma_{ij}}, \boldsymbol{w}_h|_{\Gamma_{ji}}, \boldsymbol{n}_{ij}\right) \cdot \boldsymbol{\varphi}_h \, dS - \\ &\sum_{i\in I} \int_{K_i} \sum_{s=1,2} \frac{\partial \boldsymbol{\varphi}_h}{\partial x_s} \cdot \boldsymbol{f}_s \, d\boldsymbol{x}, \quad \boldsymbol{w}_h \in C^1\left(\boldsymbol{X}_h; [0,T]\right), \quad \boldsymbol{\varphi}_h \in \boldsymbol{X}_h, \end{aligned} \tag{1.60}$$

the DG discretization of problem (1.1) results in the following initial value problem

$$\begin{aligned} &\boldsymbol{w}_h \in C^1\left([0,T]\,; \boldsymbol{X}_h\right), \\ &\frac{d}{dt}\left(\boldsymbol{w}_h(t), \boldsymbol{\varphi}_h\right) + b_h(\boldsymbol{w}_h(t), \boldsymbol{\varphi}_h) = 0, \quad \forall \boldsymbol{\varphi}_h \in B, \\ &\boldsymbol{w}_h(0) = \boldsymbol{w}_h^0. \end{aligned} \tag{1.61}$$

The basis functions are constructed according to the choice of the grid and of the space $\boldsymbol{X}_h$.
A suitable choice of methods for the discretization in time and space, i.e. of suitable time stepping methods and quadrature rules, completes the discretization and yields an algorithm to update the unknown values of the numerical solution in each cell for each time interval.
Implementation examples will be presented and discussed in the next chapter.

Chapter 2

Implementation of DG schemes

2.1 Implementation with Rectangles

As a first implementation example we will consider piecewise bilinear test functions on a rectangular grid, cf. Figure 2.1. The elements K_i of the grid are numbered with indices $i \in I$, where I is a suitable index set. By $A_{i\nu}$, $\nu \in V(i) = \{0, \dots 3\}$ we denote the four vertices of a cell K_i. The test functions are associated with the vertices $A_{i\nu}$ of the cells. The set B of basis functions then has the following form

$$\begin{aligned}
&B = \left\{ \boldsymbol{\phi}_{k\mu}^1, \dots, \boldsymbol{\phi}_{k\mu}^m \right\}, \quad k \in I,\ \mu \in V(k), \\
&\boldsymbol{\phi}_{k\mu}^1 = \left(\phi_{k\mu}, 0, \dots, 0\right)^T,\ \dots,\ \boldsymbol{\phi}_{k\mu}^m = \left(0, \dots, 0, \phi_{k\mu}\right)^T, \\
&\phi_{k\mu}(A_{i\nu}) = \delta_{ik}\delta_{\nu\mu},
\end{aligned} \tag{2.1}$$

where I denotes the index set of mesh cells. The number of basis functions is equal to m times the number of vertices, i.e. in our case m times the number of cells times four, where m is again the dimension of the considered equation system.
In each cell, $\boldsymbol{w}_h$ can be expressed as a linear combination of the basis functions associated with the vertices of this cell. Evaluation of the q-th component of $\boldsymbol{w}_h$ in each vertex $A_{k\mu}$ gives the coefficients $w_{k\mu}^q$ in the linear combination

$$\boldsymbol{w}_h|_{K_k} = \sum_{\substack{q=1 \\ \mu \in V(k)}}^{m} w_{k\mu}^q \boldsymbol{\phi}_{k\mu}^q. \tag{2.2}$$

When expressing $\boldsymbol{w}_h$ in the scalar product in (1.61) according to (2.2) we get

$$\begin{aligned}
(\boldsymbol{w}_h, \boldsymbol{\varphi}_h) &= \sum_{k \in I} \int_{K_k} \sum_{\substack{q=1 \\ \mu \in V(k)}}^{m} w_{k\mu}^q(t) \boldsymbol{\phi}_{k\mu}^q \cdot \boldsymbol{\varphi}_h \, d\boldsymbol{x} \\
&= \sum_{k \in I} \sum_{\substack{r=1 \\ \mu \in V(k)}}^{m} w_{k\mu}^q(t) \int_{K_k} \boldsymbol{\phi}_{k\mu}^q \cdot \boldsymbol{\varphi}_h \, d\boldsymbol{x}.
\end{aligned} \tag{2.3}$$

We have

$$\boldsymbol{\varphi}_h \in B = \left\{\boldsymbol{\phi}_{i\nu}^1, \dots, \boldsymbol{\phi}_{i\nu}^m\right\}, \quad i \in I,\ \nu \in V(i). \tag{2.4}$$

Equation (1.61) yields

$$\frac{d}{dt}\left(\boldsymbol{w}_h, \boldsymbol{\phi}_{i\nu}^r\right) + b_h(\boldsymbol{w}_h(t), \boldsymbol{\phi}_{i\nu}^r) = 0, \quad i \in I, \nu \in V(i), r = 1, \dots, m. \tag{2.5}$$

We will now consider the scalar product on the left hand side of (2.5) more closely. We state that

$$(\boldsymbol{w}_h, \boldsymbol{\phi}_{i\nu}^r) = \sum_{k\in I} \sum_{\substack{q=1 \\ \mu\in V(k)}}^{m} w_{k\mu}^q(t) \int_{K_k} \boldsymbol{\phi}_{k\mu}^q \cdot \boldsymbol{\phi}_{i\nu}^r \, d\boldsymbol{x}, \quad i \in I,\ \nu \in V(i), r = 1, \dots, m. \tag{2.6}$$

According to (2.1) it holds that

$$\left(\boldsymbol{\phi}_{k\mu}^q, \boldsymbol{\phi}_{i\nu}^r\right) = 0, \qquad \text{for } q \neq r \text{ or } k \neq i. \tag{2.7}$$

That is why we get

$$(\boldsymbol{w}_h, \boldsymbol{\phi}_{i\nu}^r) = \sum_{\mu\in V(i)} w_{i\mu}^r(t) \int_{K_i} \boldsymbol{\phi}_{i\mu}^r \cdot \boldsymbol{\phi}_{i\nu}^r \, d\boldsymbol{x}, \quad \nu \in V(i), \tag{2.8}$$

and thus

$$\begin{pmatrix} (\boldsymbol{w}_h, \boldsymbol{\phi}_{i0}^r) \\ (\boldsymbol{w}_h, \boldsymbol{\phi}_{i1}^r) \\ (\boldsymbol{w}_h, \boldsymbol{\phi}_{i2}^r) \\ (\boldsymbol{w}_h, \boldsymbol{\phi}_{i3}^r) \end{pmatrix} = \mathbb{M} \begin{pmatrix} w_{i0}^r(t) \\ w_{i1}^r(t) \\ w_{i2}^r(t) \\ w_{i3}^r(t) \end{pmatrix}, \quad i \in I, \tag{2.9}$$

where $\mathbb{M}$ is the so-called *mass matrix* for the cell K_i

$$\mathbb{M} = \begin{pmatrix} \int_{K_i} \boldsymbol{\phi}_{i0}^r \boldsymbol{\phi}_{i0}^r \, d\boldsymbol{x} & \dots & \int_{K_i} \boldsymbol{\phi}_{i3}^r \boldsymbol{\phi}_{i0}^r \, d\boldsymbol{x} \\ \int_{K_i} \boldsymbol{\phi}_{i0}^r \boldsymbol{\phi}_{i1}^r \, d\boldsymbol{x} & \dots & \int_{K_i} \boldsymbol{\phi}_{i3}^r \boldsymbol{\phi}_{i1}^r \, d\boldsymbol{x} \\ \int_{K_i} \boldsymbol{\phi}_{i0}^r \boldsymbol{\phi}_{i2}^r \, d\boldsymbol{x} & \dots & \int_{K_i} \boldsymbol{\phi}_{i3}^r \boldsymbol{\phi}_{i2}^r \, d\boldsymbol{x} \\ \int_{K_i} \boldsymbol{\phi}_{i0}^r \boldsymbol{\phi}_{i3}^r \, d\boldsymbol{x} & \dots & \int_{K_i} \boldsymbol{\phi}_{i3}^r \boldsymbol{\phi}_{i3}^r \, d\boldsymbol{x} \end{pmatrix}. \tag{2.10}$$

When formulated for the whole grid, the mass matrix has the entries $\left(\boldsymbol{\phi}_{k\mu}^q, \boldsymbol{\phi}_{i\nu}^r\right)$ for which we can state that

$$\left(\boldsymbol{\phi}_{k\mu}^q, \boldsymbol{\phi}_{i\nu}^r\right) = 0, \qquad \text{for } q \neq r \text{ or } k \neq i. \tag{2.11}$$

Unfortunately, for the above choice of grid and basis functions the mass matrix is full, i.e. not a diagonal matrix, which would be the case if the basis functions were orthogonal respecting the $[L^2]^m$-scalar product, i.e.

$$\left(\boldsymbol{\phi}_{i\mu}^r, \boldsymbol{\phi}_{i\nu}^r\right) = \int_{K_i} \boldsymbol{\phi}_{i\mu}^r \cdot \boldsymbol{\phi}_{i\nu}^r \, d\boldsymbol{x} = \delta_{\mu\nu}. \tag{2.12}$$

Later in this chapter we will present an implementation with triangles which implies an $[L^2]^m$-orthogonality of the basis functions.
If we express $\boldsymbol{w}_h$ in (2.5) in terms of (2.9) and of the vectors

$$\boldsymbol{w}_i^r := \begin{pmatrix} w_{i0}^r(t) \\ w_{i1}^r(t) \\ w_{i2}^r(t) \\ w_{i3}^r(t) \end{pmatrix}, \quad \boldsymbol{b_h}(\boldsymbol{w}_h(t)) := \begin{pmatrix} b_h(\boldsymbol{w}_h(t), \boldsymbol{\phi}_{i0}^r) \\ b_h(\boldsymbol{w}_h(t), \boldsymbol{\phi}_{i1}^r) \\ b_h(\boldsymbol{w}_h(t), \boldsymbol{\phi}_{i2}^r) \\ b_h(\boldsymbol{w}_h(t), \boldsymbol{\phi}_{i3}^r) \end{pmatrix}, \tag{2.13}$$

we get

$$\frac{d}{dt}\mathbb{M}\boldsymbol{w}_i^r(t) + \boldsymbol{b_h}(\boldsymbol{w}_h(t)) = 0, \quad i \in I,\, r = 1, \ldots, m. \tag{2.14}$$

Time Discretization

For the time discretization of problem (1.61) a second order Runge-Kutta method has been implemented:

$$\begin{aligned} \boldsymbol{W}^{(0)} &= \boldsymbol{W}_h^n, \\ \boldsymbol{W}^{(1)} &= \boldsymbol{W}^{(0)} + \tau_n \boldsymbol{\Phi}_h\left(\boldsymbol{W}^{(0)}\right), \\ \boldsymbol{W}^{(2)} &= \boldsymbol{W}^{(1)} + \tau_n \boldsymbol{\Phi}_h\left(\boldsymbol{W}^{(1)}\right), \\ \boldsymbol{W}_h^{n+1} &= \frac{1}{2}\left(\boldsymbol{W}^{(0)} + \boldsymbol{W}^{(2)}\right). \end{aligned} \tag{2.15}$$

However, for the simplicity of the exploration we will formulate the DG update in the following using the explicit Euler method:

$$\boldsymbol{W}_h^{n+1} = \boldsymbol{W}_h^n + \tau_n \boldsymbol{\Phi}_h\left(\boldsymbol{W}_h^n\right). \tag{2.16}$$

Application to problem (1.61) yields

$$\begin{aligned} &\boldsymbol{w}_h^{n+1} \in \boldsymbol{X}_h, \\ &\left(\boldsymbol{w}_h^{n+1}, \boldsymbol{\varphi}_h\right) = \left(\boldsymbol{w}_h^n, \boldsymbol{\varphi}_h\right) - \tau_n b_h\left(\boldsymbol{w}_h^n, \boldsymbol{\varphi}_h\right) \quad \forall \boldsymbol{\varphi}_h \in \boldsymbol{X}_h. \end{aligned} \tag{2.17}$$

or equally

$$\mathbb{M}\left(\boldsymbol{w}_i^r\right)^{n+1} = \mathbb{M}\left(\boldsymbol{w}_i^r\right)^n - \tau_n \boldsymbol{b_h}(\boldsymbol{w}_h(t)) = 0, \quad i \in I,\, r = 1, \ldots, m. \tag{2.18}$$

The inversion of $\mathbb{M}$ yields the vector of unknowns $\left(\boldsymbol{w}_{i\nu}^r\right)^{n+1}$ at the new time level

$$\left(\boldsymbol{w}_i^r\right)^{n+1} = \left(\boldsymbol{w}_i^r\right)^n - \mathbb{M}^{-1}\tau_n \boldsymbol{b_h}(\boldsymbol{w}_h(t)) = 0, \quad i \in I,\, r = 1, \ldots, m. \tag{2.19}$$

Obviously a diagonal mass matrix $\mathbb{M}$ would be desirable.
The time discretization of (1.61) with the second order Runge-Kutta method is analogous.

Space Discretization

We will now discuss how to approximate the integral terms of (2.14). In Chapter 3 we will enter into theoretical results concerning the order of convergence of the DG scheme. A suitable choice of quadrature rules is necessary, if a second order convergence is to be obtained. More precisely, in [5], Cockburn et al. state that for second order convergence on a rectangular grid, the approximations of the volume integrals have to be exact for polynomials of degree 4 and the approximations of the surface integrals for polynomials of degree 5. We will mainly use Gaussian quadrature rules for the discretization of the integral terms. Using k points, the Gauss quadrature rule is exact for polynomials of degree $2k-1$. Consequently on a rectangular grid we have to use Gauss quadrature rules with 3 points for the surface integrals and 3×3 points for the volume integrals.
The one-dimensional Gauss quadrature formula using k points reads as follows

$$\int_a^b f(x)\,dx \approx \frac{b-a}{2}\sum_{\nu=1}^{k} c_\nu f\left(\frac{b-a}{2}x_\nu + \frac{a+b}{2}\right). \tag{2.20}$$

For the volume integrals we apply the one-dimensional Gauss quadrature rule in both space directions and get

$$\int_c^d \int_a^b f(x)\,dxdy \approx \frac{(b-a)(d-c)}{4}\sum_{i,j=1}^{k} c_i c_j f\left(\frac{b-a}{2}x_i + \frac{a+b}{2}, \frac{d-c}{2}y_j + \frac{c+d}{2}\right). \tag{2.21}$$

Discretization of the mass matrix

For each cell in the quadrilateral grid with cell width Δx_1 and Δx_2 we get the following mass matrix

$$\mathbb{M} = \begin{pmatrix} 4 & 2 & 1 & 2 \\ 2 & 4 & 2 & 2 \\ 1 & 2 & 4 & 1 \\ 2 & 2 & 1 & 4 \end{pmatrix} \frac{\Delta x_1 \Delta x_2}{36}, \tag{2.22}$$

and its inverse reads

$$\mathbb{M}^{-1} = \begin{pmatrix} 4 & -2 & 1 & -2 \\ -2 & 4 & -2 & 2 \\ 1 & -2 & 4 & -1 \\ -2 & 1 & -2 & 4 \end{pmatrix} \frac{4}{\Delta x_1 \Delta x_2}. \tag{2.23}$$

In order to get a diagonal mass matrix, we can approximate the integrals

$$\left(\boldsymbol{\phi}_{i\nu}^r \boldsymbol{\phi}_{i\mu}^r\right) = \int_{K_i} \boldsymbol{\phi}_{i\nu}^r \cdot \boldsymbol{\phi}_{i\mu}^r \, d\boldsymbol{x} \tag{2.24}$$

by the trapezoidal rule as

$$\left(\boldsymbol{\phi}_{i\nu}^r \boldsymbol{\phi}_{i\mu}^r\right) \approx \frac{\Delta x_1 \Delta x_2}{4} \sum_{\eta \in V(i)} \boldsymbol{\phi}_{i\nu}^r\left(A_{i\eta}\right) \cdot \boldsymbol{\phi}_{i\mu}^r\left(A_{i\eta}\right). \tag{2.25}$$

Assumption (2.1) implies that

$$\phi_{i\nu}(A_{i\eta}) = \delta_{\nu\eta}, \tag{2.26}$$

so that (2.25) reduces to

$$\begin{aligned}\left(\boldsymbol{\phi}^r_{i\nu}\boldsymbol{\phi}^r_{i\mu}\right) &\approx \frac{\Delta x_1 \Delta x_2}{4} \sum_{\nu,\mu \in V(i)} \boldsymbol{\phi}^r_{i\nu}\left(A_{i\nu}\right) \cdot \boldsymbol{\phi}^r_{i\mu}\left(A_{i\mu}\right) \\ &= \frac{\Delta x_1 \Delta x_2}{4} \delta_{\nu\mu}.\end{aligned} \tag{2.27}$$

Consequently the trapezoidal rule yields the following diagonal approximate mass matrix $\tilde{\mathbb{M}}$

$$\tilde{\mathbb{M}} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \frac{\Delta x_1 \Delta x_2}{4}. \tag{2.28}$$

It should be pointed out that this approximation of the volume integrals is not exact for polynomials of degree 4.
Note that the entries of the mass matrix are integrals over the product of two bilinear functions. Consequently a quadrature rule which is exact for polynomials of degree 4 – as is given by the Gauss quadrature with 3 × 3 points – yields the exact entries of the mass matrix.

Discretization of the form b_h

We will now discretize the form $b_h(\boldsymbol{w}_h(t^n), \boldsymbol{\phi}^{\boldsymbol{r}}_{\boldsymbol{i\nu}})$. The surface integrals have been discretized using Gauss quadrature rules. The Gauss quadrature which is exact for polynomials of degree 3 uses the following computational points and weights

$$\begin{aligned} x_1 &= -\tfrac{1}{\sqrt{3}}, & c_1 &= 1, \\ x_2 &= \tfrac{1}{\sqrt{3}}, & c_2 &= 1. \end{aligned} \tag{2.29}$$

Using

$$\begin{aligned} x_1 &= -\sqrt{\tfrac{3}{5}}, & c_1 &= \tfrac{5}{9}, \\ x_2 &= 0, & c_2 &= \tfrac{8}{9}, \\ x_3 &= \sqrt{\tfrac{3}{5}}, & c_3 &= \tfrac{5}{9}, \end{aligned} \tag{2.30}$$

we get a quadrature rule which is exact for polynomials of degree 5.
For the discretization of the volume integrals we applied the above formulated Gauss quadrature rules in both x_1- and x_2-direction.

Limiting

When approximating discontinuous solutions with piecewise polynomial functions of order one or higher, local maxima and minima can arise in the numerical solution, which are unphysical. To avoid this effect, a limiter has to be applied at each time level. The limiter smooths local steep gradients and spurious oscillations in the numerical solution.
We have applied the so-called *minmod limiter*, cf. [4].
In the following we will formulate this limiting strategy when applied to the DG approximation of a scalar equation. For the sake of notational simplicity we will only consider slopes in x_1-direction. As described above the numerical solution will be stored in the vertices of the cells of the rectangular grid. Averaging yields the values w_m, w_l and w_r of the numerical solution, which are placed in the cell midpoint, the midpoint of the left cell interface and the midpoint of the right cell interface, respectively. By w_{ml}, w_{mr} we denote the value in the midpoint of the left and right neighbouring cell. We then set

$$\begin{aligned} m &= \frac{w_r - w_l}{\Delta x_1}, \\ m_l &= \frac{w_m - w_{ml}}{\Delta x_1}, \\ m_r &= \frac{w_{mr} - w_m}{\Delta x_1}, \end{aligned} \tag{2.31}$$

and apply the minmod strategy, which changes the slope m in the considered cell according to the following formula

$$m = \begin{cases} \min(m, m_l, m_r), & \text{if } \operatorname{sign}(m) = \operatorname{sign}(m_l) = \operatorname{sign}(m_r) \\ = 0, & \text{else.} \end{cases} \tag{2.32}$$

In x_2-direction we procede analogously. Thus locally steep gradients are smoothed out, but the cell averaged value of the solution $\int_K w\,dx$ remains unchanged, so that the conservation property of the DG scheme is not destroyed.

Boundary treatment

In order to be able to apply (1.59) at boundary cells, too, we have implemented a layer of *ghost cells*, lying outside the computational domain. The values in the cells which are placed at the boundary of the computational domain are either copied periodically or extrapolated into these ghost cells. With the help of this implementation, each boundary cell has four uniquely defined neighbouring cells and the fluxes through the cell interfaces can be computed at the boundary equally as in the interior of the computational domain according to (1.59).

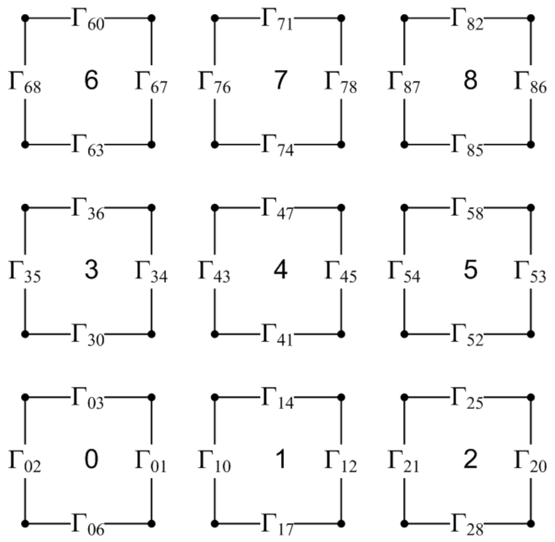


Figure 2.1: Rectangular grid.

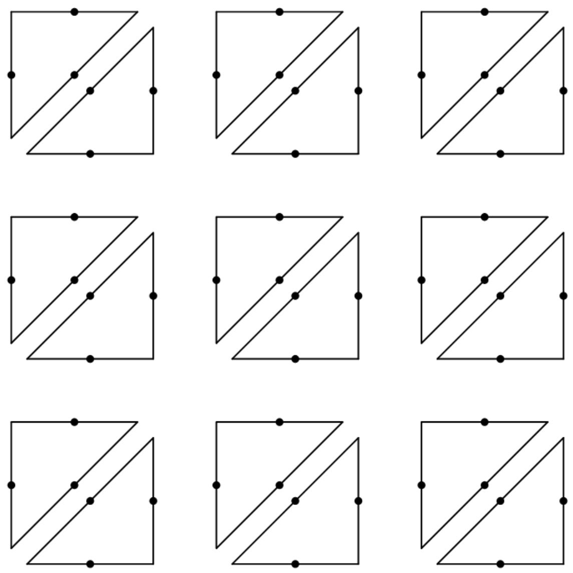


Figure 2.2: Triangular grid.

2.2 Implementation with Triangles

In this work we have besides rectangular meshes examined the effect of a triangular grid and a corresponding choice of basis functions on the accuracy, stability and convergence of the DG scheme. In order to model technically relevant flows in complex geometries, it will be appropriate to subdivide the approximate domain Ω_h into arbitrary triangles, yielding irregular meshes. In this work, we were especially interested in a comparison of different discretizations of the integrals and their effects on the numerical solution. For this purpose we have implemented the DG method on a grid, that is obtained when subdividing each cell of the quadrilateral grid into two triangles as is depicted in Figure 2.2. We should point out that the implementation using irregular triangular grids is analogous, although it is more technical.

Basis functions

For the triangular grid the approximate solution and test functions are piecewise linear. The elements K_i of the grid are numbered with indices $i \in I$, where I is a suitable index set. By $Q_{i\nu}$, $\nu \in V(i) = \{0, \ldots 2\}$ we denote the midpoints of the cell edges of a triangular element K_i. The basis functions $\boldsymbol{\phi}^r_{k\mu}$ are associated with the edge midpoints $Q_{k\mu}$ and are consequently defined by the following condition

$$\begin{aligned}
&B = \left\{\boldsymbol{\phi}^1_{k\mu}, .., \boldsymbol{\phi}^m_{k\mu}\right\}, \quad k \in I,\ \mu \in V(k), \\
&\boldsymbol{\phi}^1_{k\mu} = (\phi_{k\mu}, 0, \ldots, 0)^T, \ \ldots, \ \boldsymbol{\phi}^m_{k\mu} = (0, \ldots, 0, \phi_{k\mu})^T, \\
&\phi_{k\mu}(Q_{i\nu}) = \delta_{ik}\delta_{\nu\mu}.
\end{aligned} \tag{2.33}$$

In the following we will consider the different terms of

$$\frac{d}{dt} M \boldsymbol{w}^r_i(t) + \boldsymbol{b_h}(\boldsymbol{w}_h(t)) = 0, \quad i \in I,\, r = 1, \ldots, m, \tag{2.34}$$

for this choice of grid and basis functions.

Time Discretization

For the time discretization we have implemented the second order Runge-Kutta method presented in the previous section.

Space discretization

In order to integrate the piecewise linear basis functions over a considered cell K_i or cell interface Γ_{ij}, quadrature rules of lower order are sufficient than for the piecewise bilinear functions on our quadrilateral grid. The convergence results presented in Chapter 3 indicate that quadrature rules which are exact for polynomials of degree 2 should be used for the approximation of the volume integrals and that the surface integrals should be approximated with quadrature rules which are exact for polynomials of degree 3 if a second order scheme is to be constructed.

Discretization of the mass matrix

Again we have by definition

$$\left(\boldsymbol{\phi}^r_{i\nu}, \boldsymbol{\phi}^q_{k\mu}\right) = 0, \qquad \text{for } q \neq r \text{ or } i \neq k. \tag{2.35}$$

Note that for this choice of grid and basis functions, we get for a cell K_i with leg length Δx_1 and Δx_2

$$\left(\boldsymbol{\phi}^r_{i\nu}, \boldsymbol{\phi}^r_{i\mu}\right) = \int_{K_i} \boldsymbol{\phi}^r_{i\nu} \cdot \boldsymbol{\phi}^r_{i\mu}\, d\boldsymbol{x} = \delta_{\mu\nu} \frac{\Delta x_1 \Delta x_2}{6}, \tag{2.36}$$

i.e. the basis functions are orthogonal respecting the L^2-scalar product, and $\mathbb{M}$ is diagonal

$$\mathbb{M} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \frac{\Delta x_1 \Delta x_2}{6}. \tag{2.37}$$

Consequently, for the triangular grid, the application of the trapezoidal rule

$$\begin{aligned} \left(\boldsymbol{\phi}^r_{i\mu} \boldsymbol{\phi}^r_{i\nu}\right) &= \frac{\Delta x \Delta y}{6} \sum_{\mu,\nu \in V(i)} \boldsymbol{\phi}^r_{i\mu}\left(Q_{i\mu}\right) \cdot \boldsymbol{\phi}^r_{i\nu}\left(Q_{i\nu}\right) \\ &= \frac{\Delta x_1 \Delta x_2}{6} \delta_{\mu\nu} \end{aligned} \tag{2.38}$$

yields the exact mass matrix $\mathbb{M}$.

Discretization of the form b_h

The surface integrals are approximated with the two-point Gauss quadrature rule, which is exact for polynomials of degree 3.
The volume integrals are approximated by the trapezoidal rule. Summing over the mid-points of the cell interfaces we get the following approximate volume integrals

$$\int_{K_k} \sum_{s=1,2} \boldsymbol{f}_s \cdot \frac{\partial \boldsymbol{\phi}^r_{k\mu}}{\partial x_s}\, d\boldsymbol{x} \approx \frac{\Delta x \Delta y}{6} \sum_{s=1,2} \sum_{\nu \in V(k)} \boldsymbol{f}_s\left(Q_{k\nu}\right) \cdot \frac{\partial \boldsymbol{\phi}^r_{k\mu}}{\partial x_s}\left(Q_{k\nu}\right). \tag{2.39}$$

The trapezoidal rule is exact for polynomials of degree 2.

Limiting

For the triangular grid, it is not intuitively clear how to implement the minmod limiter (2.32) presented in Section 2.1 for the rectangular grid. We have therefore followed the approach of adaptive limiting proposed by Dolejší, see Feistauer et al. [2].
The basic idea of this limiter is to replace the linear numerical solution on a cell K_i by its integral average over this cell, if the discontinuity on the cell interfaces $\Gamma_{ij}, j \in V(i)$ is estimated as too high. Thus, in the vicinity of local discontinuities, the piecewise linear

solution is made constant on chosen cells. The selection of those cells is carried out by the so-called *discontinuity indicator* $g(i)$ which is obtained by integrating over the jumps on the interfaces Γ_{ij} of the considered cell K_i

$$g(i) = \frac{1}{|\Gamma_{ij}|^{\alpha}} \sum_{j \in V(i)} \int_{\Gamma_{ij}} (u_h|_{\Gamma_{ij}} - u_h|_{\Gamma_{ji}})^2 dS, \quad \alpha \in (1,5). \tag{2.40}$$

If we denote by $\boldsymbol{w}_h^{n+1}$ the solution at time t^{n+1} obtained from the Euler update

$$\boldsymbol{w}_h^{n+1} = \boldsymbol{w}_h^n + M^{-1}\tau_n b_h(\boldsymbol{w}_h^n, \boldsymbol{\varphi_h}) = 0, \quad i \in I, \tag{2.41}$$

– or analogously from the Runge-Kutta update – and by $\tilde{\boldsymbol{w}}_h^{n+1}$ the solution at time t^{n+1} modified by the adaptive limiter, we can formulate the limiting strategy in the following way:

$$\tilde{\boldsymbol{w}}_h^{n+1}|_{K_i} = \begin{cases} \frac{1}{|K_i|}\int_{K_i} \boldsymbol{w}_h^{n+1}\, dx, & \text{if } g(i) > 1 \\ \boldsymbol{w}_h^{n+1}|_{K_i}, & \text{else.} \end{cases} \tag{2.42}$$

Chapter 3

Properties of Numerical Methods

Accuracy and Convergence

The accuracy of the numerical method can be measured by the difference between the numerical solution and the exact solution, i.e. by the so-called *global error* ϵ. This can be expressed as

$$\epsilon \equiv \|u(t^n) - U_h^n\|_X \leq \|u(t^n) - P_h u(t^n)\|_X + \|P_h u(t^n) - U_h^n\|_X, \tag{3.1}$$

where the first term at the RHS denotes the projection error generated by a projection P_h onto a suitable piecewise polynomial space. The second term is the evolutionary error, which is in fact generated by the truncation error assuming that the numerical scheme is stable. The truncation error can be computed by means of the Taylor expansion and thus, of course, a smooth exact solution has to be assumed. At discontinuities and steep gradients the truncation error tends to be significant. In order to reduce oscillations of higher order schemes at discontinuities, further techniques, such as limiters, have to be added.
The global error should be proportional to a power of the time step Δt as well as of the mesh size Δx. Usually we have

$$\epsilon = O(\Delta t^p, \Delta x^q). \tag{3.2}$$

We then say that the numerical scheme is of *order* q in time and p in space. The order describes the rate with which the accuracy of the numerical solution increases in dependence of the time step and/or the mesh size.
For the three numerical schemes presented in this work, theoretical results concerning the order of converge can be found in literature, stating that for general scalar multidimensional nonlinear equations, it holds that

$$\epsilon \leq O(\Delta x^{\frac{1}{4}}). \tag{3.3}$$

Only for linear equations or systems the order of the schemes can be derived on the basis of the Taylor expansion. For this case, it can be formally shown for the FV schemes, that

$$\epsilon \leq O(\Delta x^2). \tag{3.4}$$

M. Lukáčová-Medvid'ová et al. confirmed this result for the multidimensional FVEG scheme in [7]. For the DG scheme, implemented with polynomials of order k, Cockburn et al. [5] proved with the help of the Taylor expansion, that in order to obtain a scheme of order $k+1$, the quadrature rules for the volume integrals must be exact for polynomials of degree $2k$ and the approximation of the surface integrals have to be exact for polynomials of degree $2k+1$.
In consequence, using piecewise bilinear functions, we have to approximate the volume integrals by quadrature rules exact for polynomials of degree 4 and the surface integrals by quadrature rules exact for polynomials of degree 5. For piecewise linear functions the quadrature rules have to be exact for polynomials of degree 2 and 3, respectively.
In order to find out whether our implementations of the different schemes reflect this theoretical result, we have inspected the order of convergence by systematically refining the grid and examining the behaviour of the global error. If the exact solution is known, the order of convergence in a certain norm $||\cdot||_X$ can be computed in the following way

$$\mathrm{EOC} = \log_2 \left(\frac{\left\| u^n_{N/2} - u^n_{ref} \right\|_X}{\left\| u^n_N - u^n_{ref} \right\|_X} \right), \tag{3.5}$$

where the EOC is the so-called *experimental order of convergence*, N denotes the number of cells in each direction, u denotes the numerical and u_{ref} the exact solution. A suitable choice of space is $X = L^2(\Omega)$ or $X = L^1(\Omega)$. It is possible, too, to compute the EOC on the basis of the numerical solution obtained on three successively refined grids, if the exact solution is not known. We then get

$$\mathrm{EOC} = \log_2 \left(\frac{\left\| u^n_{N/2} - u^n_N \right\|_X}{\left\| u^n_N - u^n_{2N} \right\|_X} \right). \tag{3.6}$$

In our numerical experiments with the Burgers equation we will use (3.5). In general, it is not possible to compute the exact solution, so that the experimental order of convergence will be computed according to (3.6).

Stability

A prerequisite for convergence is the stability of the numerical method. Stability means that the numerical solution stays bounded as long as the initial data is bounded, i.e.

$$\|U^n\|_X \leq \|U^0\|_X, \quad t^n \in (0,T), \tag{3.7}$$

where X is a suitable vector space. For linear problems $X = L^2(\Omega)$ is a suitable choice and we speak about the *von Neumann stability analysis*, based on the Fourier transformation. For nonlinear problems, such as the shallow water system (5.1), (5.2), this problem is more difficult. For one-dimensional nonlinear systems $X = L^1(\Omega) \cap \mathrm{BV}$ – where BV denotes spaces with bounded variations. In the case of multidimensional scalar equations, such as the Burgers equation, $X = L^\infty(\Omega)$ can be an appropriate choice, cf. [1] for a stability analysis of monotone FV schemes. For time explicit numerical schemes it is necessary to fulfill the so-called *Courant-Friedrichs-Lewy stability condition* (CFL condition) in order to satisfy (3.7). The CFL condition is a condition on the relationship between Δt and Δx. It expresses the fact that the time step has to be chosen so small that any discrete wave can propagate mostly at the next mesh cell at one time step. For a two-dimensional problem we get

$$\Delta t \leq \frac{CFL\,\Delta x_s}{\sigma(\mathbb{A}_s(\boldsymbol{w}^n))}, \quad s = 1, 2. \tag{3.8}$$

Here $\sigma(\mathbb{A}_s)$ denotes the spectral radius of the flux Jacobian $\mathbb{A}_s$ and $\boldsymbol{w}^n$ the solution at time t^n. Methods that only convergence for restricted time steps due to (3.7) but produce oscillations if the time step is not bounded are called *conditionally stable.*

With the von Neumann stability analysis the stability of a numerical method can be investigated and the CFL number can be found. M. Lukáčová-Medvid'ová et al. found with the help of the von Neumann analysis out that the FVEG scheme is stable for CFL numbers smaller than 0.75 for the second order schemes and 0.79 for the first order methods, cf. [7], [9] for more details.

Chapter 4

Burgers Equation

The Burgers equation is one of the simplest nonlinear hyperbolic conservation laws. In two space dimensions it reads as follows

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x_1} + u\frac{\partial u}{\partial x_2} = 0 \text{ in } \Omega \times (0, T). \tag{4.1}$$

Comparison with the general formulation (1.1) of a hyperbolic problem yields

$$\begin{aligned} f(u) &= \frac{u^2}{2}, \\ \mathbb{A}(u) &= u, \end{aligned} \tag{4.2}$$

and we have $\mathbb{A}(u)u \neq f(u)$. The homogeneity condition (1.27) is not fulfilled. We have therefore mainly used the Van Leer scheme for our numerical experiments. In the test cases 2 and 4, however, we show numerical results for the Vijayasundaram scheme and the Steger-Warming scheme, too.

4.1 Test cases

In our numerical experiments we focused on the examination of the accuracy, stability and convergence of the different schemes.
For the Burgers equation we show a comparison of the DG scheme and the FV scheme as well as a of the three different implementations of the DG scheme, presented in the Chapter 2.
In four test cases we will analyse the numerical solution of the Burgers equation (4.1) with the initial data given by

$$u^0(\boldsymbol{x}) = 0.25 + 0.5\sin(\pi(x_1 + x_2)), \quad \boldsymbol{x} \in \Omega. \tag{4.3}$$

On the basis of these test cases we will examine the properties of the different implementations of the DG scheme and of the standard FV scheme. All test cases are based on the following discretizations.

Time Discretization:	second order Runge-Kutta method
Domain size:	$[-1, 1] \times [-1, 1]$
Boundary Treatment:	periodic boundary condition

The test cases 1 and 2 consider the discontinuous solution of (4.1), (4.3) at time $t = 0.4$, cf. Figures 4.1 to 4.2 and 4.7 to 4.8. At $t = 0.1$ the solution of (4.1), (4.3) is smooth and therefore suitable for the computation of the order of convergence of the different schemes, which is examined in the test case 4 and shown in the Tables 4.1 to 4.6. Test case 2 analyses the stability of the numerical solution for both the continuous and the discontinuous case. The results can be seen in Figures 4.3 to 4.6.

Test Case 1 Accuracy

Details

Computation of the numerical fluxes:	Van Leer
Limiting:	minmod
CFL number:	0.45
Simulation time:	0.4 (discontinuous solution)
Grid size:	30×30, 80×80 and 200×200 cells
Implementation of the DG scheme:	rectangles, mass matrix approximated by a trapezoidal rule
Volume integrals in b_h (DG scheme):	Gauss quadrature with 2×2 points
Surface integrals (DG and FV scheme):	Gauss quadrature with 2 points

In Figure 4.1 to Figure 4.2 we compare the accuracy of the DG scheme with that of the FV scheme for problem (4.1), (4.3) on different grids. We can see that for course meshes the discontinuities are slightly smoothed out, but for finer meshes, the solution converges to the exact solution and the discontinuities are resolved sharply, by both schemes.

Test Case 2A Stability

Details	
Computation of the numerical fluxes:	Van Leer
Limiting:	minmod
Simulation time:	0.4 (discontinuous solution)
Grid size:	80×80 cells
Implementation of the DG scheme:	rectangles, mass matrix approximated by a trapezoidal rule
Volume integrals in b_h (DG scheme):	Gauss quadrature with 2×2 points
Surface integrals (DG and FV scheme):	Gauss quadrature with 2 points

We show the discontinuous numerical solution of (4.1), (4.3) for a series of augmenting CFL numbers in Figure 4.3. Here the mesh sizes $\Delta x_1, \Delta x_2$ are constant so that the CFL number is closely related to the time step. One can see a convergence in dependence of the time step, the solution approaches the exact solution for small CFL numbers, whereas at a CFL number $0,5$ the stability limit for the computation of the discontinuous solution is reached, at a CFL number of 0.6 the solution begins to oscillate.

Test Case 2B Stability

Details	
Computation of the numerical fluxes:	Van Leer
Limiting:	minmod
Simulation time:	0.1 (smooth solution)
Grid size:	80×80 cells
Implementation of the DG scheme:	– rectangles, mass matrix approximated by a trapezoidal rule – rectangles, exact mass matrix – triangles
Volume integrals in b_h (DG scheme):	Gauss quadrature with 2×2 points
Surface integrals (DG and FV scheme):	Gauss quadrature with 2 points

A comparison of the different implementations of the DG scheme for the solution of the Burgers equation at $t = 0.1$ in Figure 4.4 to 4.6 shows that not all implementations are equally stable. Whereas the implementation with rectangles and approximate mass matrix is stable at a CFL number of 0.9 for 160×160 cells, and more, cf. Figure 4.5, the implementation with the exact mass matrix is at such a high CFL number only stable for up to 100×100 cells. If the grid is refined further, lower CFL numbers have to be chosen, as can be seen in Figure 4.4. Our implementation with triangles shows already oscillations at 80×80 cells at a CFL number of 0.4, cf. Figure 4.6.

Test Case 3 Flux functions

Details	
Computation of the numerical fluxes:	– Van Leer – Steger-Warming – Vijayasundaram
Limiting:	minmod
Simulation time:	0.4 (discontinuous solution)
CFL number:	0.45
Grid size:	80×80 cells
Implementation of the DG scheme:	rectangles, mass matrix approximated by a trapezoidal rule
Volume integrals in b_h (DG scheme):	Gauss quadrature with 2×2 points
Surface integrals (DG and FV scheme):	Gauss quadrature with 2 points

In the Figures 4.7 and 4.8 the numerical solution of (4.1), (4.3) is shown for different numerical fluxes. Although the Burgers equation does not fulfill the homogeneity condition (1.27), the use of the Vijayasundaram scheme and of the Steger-Warming scheme, does not negatively influence the accuracy or convergence of the schemes.

Test Case 4A Convergence

Details	
Computation of the numerical fluxes:	Van Leer
Limiting:	no limiting
Simulation time:	0.1 (smooth solution)
Implementation of the DG scheme:	
– Implementation 1:	
rectangles	
mass matrix:	approximated by a trapezoidal rule
volume integrals in b_h:	Gauss quadrature with 3×3 points
surface integrals:	Gauss quadrature with 3 points
– Implementation 2:	
rectangles	
mass matrix:	exact
volume integrals in b_h:	Gauss quadrature with 3×3 points
surface integrals:	Gauss quadrature with 3 points
– Implementation 3:	
triangles	
mass matrix:	exact
volume integrals in b_h:	trapezoidal rule
surface integrals:	Gauss quadrature with 2 points
Implementation of the FV scheme:	
Surface integrals:	Gauss quadrature with 3 points

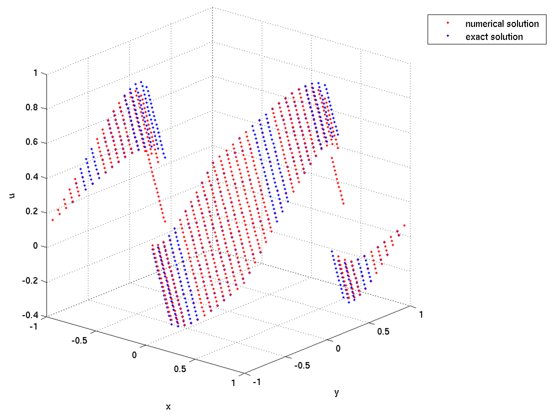
For our convergence experiments in Tables 4.1 to 4.4 we again consider the smooth solution at $t = 0.1$ of problem (4.1), (4.3). When comparing the order of convergence of the different implementations at low CFL numbers, an implementation of the DG scheme with triangles or rectangles and exact mass matrix seems to be desirable. However, the instability of these implementations, which is of course reflected by the EOCs, speaks in favour of an implementation with rectangles and approximate mass matrix. Unfortunately, the approximation of the integrals in the mass matrix with the trapezoidal rule is not enough in order to achieve the second order convergence. Here we obtain an EOC of about 1.49 for the L^2-norm. The EOC computed in the L^1-norm approaches a value of 1.7 for increasing cell numbers. With the FV scheme we have obtained a stable experimental order of convergence of about 2 which confirms the formal truncation error analysis in case of linear problems.

Test Case 4B Convergence

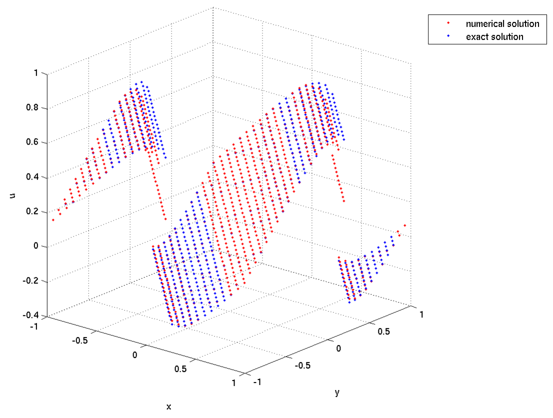
Details

Computation of the numerical fluxes:	– Van Leer
	– Steger-Warming
	– Vijayasundaram
Limiting:	no limiting
Simulation time:	0.1 (smooth solution)
CFL number:	0.4
Implementation of the DG scheme:	rectangles, mass matrix approximated by a trapezoidal rule
Volume integrals in b_h (DG scheme):	Gauss quadrature with 3×3 points
Surface integrals (DG and FV scheme):	Gauss quadrature with 3 points

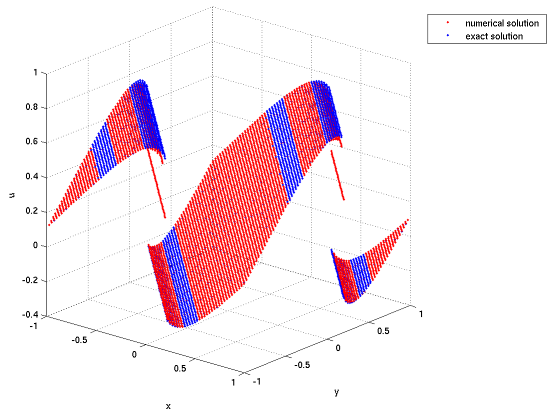
In Tabels 4.5 and 4.6 we show the experimental order of convergence for different numerical flux functions. No relevant differences can be seen in the obtained EOCs.



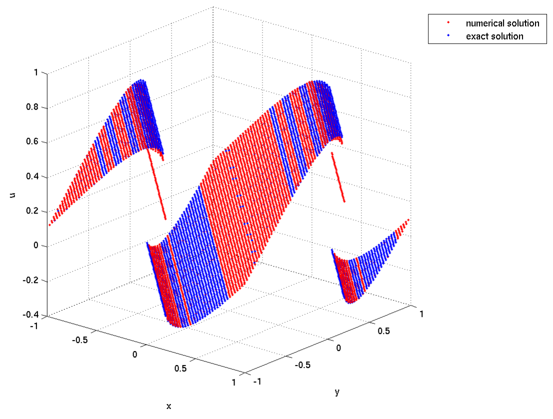
DG scheme, 30 × 30 cells



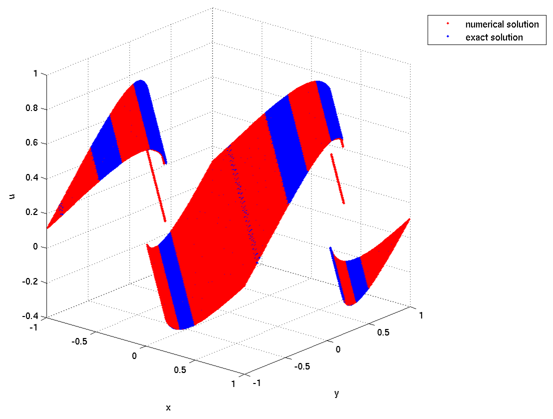
FV scheme, 30 × 30 cells



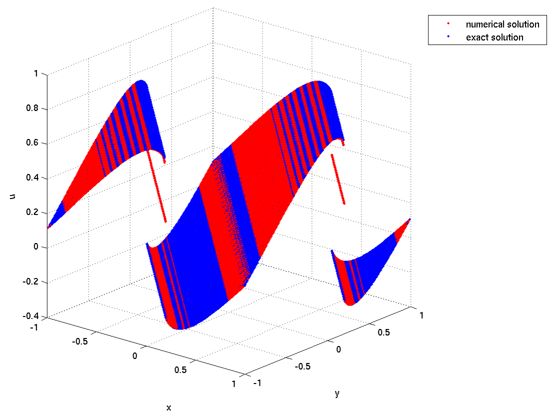
DG scheme, 80 × 80 cells



FV scheme, 80 × 80 cells

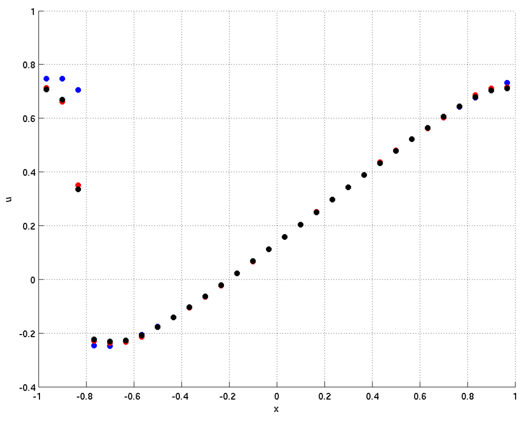


DG scheme, 200 × 200 cells

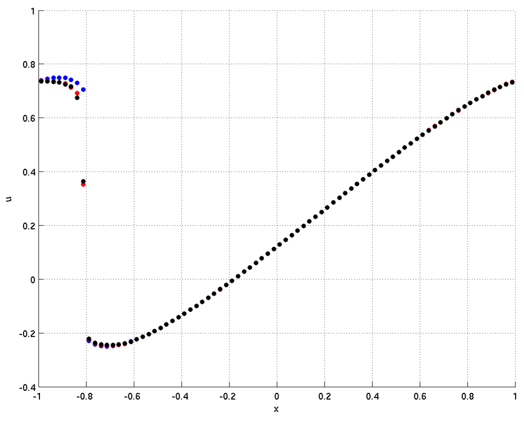


FV scheme, 200 × 200 cells

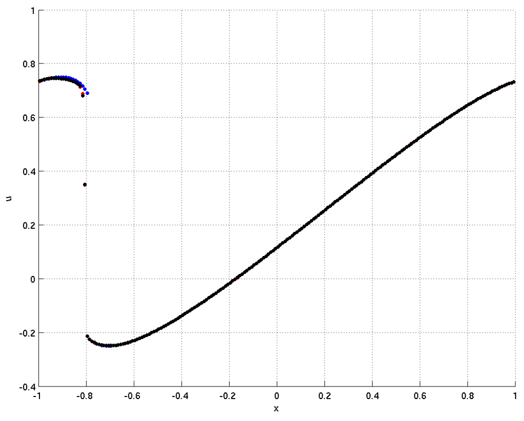
Figure 4.1: Test case 1, accuracy of FV and DG scheme for different grid sizes.



30 × 30 cells

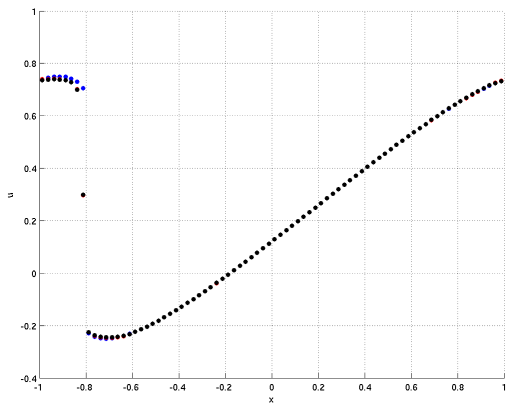


80 × 80 cells

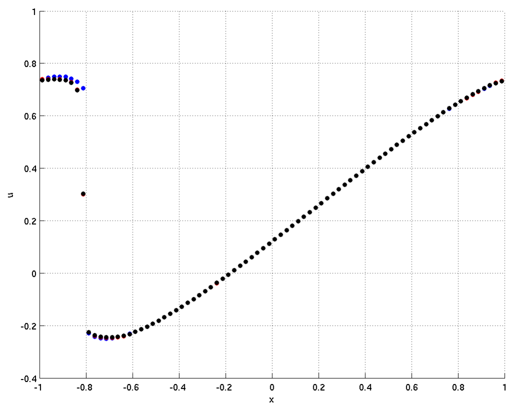


200 × 200 cells

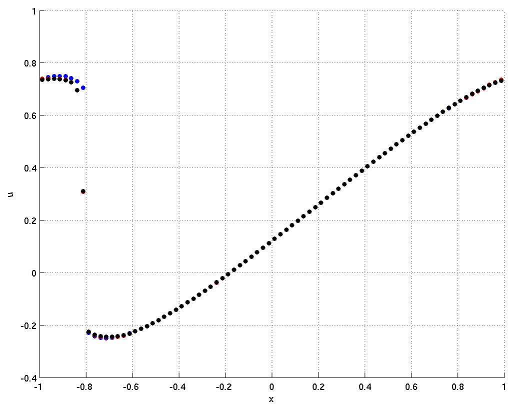
Figure 4.2: Test case 1, accuracy of FV and DG scheme for different grid sizes, u at $y = 0$, blue dots: exact solution, red dots: FV solution, black dots: DG solution, remarkable differences between the lines occur only at the discontinuity.



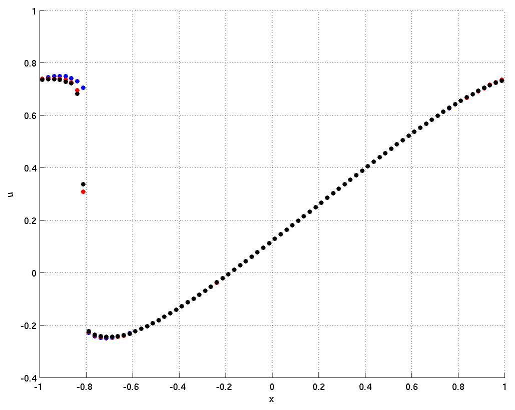
$CFL = 0.1$



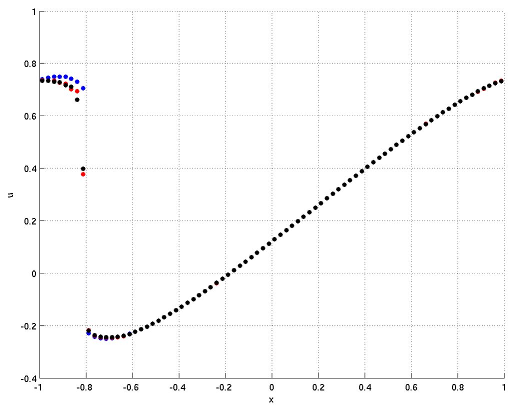
$CFL = 0.2$



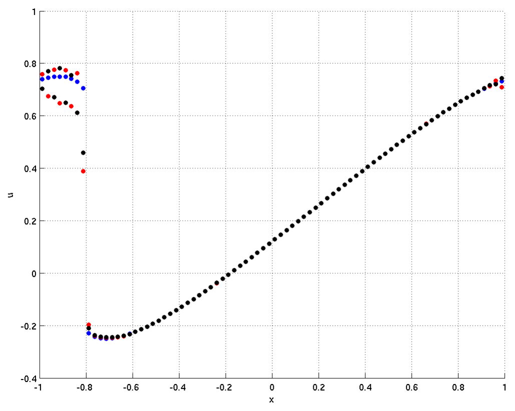
$CFL = 0.3$



$CFL = 0.4$

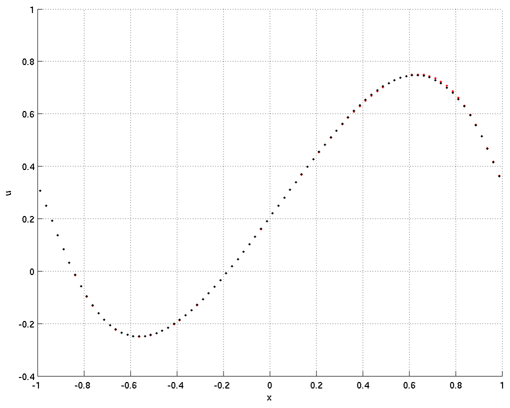


$CFL = 0.5$

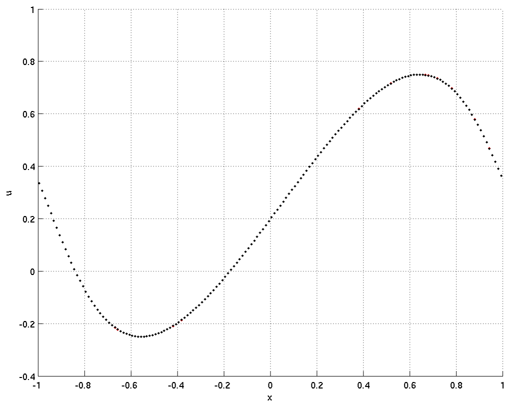


$CFL = 0.6$

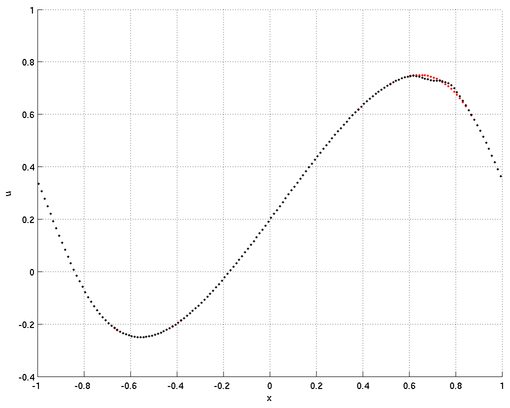
Figure 4.3: Test case 2A, stability of DG and FV scheme for different CFL numbers, u at $y = 0$, blue dots: exact solution, red dots: FV solution, black dots: DG solution.



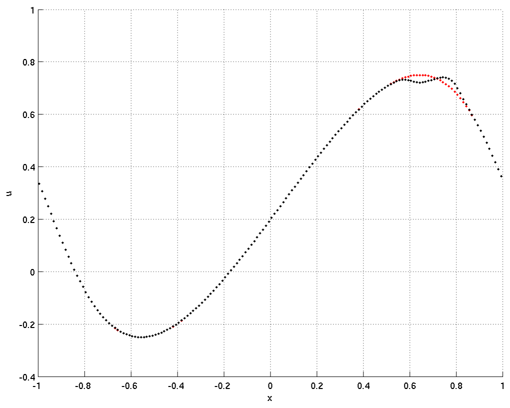
100×100 cells, $CFL = 0.9$



160×160 cells, $CFL = 0.2$

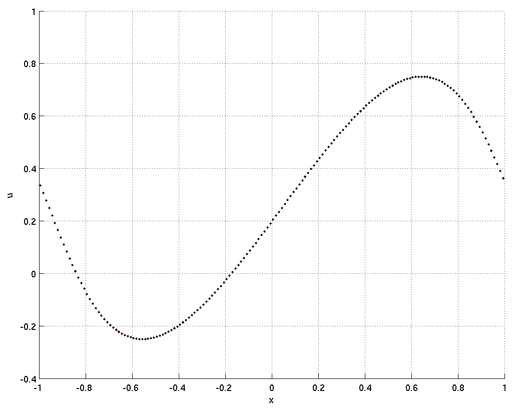


160×160 cells, $CFL = 0.3$



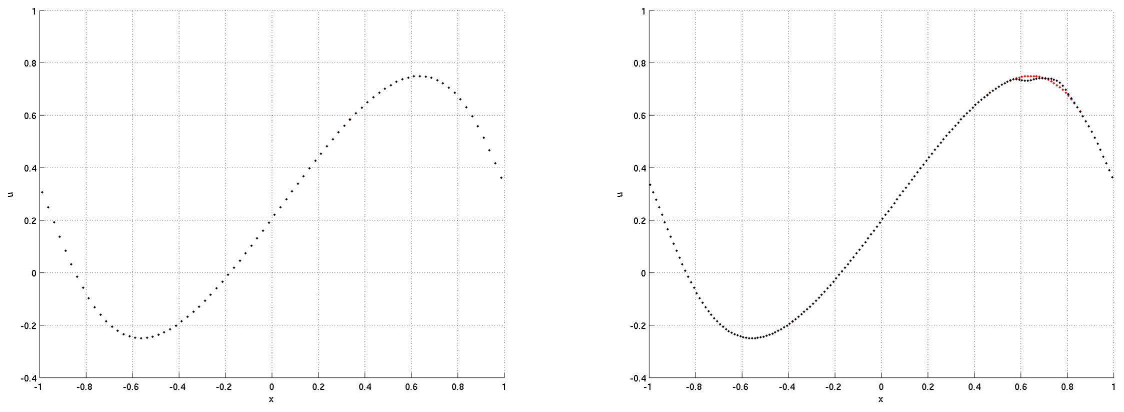
160×160 cells, $CFL = 0.4$

Figure 4.4: Test case 2B, stability of DG scheme for different implementations, rectangles, exact mass matrix, u at $y = 0$, red dots: exact solution, black dots: DG solution.



160×160 cells, $CFL = 0.9$

Figure 4.5: Test case 2B, stability of DG scheme for different implementations, rectangles, mass matrix with a trapezoidal rule, u at $y = 0$, red dots: exact solution, black dots: DG solution.



80×80 cells, $CFL = 0.3$ 80×80 cells, $CFL = 0.4$

Figure 4.6: Test case 2B, stability of DG scheme for different implementations, triangles, u at $y = 0$, red dots: exact solution, black dots: DG solution.

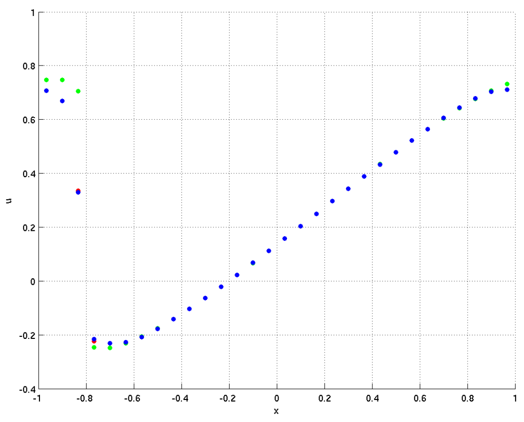


Figure 4.7: Test case 3, DG scheme for different numerical fluxes, u at $y = 0$, green dots: exact solution, black dots: Van Leer, red dots: Vijayasundaram, blue dots: Steger Warming, no remarkable differences between the numerical fluxes are visible.

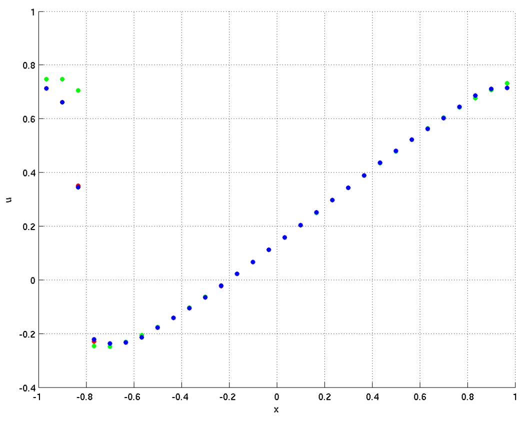


Figure 4.8: Test case 3, FV scheme for different numerical fluxes, u at $y = 0$, green dots: exact solution, black dots: Van Leer, red dots: Vijayasundaram, blue dots: Steger Warming, no remarkable differences between the numerical fluxes are visible.

$N \times N$ cells	$\lVert u_N^n - u_{ref}^n \rVert_{L^2}$	EOC_{L^2}	$\lVert u_N^n - u_{ref}^n \rVert_{L^1}$	EOC_{L^1}
20×20	0.002000		0.002885	
40×40	0.000444	2.171368	0.000659	2.130221
80×80	0.000100	2.150560	0.000156	2.078732
160×160	0.000024	2.058894	0.000038	2.037475
320×320	0.000006	2.000000	0.000010	1.925999

EOCs for $CFL = 0.1$

$N \times N$ cells	$\lVert u_N^n - u_{ref}^n \rVert_{L^2}$	EOC_{L^2}	$\lVert u_N^n - u_{ref}^n \rVert_{L^1}$	EOC_{L^1}
20×20	0.003050		0.004430	
40×40	0.000848	1.846673	0.001180	1.908520
80×80	0.000230	1.882430	0.000309	1.933108
160×160	0.000062	1.891294	0.000082	1.913911
320×320	0.000016	1.954196	0.000021	1.965235

EOCs for $CFL = 0.4$

Table 4.1: Test case 4A, Convergence of FV scheme for different CFL numbers.

$N \times N$ cells	$\lVert u_N^n - u_{ref}^n \rVert_{L^2}$	EOC_{L^2}	$\lVert u_N^n - u_{ref}^n \rVert_{L^1}$	EOC_{L^1}
20×20	0.022113		0.034918	
40×40	0.007894	1.486066	0.011794	1.565918
80×80	0.002820	1.485061	0.003900	1.596507
160×160	0.001001	1.494253	0.001234	1.660132
320×320	0.000354	1.499621	0.000375	1.718380

EOCs for $CFL = 0.1$

$N \times N$ cells	$\lVert u_N^n - u_{ref}^n \rVert_{L^2}$	EOC_{L^2}	$\lVert u_N^n - u_{ref}^n \rVert_{L^1}$	EOC_{L^1}
20×20	0.021296		0.033472	
40×40	0.007675	1.472344	0.011413	1.552277
80×80	0.002773	1.468719	0.003811	1.582437
160×160	0.000992	1.483036	0.001211	1.653971
320×320	0.000352	1.494765	0.000370	1.710602

EOCs for $CFL = 0.4$

Table 4.2: Test case 4A, convergence of DG scheme for different CFL numbers, implementation: rectangles, mass matrix approximated with a trapezoidal rule.

$N \times N$ cells	$\left\|u_N^n - u_{ref}^n\right\|_{L^2}$	EOC_{L^2}	$\left\|u_N^n - u_{ref}^n\right\|_{L^1}$	EOC_{L^1}
20×20	0.013159		0.021236	
40×40	0.003363	1.968229	0.005412	1.972278
80×80	0.000851	1.982518	0.001365	1.987261
160×160	0.000214	1.991548	0.000343	1.992620
320×320	0.000054	1.986579	0.000086	1.995800

EOCs for $CFL = 0.1$

$N \times N$ cells	$\left\|u_N^n - u_{ref}^n\right\|_{L^2}$	EOC_{L^2}	$\left\|u_N^n - u_{ref}^n\right\|_{L^1}$	EOC_{L^1}
20×20	0.013080		0.021250	
40×40	0.003352	1.964268	0.005417	1.971897
80×80	0.000848	1.982886	0.001368	1.985426
160×160	0.000217	1.966369	0.000349	1.970769
320×320	0.008637	-5.314763	0.002584	-2.888307

EOCs for $CFL = 0.2$

$N \times N$ cells	$\left\|u_N^n - u_{ref}^n\right\|_{L^2}$	EOC_{L^2}	$\left\|u_N^n - u_{ref}^n\right\|_{L^1}$	EOC_{L^1}
20×20	0.012923		0.021263	
40×40	0.003348	1.948570	0.005497	1.951629
80×80	0.015604	-2.220544	0.007657	-0.478135
160×160	0.056578	-1.858325	0.023571	-1.622162
320×320	0.040961	0.465990	0.016305	0.531699

EOCs for $CFL = 0.3$

Table 4.3: Test case 4A, convergence of DG scheme for different CFL numbers, implementation: rectangles, exact mass matrix.

$N \times N$ cells	$\left\|u_N^n - u_{ref}^n\right\|_{L^2}$	EOC_{L^2}	$\left\|u_N^n - u_{ref}^n\right\|_{L^1}$	EOC_{L^1}
20×20	0.011015		0.016691	
40×40	0.003021	1.866371	0.004494	1.892998
80×80	0.000787	1.940591	0.001164	1.948909
160×160	0.000201	1.969168	0.000295	1.980304
320×320	0.000051	1.978626	0.000075	1.975752

EOCs for $CFL = 0.1$

$N \times N$ cells	$\left\|u_N^n - u_{ref}^n\right\|_{L^2}$	EOC_{L^2}	$\left\|u_N^n - u_{ref}^n\right\|_{L^1}$	EOC_{L^1}
20×20	0.012354		0.018048	
40×40	0.002886	2.097835	0.004297	2.070437
80×80	0.000815	1.824199	0.001190	1.852368
160×160	0.000219	1.895869	0.000311	1.935975
320×320	0.000059	1.892144	0.000080	1.958843

EOCs for $CFL = 0.3$

$N \times N$ cells	$\left\|u_N^n - u_{ref}^n\right\|_{L^2}$	EOC_{L^2}	$\left\|u_N^n - u_{ref}^n\right\|_{L^1}$	EOC_{L^1}
20×20	0.027974		0.032120	
40×40	0.012261	1.190010	0.012568	1.353717
80×80	0.022053	-0.846898	0.016827	-0.421023
160×160	0.043989	-0.996168	0.027987	-0.733979
320×320	0.037618	0.225720	0.020683	0.436311

EOCs for $CFL = 0.4$

Table 4.4: Test case 4A, convergence of DG scheme for different CFL numbers, implementation: triangles.

$N \times N$ cells	$\left\|u_N^n - u_{ref}^n\right\|_{L^2}$	EOC_{L^2}	$\left\|u_N^n - u_{ref}^n\right\|_{L^1}$	EOC_{L^1}
20×20	0.021296		0.033472	
40×40	0.007675	1.472344	0.011413	1.552277
80×80	0.002773	1.468719	0.003811	1.582437
160×160	0.000992	1.483036	0.001211	1.653971
320×320	0.000352	1.494765	0.000370	1.710602

EOCs for Van Leer

$N \times N$ cells	$\left\|u_N^n - u_{ref}^n\right\|_{L^2}$	EOC_{L^2}	$\left\|u_N^n - u_{ref}^n\right\|_{L^1}$	EOC_{L^1}
20×20	0.021296		0.033472	
40×40	0.007675	1.472344	0.011413	1.552277
80×80	0.002773	1.468719	0.003811	1.582437
160×160	0.000992	1.483036	0.001211	1.653971
320×320	0.000352	1.494765	0.000370	1.710602

EOCs for Steger Warming

$N \times N$ cells	$\left\|u_N^n - u_{ref}^n\right\|_{L^2}$	EOC_{L^2}	$\left\|u_N^n - u_{ref}^n\right\|_{L^1}$	EOC_{L^1}
20×20	0.021296		0.033472	
40×40	0.007675	1.472344	0.011413	1.552277
80×80	0.002773	1.468719	0.003811	1.582437
160×160	0.000992	1.483036	0.001211	1.653971
320×320	0.000352	1.494765	0.000370	1.710602

EOCs for Vijayasundaram

Table 4.5: Test case 4B, convergence of DG scheme for different numerical fluxes.

$N \times N$ cells	$\left\|u_N^n - u_{ref}^n\right\|_{L^2}$	EOC_{L^2}	$\left\|u_N^n - u_{ref}^n\right\|_{L^1}$	EOC_{L^1}
20×20	0.003050		0.004430	
40×40	0.000848	1.846673	0.001180	1.908520
80×80	0.000230	1.882430	0.000309	1.933108
160×160	0.000062	1.891294	0.000082	1.913911
320×320	0.000016	1.954196	0.000021	1.965235

EOCs for Van Leer

$N \times N$ cells	$\left\|u_N^n - u_{ref}^n\right\|_{L^2}$	EOC_{L^2}	$\left\|u_N^n - u_{ref}^n\right\|_{L^1}$	EOC_{L^1}
20×20	0.003049		0.004454	
40×40	0.000844	1.853021	0.001176	1.921213
80×80	0.000230	1.875609	0.000309	1.928209
160×160	0.000062	1.891294	0.000082	1.913911
320×320	0.000016	1.954196	0.000021	1.965235

EOCs for Steger-Warming

$N \times N$ cells	$\left\|u_N^n - u_{ref}^n\right\|_{L^2}$	EOC_{L^2}	$\left\|u_N^n - u_{ref}^n\right\|_{L^1}$	EOC_{L^1}
20×20	0.003050		0.004430	
40×40	0.000848	1.846673	0.001180	1.908520
80×80	0.000230	1.882430	0.000309	1.933108
160×160	0.000062	1.891294	0.000082	1.913911
320×320	0.000016	1.954196	0.000021	1.965235

EOCs for Vijaysundaram

Table 4.6: Test case 4B, Convergence of FV scheme for different numerical fluxes.

Chapter 5

Shallow Water Equations

The shallow water equation system plays an important role in the modeling of a variety of free surface water flows, such as oceanographic flows and flows in lakes and rivers. We can find their applications in modeling atmospheric and geophysical flows. These types of flows are all characterized by a free surface, by the influence of gravity and by the vertical dimension being negligible in comparison to the horizontal dimensions.
The homogeneous shallow water equations written in conservative variables have the following form

$$\frac{\partial}{\partial t}\boldsymbol{U} + \frac{\partial}{\partial x}\boldsymbol{f}_1(\boldsymbol{U}) + \frac{\partial}{\partial y}\boldsymbol{f}_2(\boldsymbol{U}) = 0, \tag{5.1}$$

with

$$\begin{aligned} \boldsymbol{U} &= \begin{pmatrix} h \\ hu \\ hv \end{pmatrix}, \\ \boldsymbol{f}_1 &= \begin{pmatrix} hu \\ hu^2 + \frac{1}{2}gh^2 \\ huv \end{pmatrix}, \quad \boldsymbol{f}_2 = \begin{pmatrix} hv \\ hvu \\ hv^2 + \frac{1}{2}gh^2 \end{pmatrix}. \end{aligned} \tag{5.2}$$

Here and in what follows g is the gravitational accelaration.

Hyperbolic Structure

For the computation of the numerical fluxes we need the hyperbolic structure of the flux functions $\boldsymbol{f}_1, \boldsymbol{f}_2$. We denote by $\mathbb{A}_1, \mathbb{A}_2$ the Jacobians of $\boldsymbol{f}_1, \boldsymbol{f}_2$, respectively, by λ_i^1, λ_i^2 their eigenvalues and by $\boldsymbol{l}_i^1, \boldsymbol{l}_i^2$ and $\boldsymbol{r}_i^1, \boldsymbol{r}_i^2$ their left and right eigenvectors, $i = 0, 1, 2$. Furthermore we define

$$a = \sqrt{gh}. \tag{5.3}$$

The hyperbolic structure of $\boldsymbol{f}_1$ then reads

$$\begin{aligned}
&\mathbb{A}_1 = \begin{pmatrix} 0 & 1 & 0 \\ a^2 - u^2 & 2u & 0 \\ -uv & v & u \end{pmatrix}, \\
&\lambda_0^1 = u - a,\ \lambda_1^1 = u,\ \lambda_2^1 = u + a, \\
&\boldsymbol{R}_0^1 = \alpha_0 \begin{pmatrix} 1 \\ u - a \\ v \end{pmatrix}, \quad \boldsymbol{R}_1^1 = \alpha_1 \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \quad \boldsymbol{R}_2^1 = \alpha_2 \begin{pmatrix} 1 \\ v + a \\ v \end{pmatrix}, \\
&\boldsymbol{L}_0^2 = \tilde{\alpha_0} \begin{pmatrix} u + a \\ -1 \\ 0 \end{pmatrix}^T, \quad \boldsymbol{L}_1^2 = \tilde{\alpha_1} \begin{pmatrix} -v \\ 0 \\ 1 \end{pmatrix}^T, \quad \boldsymbol{L}_2^2 = \tilde{\alpha_2} \begin{pmatrix} u - a \\ -1 \\ 0 \end{pmatrix}^T.
\end{aligned} \tag{5.4}$$

For $\boldsymbol{f}_2$ we have

$$\begin{aligned}
&\mathbb{A}_2 = \begin{pmatrix} 0 & 0 & 1 \\ -uv & v & u \\ a^2 - v^2 & 0 & 2v \end{pmatrix}, \\
&\lambda_0^2 = v - a,\ \lambda_1^2 = v,\ \lambda_2^2 = v + a, \\
&\boldsymbol{R}_0^2 = \beta_0 \begin{pmatrix} 1 \\ u \\ v - a \end{pmatrix}, \quad \boldsymbol{R}_1^2 = \beta_1 \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, \quad \boldsymbol{R}_2^2 = \beta_2 \begin{pmatrix} 1 \\ u \\ v + a \end{pmatrix}, \\
&\boldsymbol{L}_0^2 = \tilde{\beta}_0 \begin{pmatrix} v + a \\ 0 \\ -1 \end{pmatrix}^T, \quad \boldsymbol{L}_1^2 = \tilde{\beta}_1 \begin{pmatrix} -u \\ 1 \\ 0 \end{pmatrix}^T, \quad \boldsymbol{L}_2^2 = \tilde{\beta}_2 \begin{pmatrix} v - a \\ 0 \\ -1 \end{pmatrix}^T.
\end{aligned} \tag{5.5}$$

The scaling factors $\alpha_i, \tilde{\alpha_i}$ and $\beta_i, \tilde{\beta}_i$ must be chosen such that

$$L_i \cdot R_j = \delta_{ij}. \tag{5.6}$$

This condition yields the following relations

$$\begin{aligned}
\alpha_0 = \frac{1}{2a\tilde{\alpha}_0}, \quad \alpha_1 = \frac{1}{\tilde{\alpha}_1}, \quad \alpha_2 = -\frac{1}{2a\tilde{\alpha}_2}, \\
\beta_0 = \frac{1}{2a\tilde{\beta}_0}, \quad \beta_1 = \frac{1}{\tilde{\beta}_1}, \quad \beta_2 = -\frac{1}{2a\tilde{\beta}_2}.
\end{aligned} \tag{5.7}$$

For the shallow water equations we have

$$\boldsymbol{A}_1 \boldsymbol{U} = \begin{pmatrix} 0 & 1 & 0 \\ a^2 - u^2 & 2u & 0 \\ -uv & v & u \end{pmatrix} \begin{pmatrix} h \\ hu \\ hv \end{pmatrix} = \begin{pmatrix} hu \\ hu^2 + gh^2 \\ huv \end{pmatrix} \neq \boldsymbol{f}_1 (\boldsymbol{U}), \tag{5.8}$$

$$\boldsymbol{A}_2\boldsymbol{U} = \begin{pmatrix} 0 & 0 & 1 \\ -uv & v & u \\ a^2 - v^2 & 0 & 2v \end{pmatrix} \begin{pmatrix} h \\ hu \\ hv \end{pmatrix} = \begin{pmatrix} hv \\ huv \\ hv^2 + gh^2 \end{pmatrix} \neq \boldsymbol{f}_2(\boldsymbol{U}) . \tag{5.9}$$

The homogeneity condition (1.27) does not hold for the shallow water equations, either. An application of the Vijayasundaram scheme and of the Steger-Warming scheme is therefore not justified. Our numerical experiments confirm this fact, cf. test case 3, Figures 5.25 to 5.36.

5.1 Test cases

For the shallow water equations our main goal is the comparison between the DG scheme and the FVEG scheme. The numerical experiments with the DG scheme have been carried out with rectangles and approximate mass matrix. The stability results for the Burgers equation described above confirm the applicability of this choice. The present implementation of the FVEG scheme is based on rectangles, too, so that the comparison is relevant. Again we show results with standard FV schemes, which are also implemented on a rectangular grid.
In three test cases we have analysed the accuracy and stability of the discontinuous solution of the so-called *Sod problem*. It describes a two-dimensional dam break and considers the shallow water equations (5.1),(5.2) for the following initial data

$$\begin{aligned} &h = 0.1, \quad u = 0, \quad v = 0, \quad \text{for } ||\boldsymbol{x}|| > 0.3,\ t = 0, \\ &h = 1, \quad\ \ u = 0, \quad v = 0, \quad \text{for } ||\boldsymbol{x}|| < 0.3,\ t = 0. \end{aligned} \tag{5.10}$$

In the fourth test case we again consider the order of convergence of the different schemes. To this end we computed the smooth numerical solution of the shallow water equations (5.1),(5.2) with initial data

$$\begin{aligned} &h = 0.25, \\ &u = 1 + 0.5\sin(\pi y) + 0.25\cos(\pi x), \\ &v = 1 + 0.25\sin(\pi x) + 0.5\cos(\pi y), \quad x, y \in \Omega,\ t = 0. \end{aligned} \tag{5.11}$$

The numerical solution of (5.1),(5.2), (5.10) is depicted in the Figures 5.1 to 5.36. The convergence of the methods for problem (5.1),(5.2), (5.11) can be seen in the Tables 5.1 to 5.3.
The following choice of discretizations and parameters are valid for all test cases.

Time Discretization	
– DG and standard FV scheme:	second order Runge-Kutta method
– FVEG scheme:	midpoint rule
Implementation of the DG scheme:	rectangles, mass matrix approximated by a trapezoidal rule
Gravitational acceleration g:	10

Test Case 1 Accuracy

Details	
Computation of the numerical fluxes:	Van Leer
Limiting:	minmod
Boundary Treatment:	constant extrapolation
CFL number:	0.45
Simulation time:	0.25
Domain size:	$[-1, 1] \times [-1, 1]$
Grid size:	100×100 and 200×200 cells
Volume integrals in b_h (DG scheme):	Gauss quadrature with 2×2 points
Surface integrals	
– DG and FV scheme:	Gauss quadrature with 2 points
– FVEG scheme:	Simpson rule

The accuracy with which the discontinuous solution of the Sod problem (5.1), (5.2), (5.10) is resolved by the DG scheme, the FVEG scheme and the standard FV scheme can be seen in Figure 5.1 to Figure 5.20. The numerical solutions obtained with these three schemes, are comparable in accuracy. The plots at $y = 0$ show slight differences in the second velocity component. Note however that the scale goes to 10^{-2}. The differences at the boundary, which attract one's attention in Figure 5.10, are only due to a different implementation of the boundary condition. Whereas for the FVEG scheme the ghost cells, which contain the extrapolated values at the boundary, are placed within the computational domain, the other two schemes are implemented with the ghost cell layers lying without the domain.
Important differences between the three schemes occur only in the resolution of multidimensional effects. The velocity contour plots show that the sharpest resolution of these effects is obtained with the FVEG scheme.

Test Case 2 Stability

Details	
Computation of the numerical fluxes:	Van Leer
Limiting:	minmod
Boundary Treatment:	constant extrapolation
Simulation time:	0.25
Domain size:	$[-1, 1] \times [-1, 1]$
Grid size:	100×100 cells
Volume integrals in b_h (DG scheme):	Gauss quadrature with 2×2 points
Surface integrals	
– DG and FV scheme:	Gauss quadrature with 2 points
– FVEG scheme:	Simpson rule

For the purpose of finding out the stability limits of the DG and the standard FV scheme we show plots of the numerical solution of the Sod problem (5.1), (5.2), (5.10) for high CFL numbers, cf. Figure 5.21 to Figure 5.24. We have already mentioned in Chapter 3 that the FVEG is stable for CFL numbers smaller than 0.75. The stability limits that we obtained for the FV scheme and the DG scheme are depicted in Figures 5.21 to 5.24. The DG scheme shows first small oscillations in the order of 10^{-2} at a CFL number of 0.9, for the FV scheme these oscillations are first visible at a CFL number of 0.7.

Test Case 3 Flux functions

Details	
Computation of the numerical fluxes:	– Van Leer
	– Steger-Warming
	– Vijayasundaram
Boundary Treatment:	constant extrapolation
Limiting:	minmod
CFL number:	0.45
Simulation time:	0.25
Domain size:	$[-1, 1] \times [-1, 1]$ and $[-1.5, 1.5] \times [-1.5, 1.5]$
Grid size:	100×100 cells and 150×150 cells
Volume integrals in b_h (DG scheme):	Gauss quadrature with 2×2 points
Surface integrals	
– DG and FV scheme:	Gauss quadrature with 2 points
– FVEG scheme:	Simpson rule

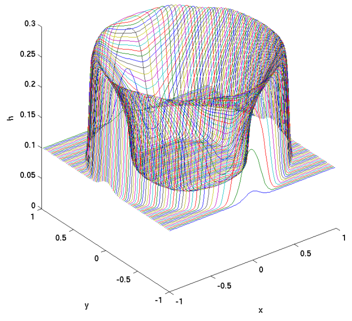
In this test case we have applied different numerical fluxes and compared the obtained numerical solution of the Sod problem (5.1), (5.2), (5.10), which can be seen in Figure 5.25 to Figure 5.36. In order to avoid an influence of boundary effects on the numerical solution, we have computed the solution on two different domain sizes $[-1, 1] \times [-1, 1]$ and

$[-1.5, 1.5] \times [-1.5, 1.5]$ with 100×100 cells and 150×150 cells, respectively.
In contrast to the Burgers equation, we can see for the shallow water equations an evident difference between the numerical flux functions. The Figures 5.25 to 5.36 show that the Vijaysundaram scheme and the Steger-Warming do not approximate the fluxes appropriately. These effects are equally visible for the DG scheme and the FV scheme. Especially at the boundary the accuracy of the solution suffers from the inappropriate application of these schemes. When considering a larger domain size, these problems are partially removed, but the Van Leer scheme is again clearly more accurate. This phenomenon can be explained by the homogeneity condition (1.27) which is not fulfilled by the shallow water equation and required by the Vijaysundaram scheme and the Steger-Warming scheme.

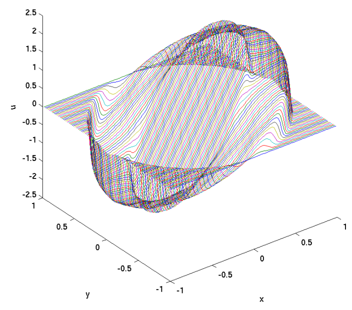
Test Case 4 Convergence

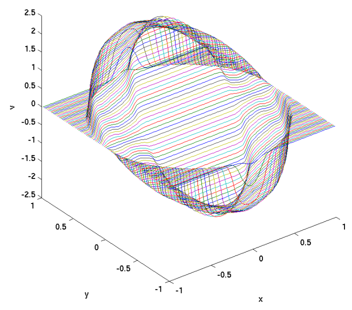
Details	
Computation of the numerical fluxes:	Van Leer
Boundary Treatment:	periodic boundary condition
Limiting	no limiting
CFL number:	0.4
Simulation time:	0.2
Domain size:	$[-1, 1] \times [-1, 1]$
Volume integrals in b_h (DG scheme):	Gauss quadrature with 3×3 points
Surface integrals	
– DG and FV scheme:	Gauss quadrature with 3 points
– FVEG scheme:	Simpson rule

Finally, we will again focus on the EOCs of the different schemes. To this end we consider in Tables 5.1 to 5.3 the smooth solution of problem (5.1), (5.2), (5.11). The FVEG scheme as well as the FV scheme show a stable EOC of about 2. Astonishingly, for the shallow water equations, the EOCs we obtained with the DG scheme are higher than those obtained for the Burgers equation with the same implementation, cf. Table 4.2. Here, the EOCs of the DG scheme approach those obtained with the FVEG scheme and the FV scheme. Note however that the absolute value of the global error indicates a superiority of the FV schemes in accuracy.
Apart from these numerical results we can further state some advantages and disadvantages of the different schemes. For the DG scheme, the choice of a wide variety of elements and basis functions is possible, and easily implemented. Thus complex geometries can be easily discretized, and higher order schemes can be constructed. For the FVEG scheme an implementation with triangles is possible, although more technical effort is necessary than for the implementation with rectangles, cf. [10]. The construction of higher order finite volume schemes is more technical, too, cf. [3]. The standard FV scheme can accommodate any cell form.



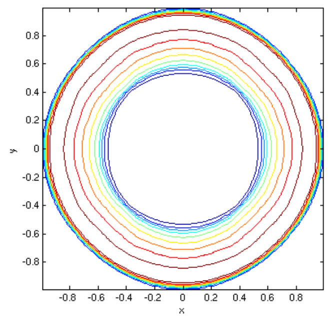
h



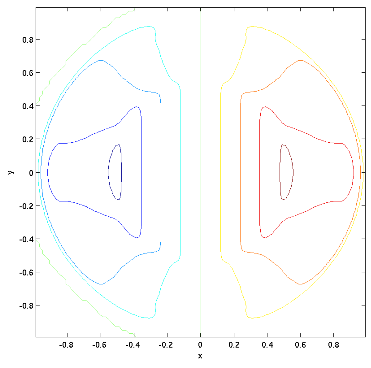


u v

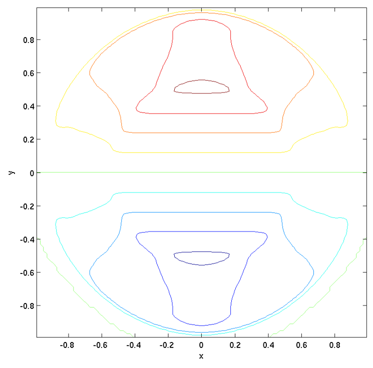
Figure 5.1: Test case 1, accuracy of DG scheme, 100×100 cells.



h

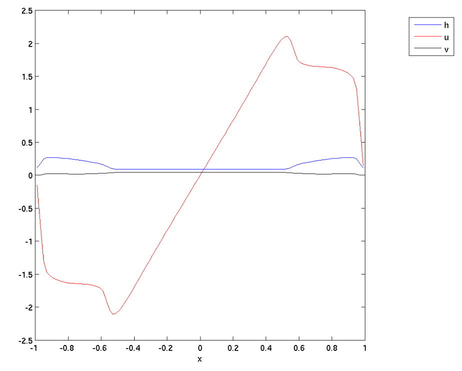


u

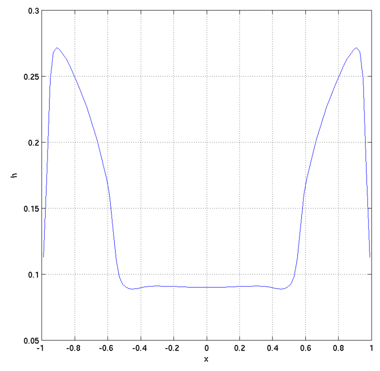


v

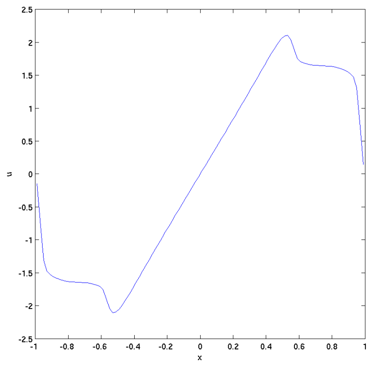
Figure 5.2: Test case 1, accuracy of DG scheme, contour plots, 100×100 cells.



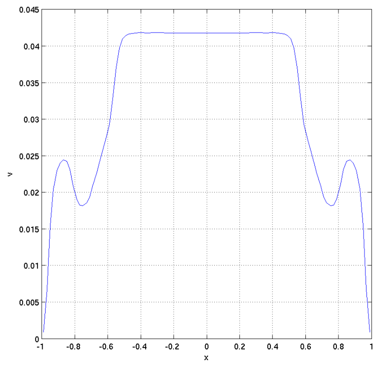
h, u, v



h

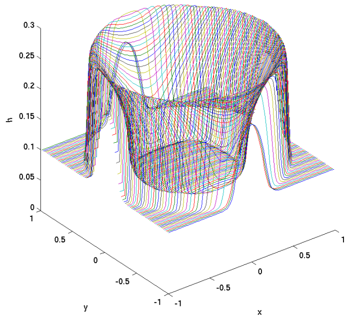


u

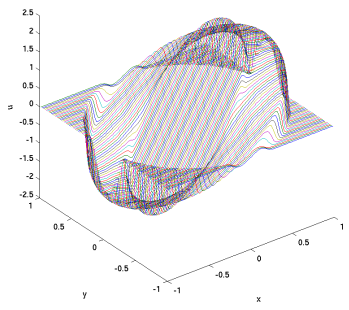


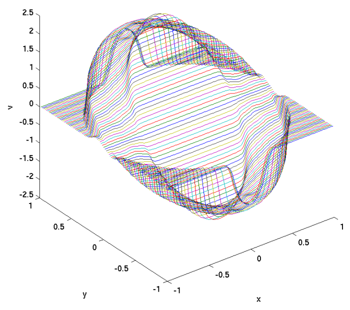
v

Figure 5.3: Test case 1, accuracy of DG scheme, plots at $y = 0$, 100×100 cells.



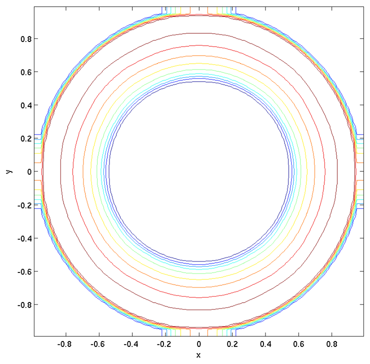
h



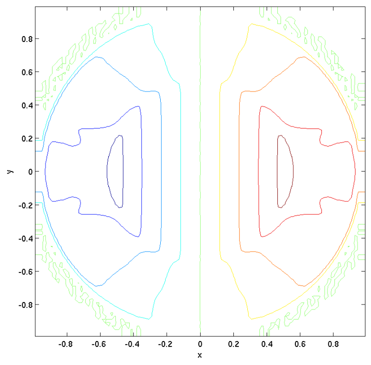


u *v*

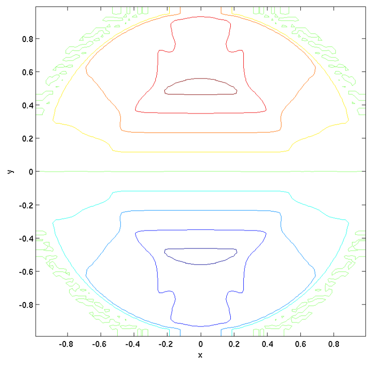
Figure 5.4: Test case 1, accuracy of FVEG scheme, 100×100 cells.



h

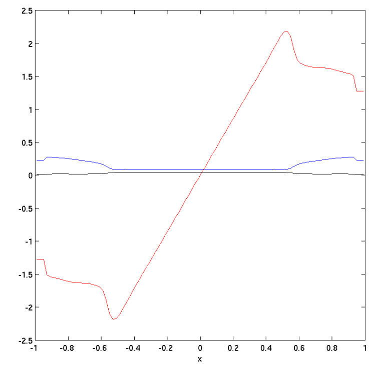


u

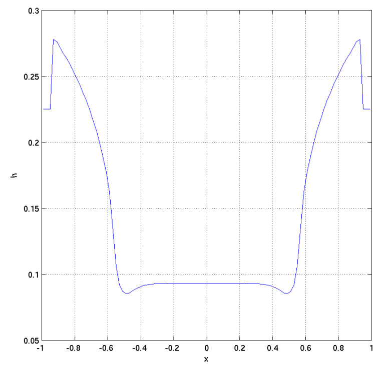


v

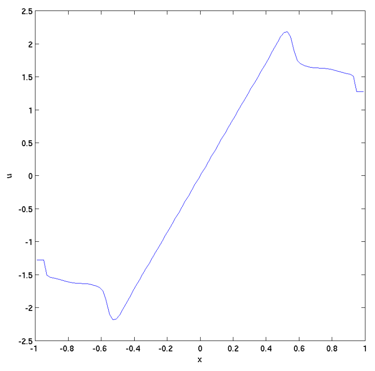
Figure 5.5: Test case 1, accuracy of FVEG scheme, contour plots, 100×100 cells.



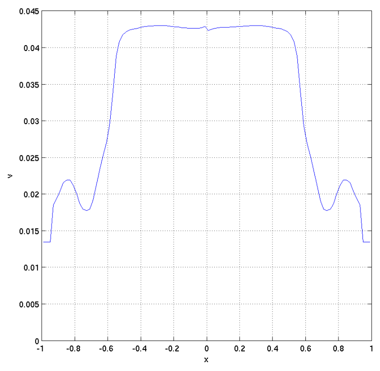
$h,\ u,\ v$



h

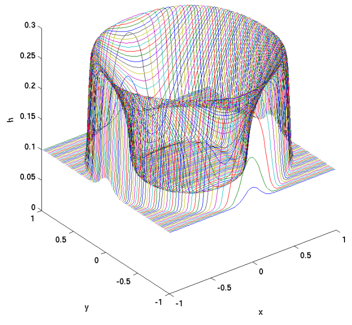


u

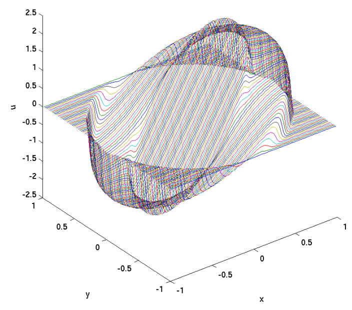


v

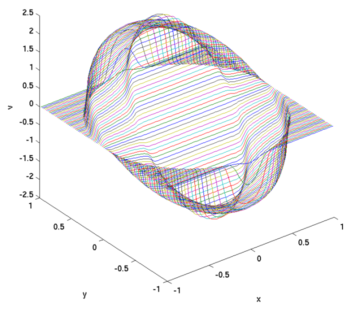
Figure 5.6: Test case 1, accuracy of FVEG scheme, plots at $y = 0$, 100×100 cells.



h

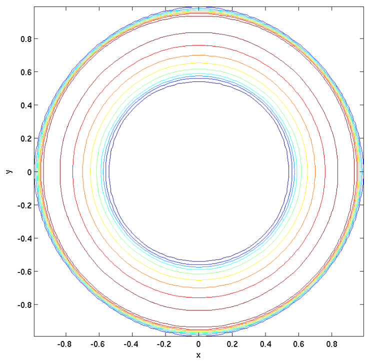


u

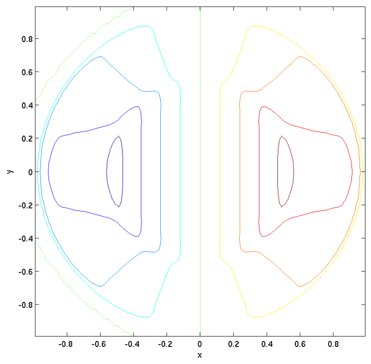


v

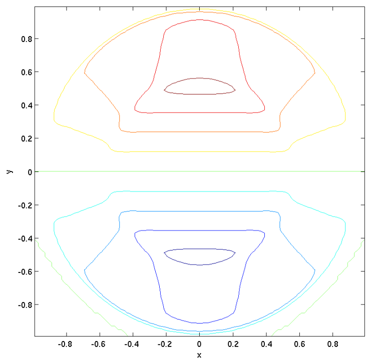
Figure 5.7: Test case 1, accuracy of FV scheme, 100×100 cells.



h

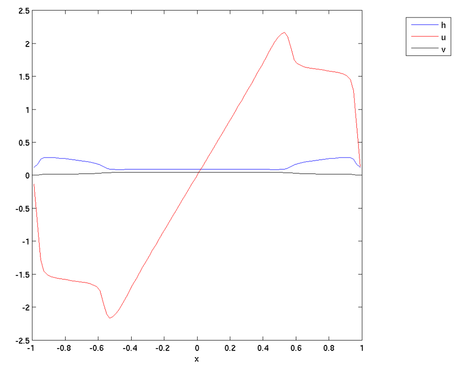


u

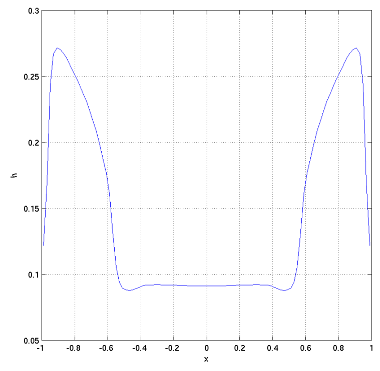


v

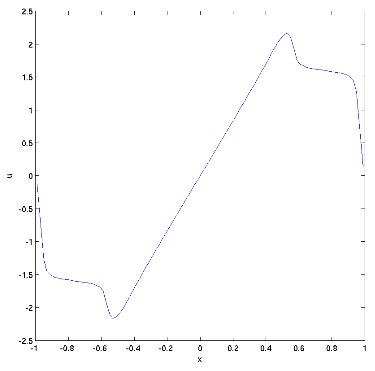
Figure 5.8: Test case 1, accuracy of FV scheme, contour plots, 100×100 cells.



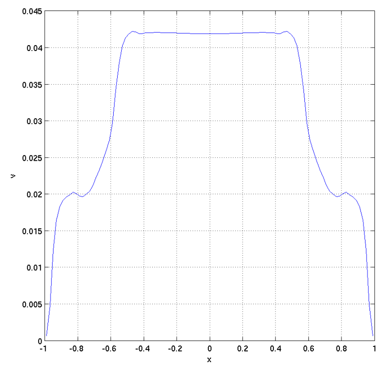
h, u, v



h

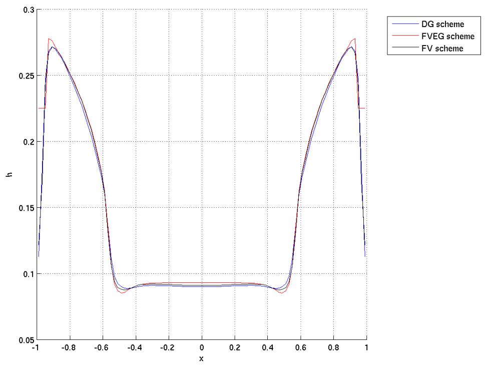


u

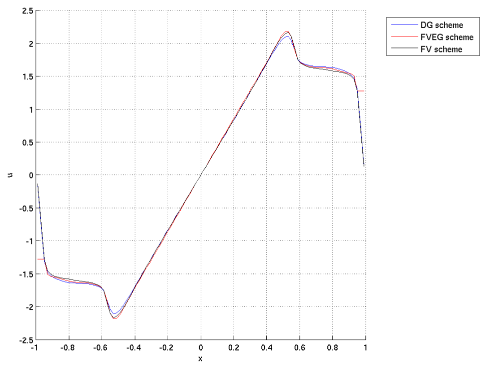


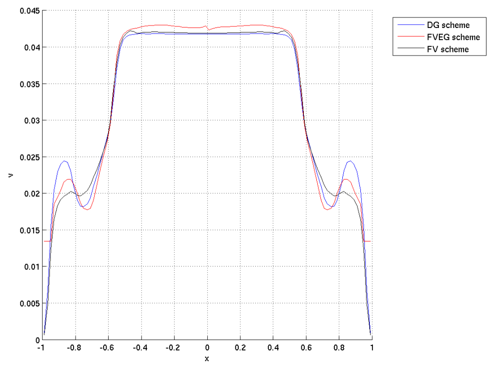
v

Figure 5.9: Test case 1, accuracy of FV scheme, plots at $y = 0$, 100×100 cells.



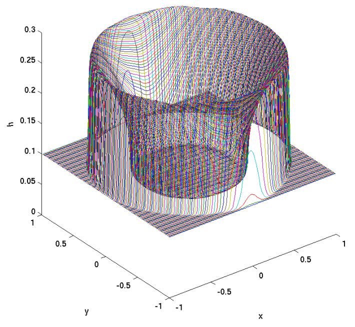
h



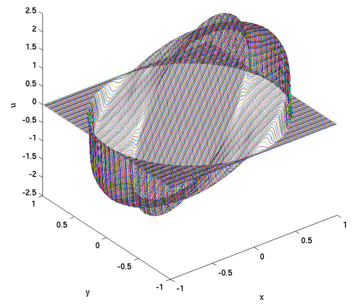


u v

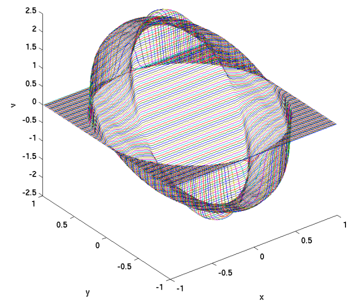
Figure 5.10: Test case 1, accuracy of DG, FVEG and FV scheme, plots at $y = 0$, 100×100 cells.



h

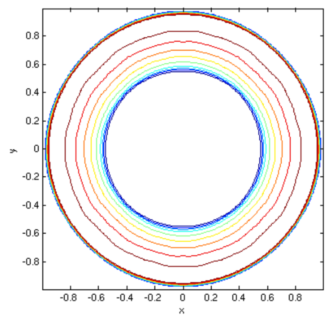


u

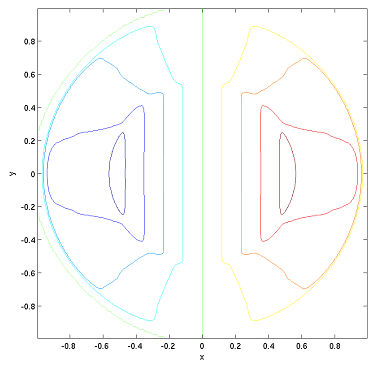


v

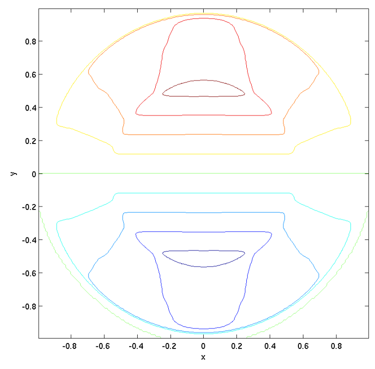
Figure 5.11: Test case 1, accuracy of the DG scheme,200 $\times$ 200 cells.



h

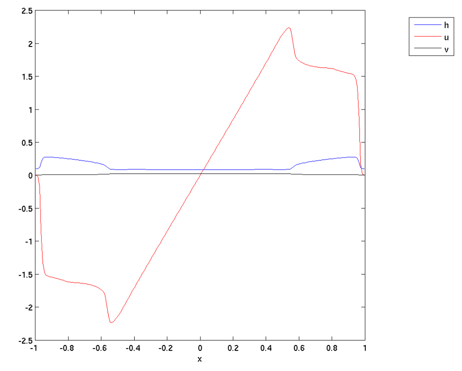


u

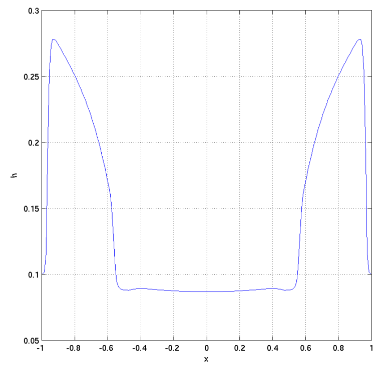


v

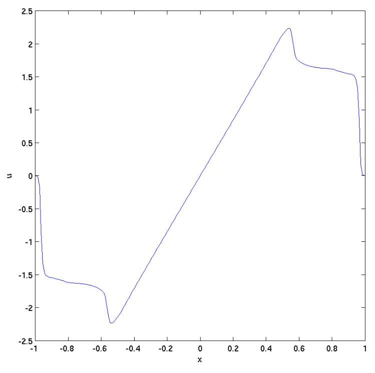
Figure 5.12: Test case 1, accuracy of the DG scheme, contour plots, 200×200 cells.



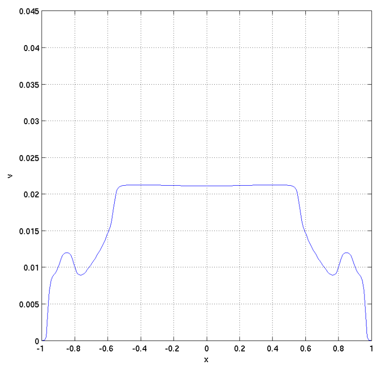
$h,\ u,\ v$



h

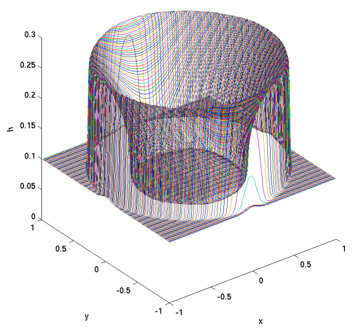


u

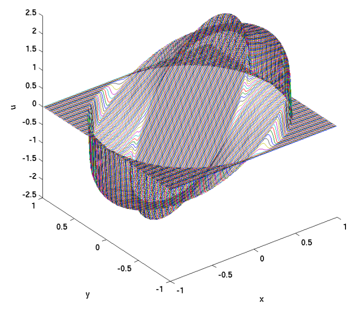


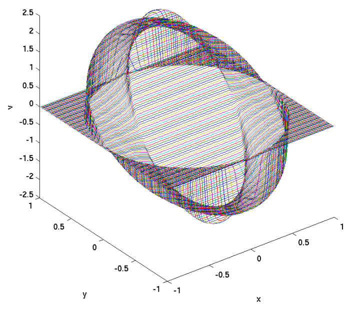
v

Figure 5.13: Test case 1, accuracy of the DG scheme, plots at $y = 0$, 200×200 cells.



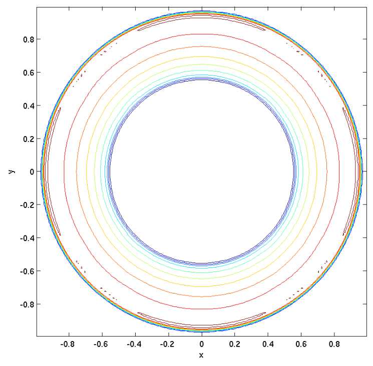
h



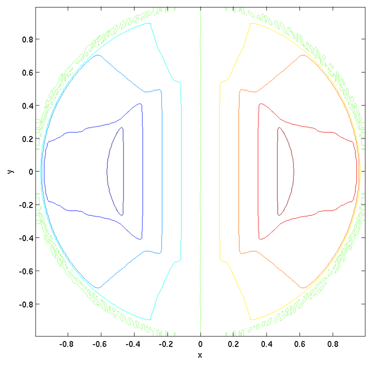


u v

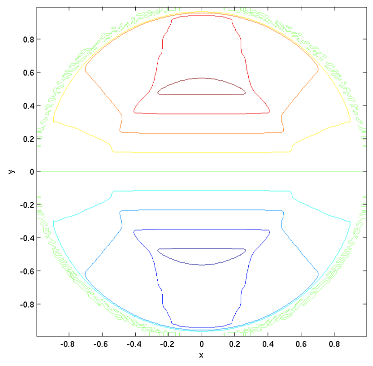
Figure 5.14: Test case 1, accuracy of the FVEG scheme, 200×200 cells.



h

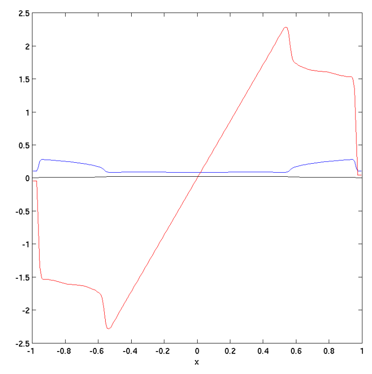


u

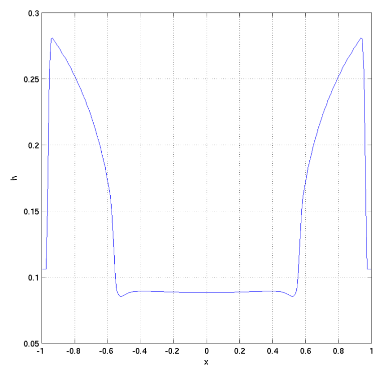


v

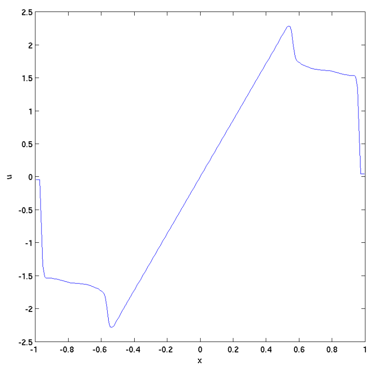
Figure 5.15: Test case 1, accuracy of the FVEG scheme, contour plots, 200×200 cells.



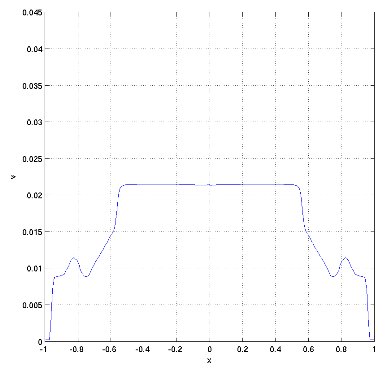
h, u, v



h

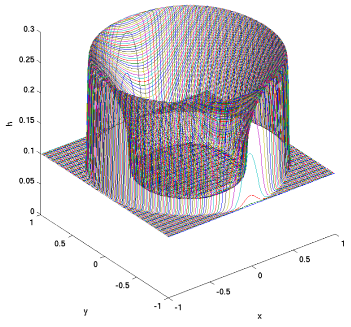


u

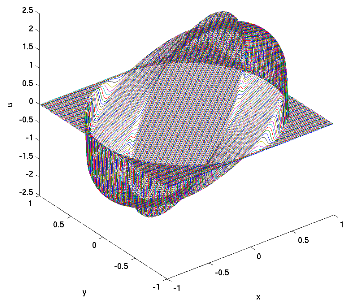


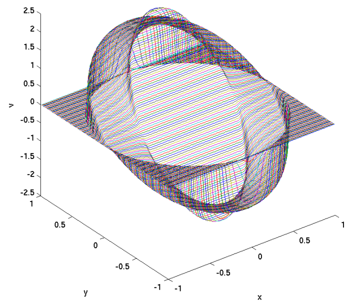
v

Figure 5.16: Test case 1, accuracy of the FVEG scheme, plots at $y = 0$, 200×200 cells.



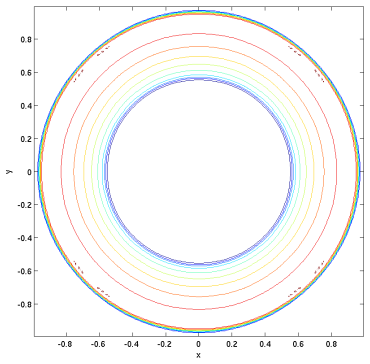
h



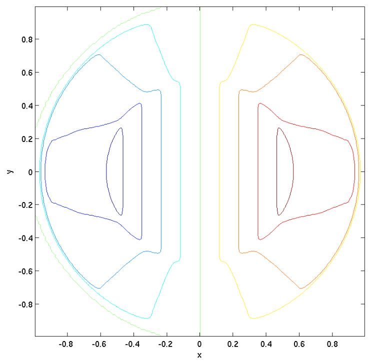


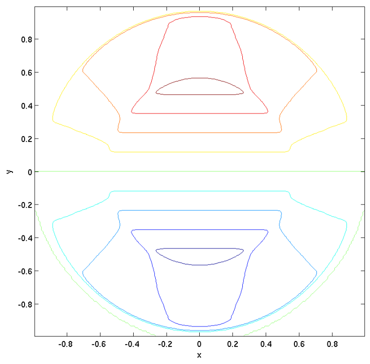
u v

Figure 5.17: Test case 1, accuracy of the FV scheme, 200×200 cells.



h





u v

Figure 5.18: Test case 1, accuracy of the FV scheme, contour plots, 200×200 cells.

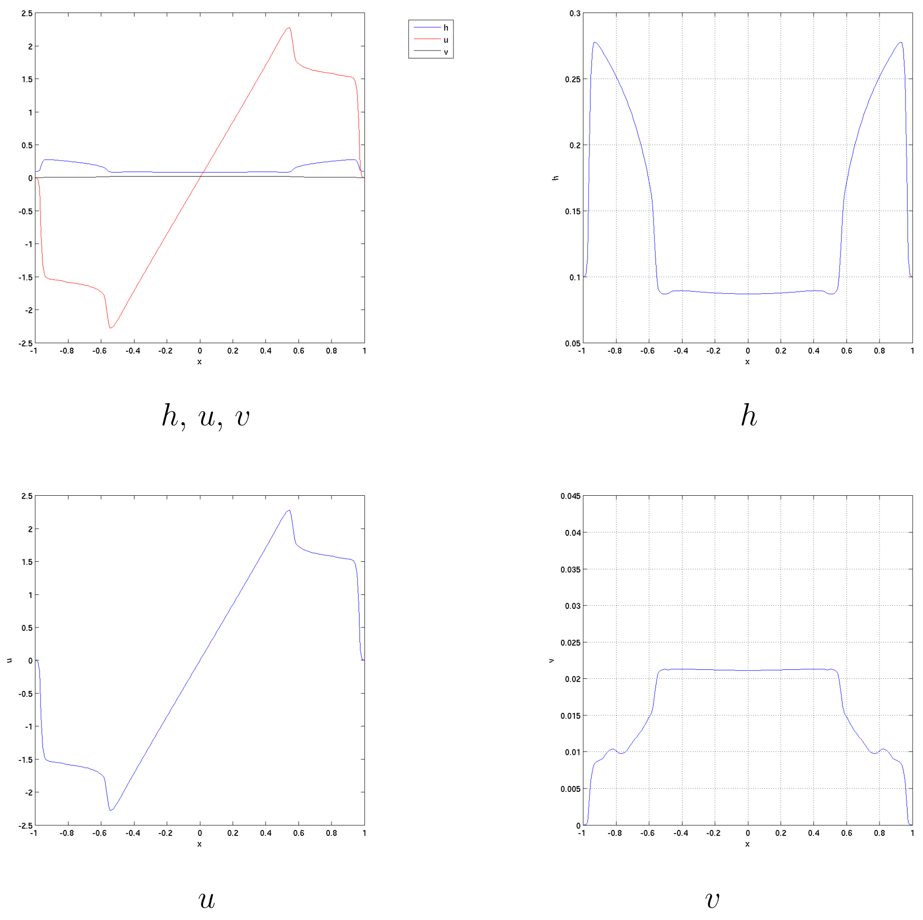
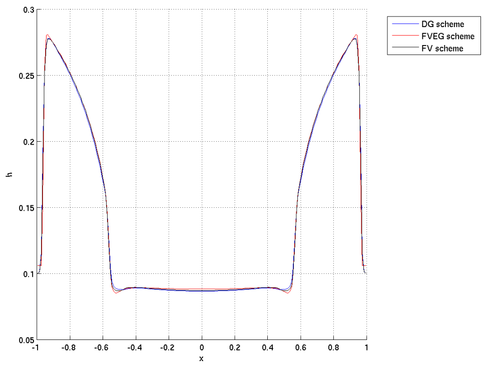
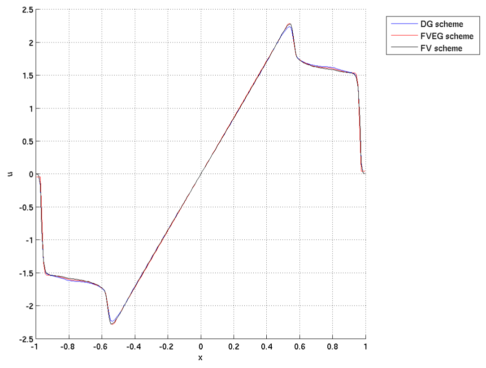
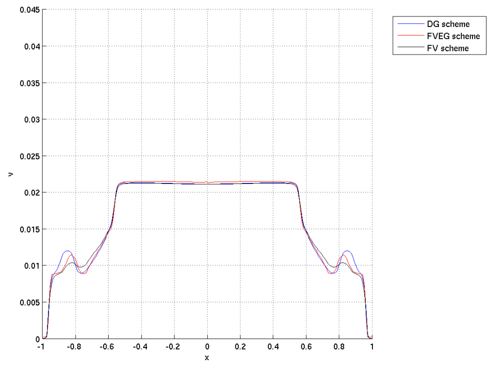


Figure 5.19: Test case 1, accuracy of the FV scheme, plots at $y = 0$, 200×200 cells.



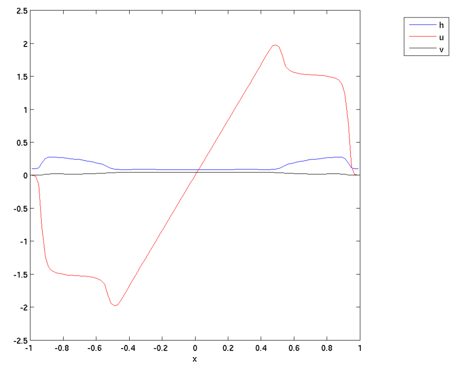
h



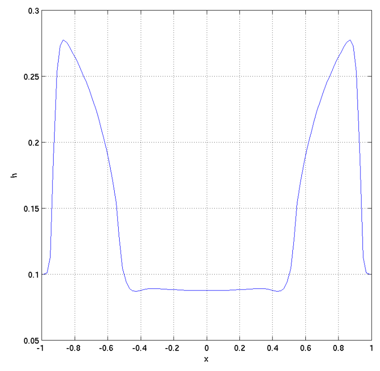


u v

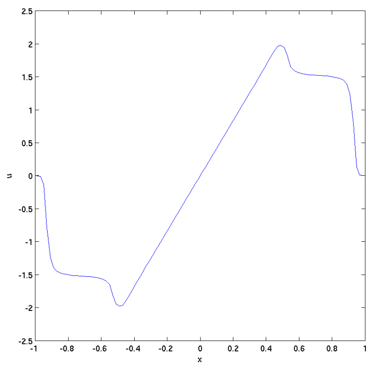
Figure 5.20: Test case 1, accuracy of DG, FVEG and FV scheme, plots at $y = 0$, 200×200 cells.



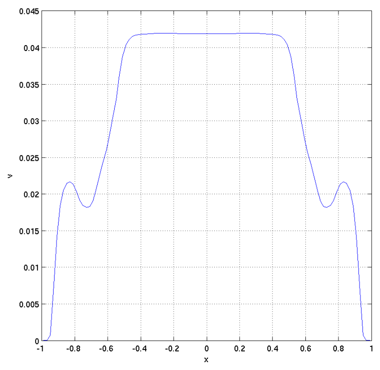
h, u, v



h

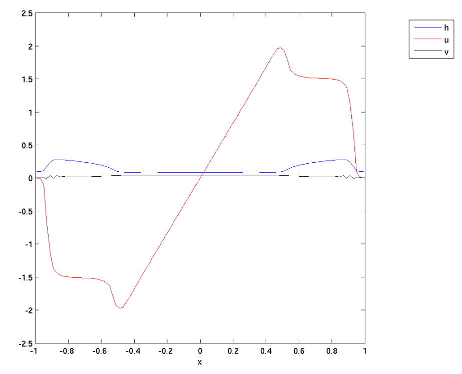


u

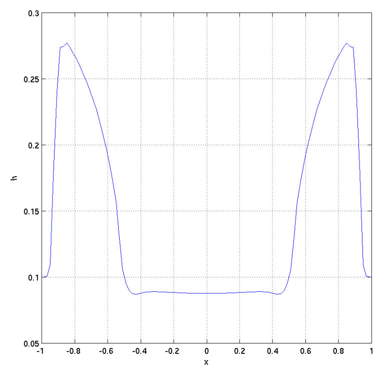


v

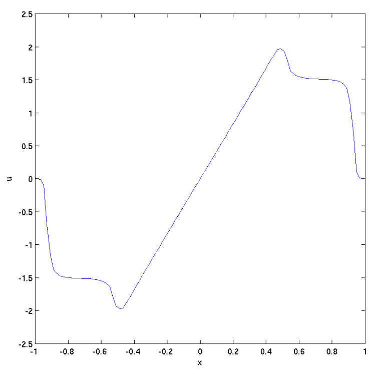
Figure 5.21: Test case 2, stability of the DG scheme, CFL = 0.8, plots at $y = 0$.



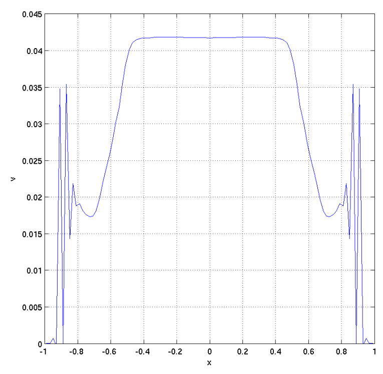
h, u, v



h

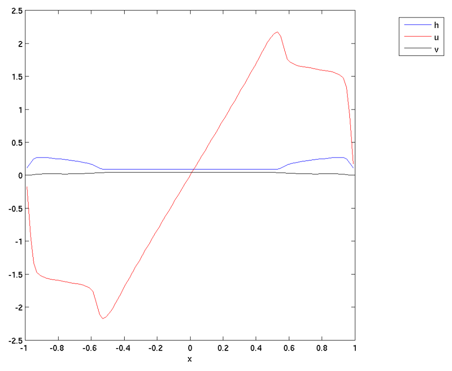


u

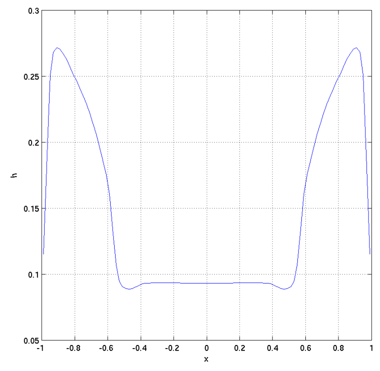


v

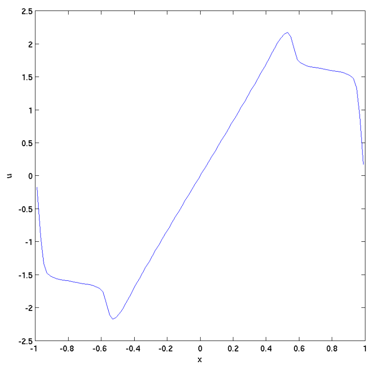
Figure 5.22: Test case 2, stability of the DG scheme, CFL = 0.9, plots at $y = 0$.



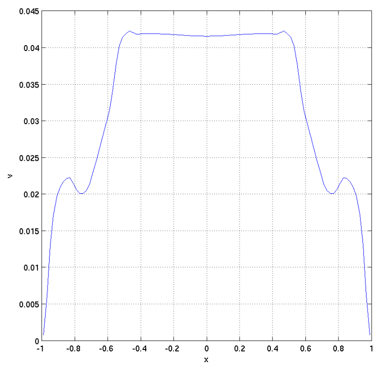
h, u, v



h

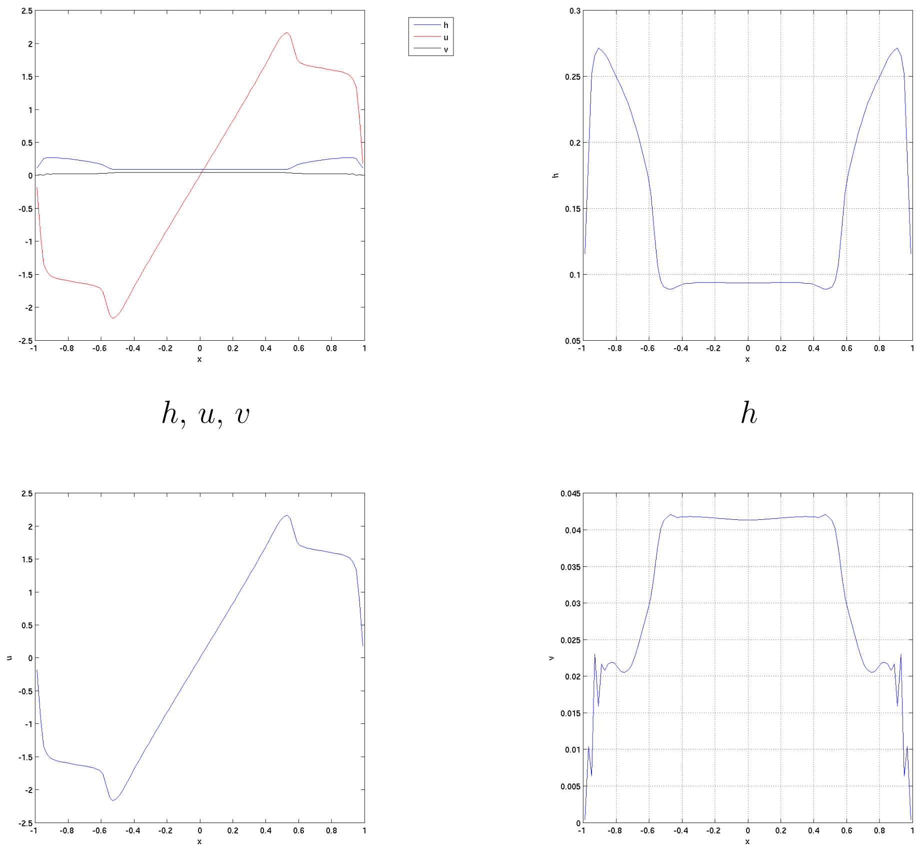


u



v

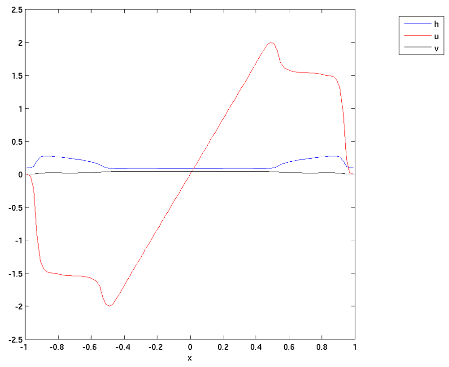
Figure 5.23: Test case 2, stability of the FV scheme, CFL = 0.6, plots at $y = 0$.



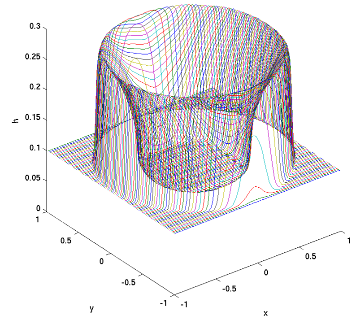
h, u, v h

u v

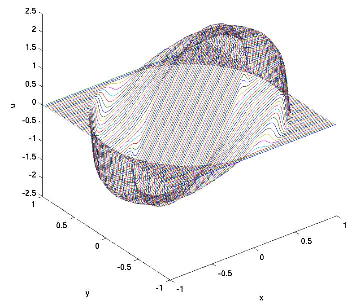
Figure 5.24: Test case 2, stability of the FV scheme, CFL $= 0.7$, plots at $y = 0$.



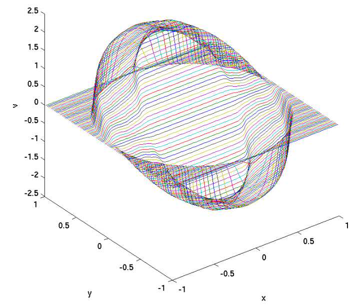
h, u, v at $y = 0$



h

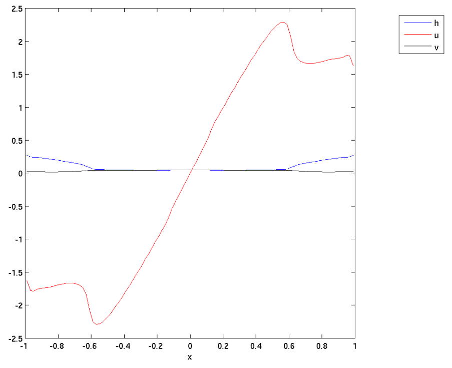


u

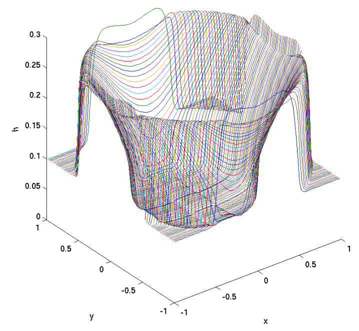


v

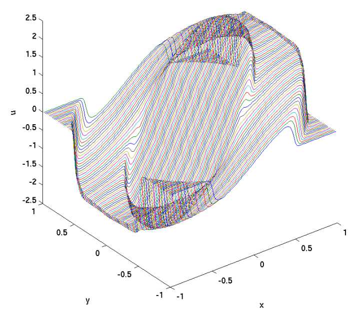
Figure 5.25: Test case 3, DG scheme for different numerical fluxes, Van Leer, domain size $[-1, 1] \times [-1, 1]$.



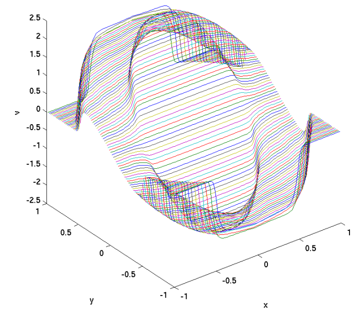
h, u, v at $y = 0$



h

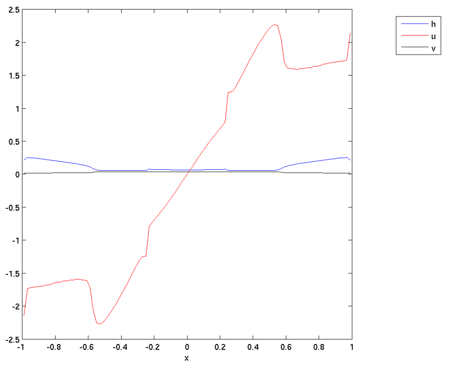


u

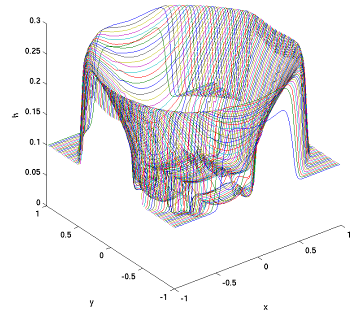


v

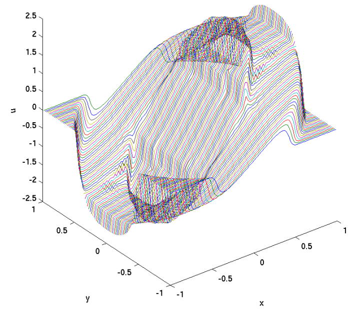
Figure 5.26: Test case 3, DG scheme for different numerical fluxes, Vijayasundaram, domain size $[-1, 1] \times [-1, 1]$.



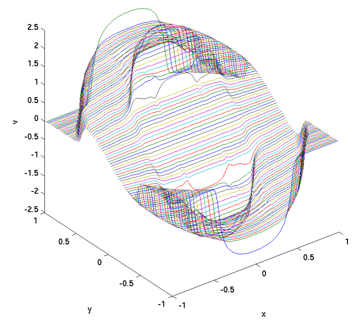
h, u, v at $y = 0$



h



u



v

Figure 5.27: Test case 3, DG scheme for different numerical fluxes, Steger-Warming, domain size $[-1, 1] \times [-1, 1]$.

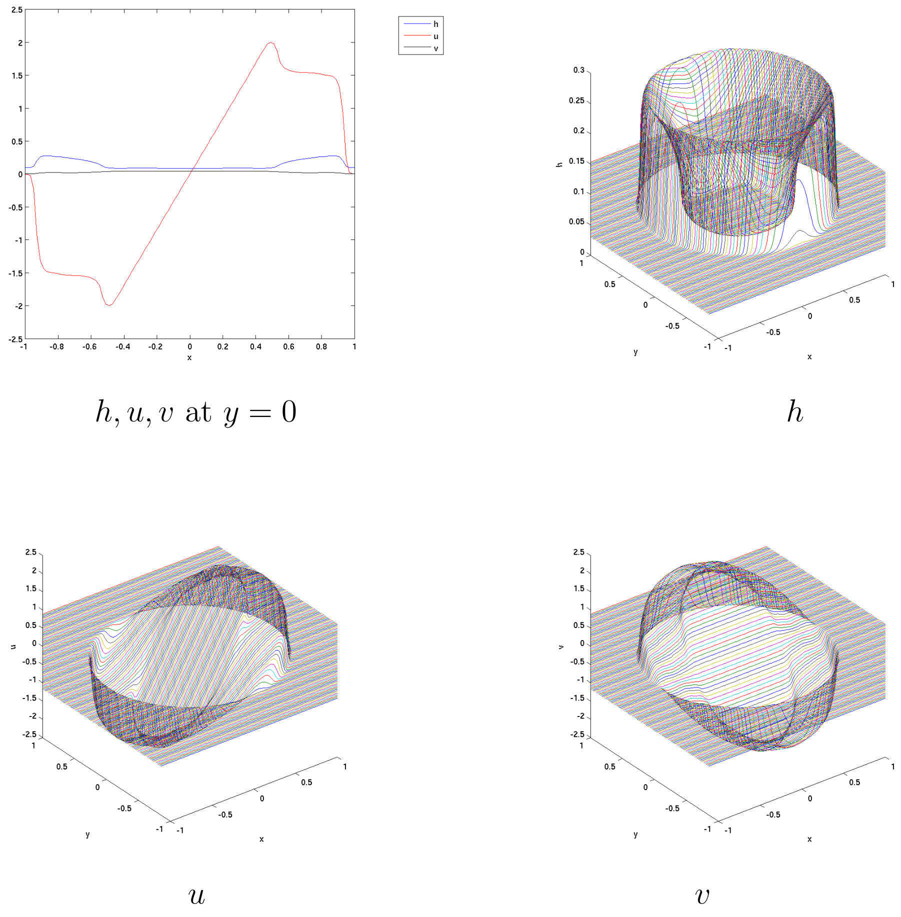


Figure 5.28: Test case 3, DG scheme for different numerical fluxes, Van Leer, domain size $[-1.5, 1.5] \times [-1.5, 1.5]$.

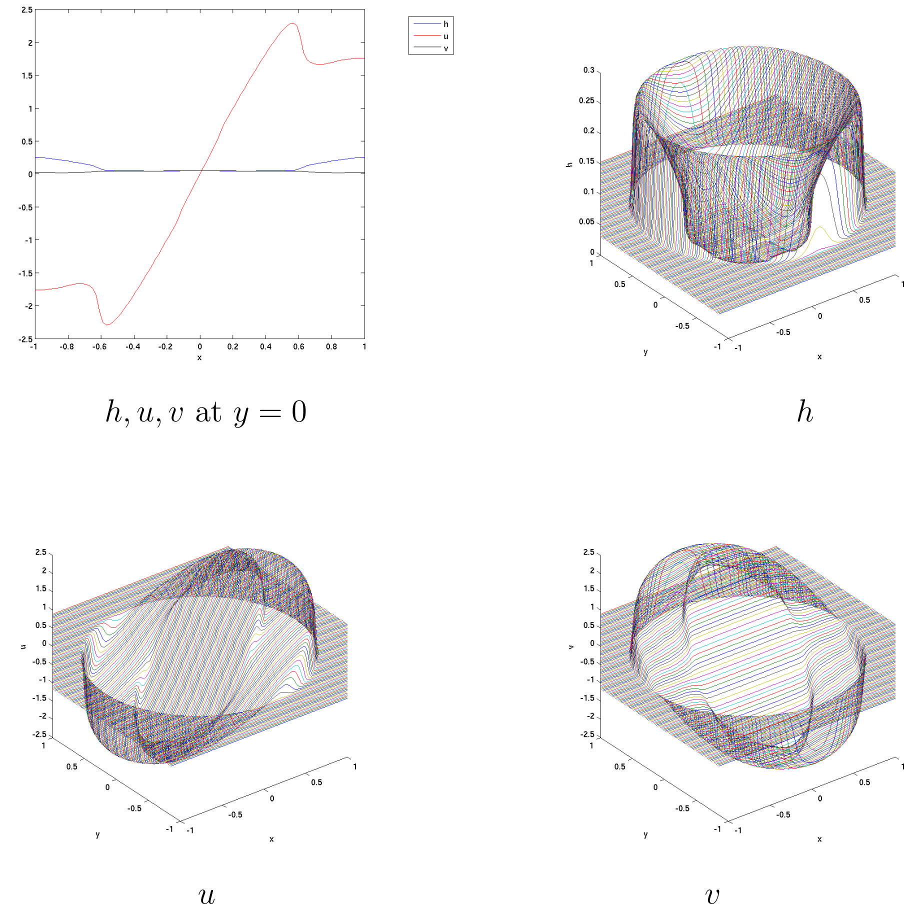
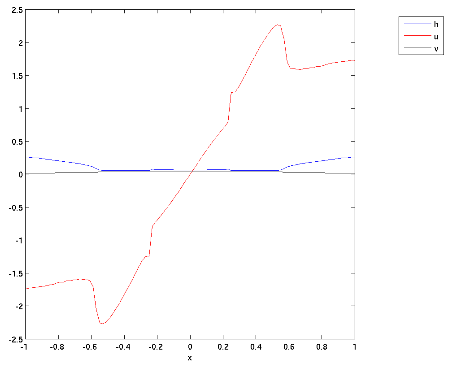
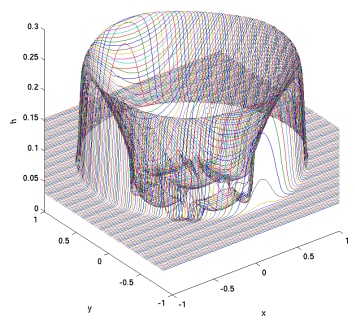


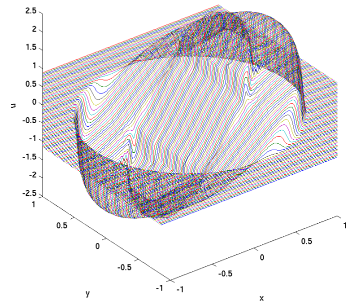
Figure 5.29: Test case 3, DG scheme for different numerical fluxes, Vijayasundaram, domain size $[-1.5, 1.5] \times [-1.5, 1.5]$.



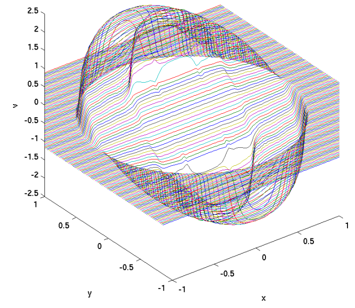
h, u, v at $y = 0$



h

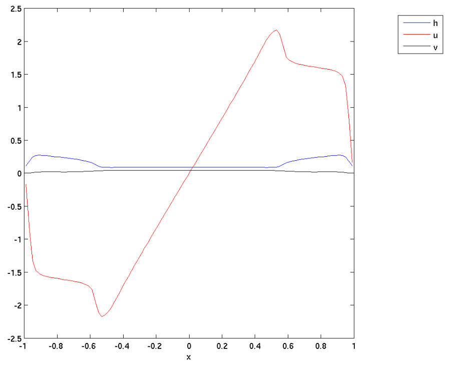


u

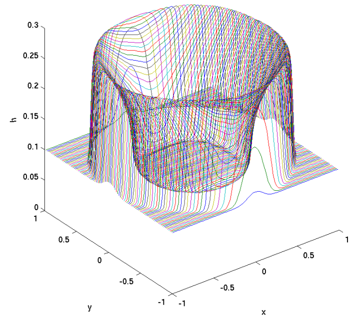


v

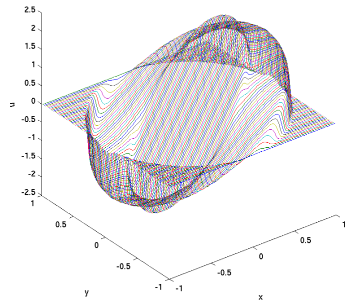
Figure 5.30: Test case 3, DG scheme for different numerical fluxes, Steger-Warming, domain size $[-1.5, 1.5] \times [-1.5, 1.5]$.



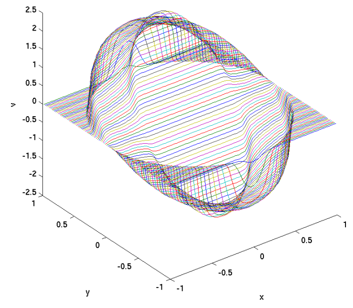
h, u, v at $y = 0$



h



u



v

Figure 5.31: Test case 3, ShallowWater/FV scheme for different numerical fluxes, Van Leer, domain size $[-1, 1] \times [-1, 1]$.

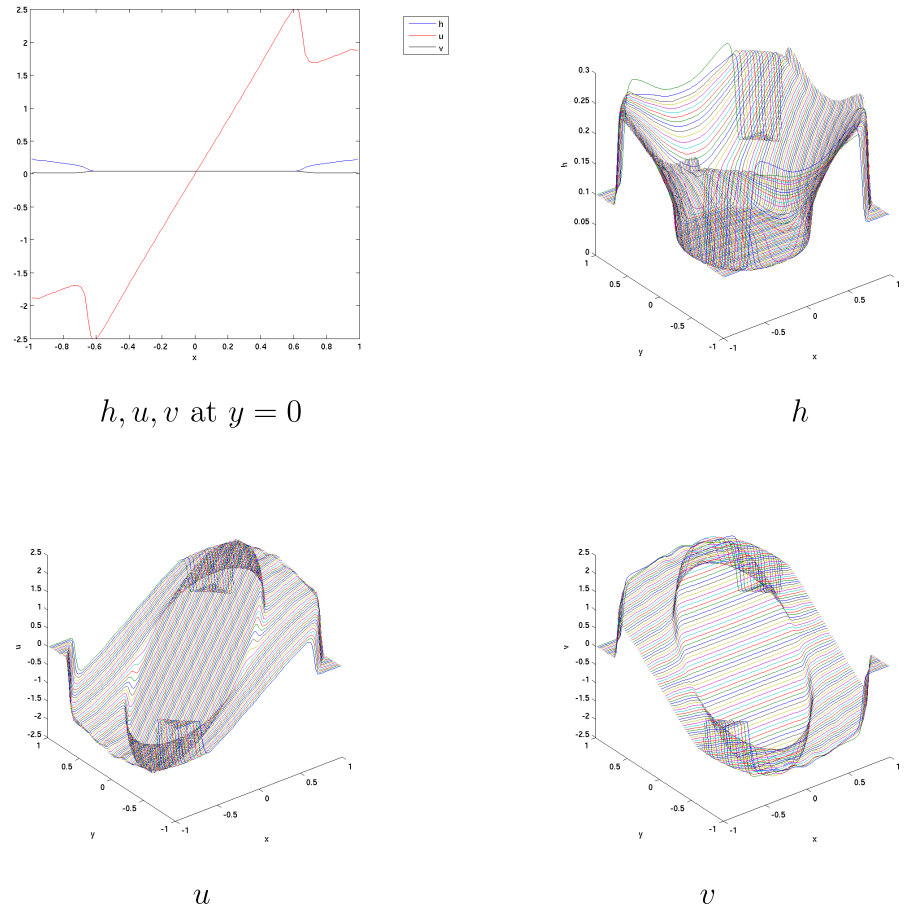
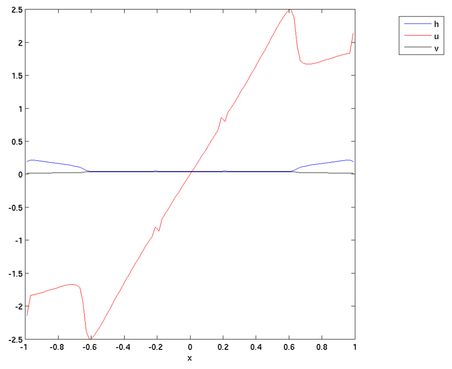
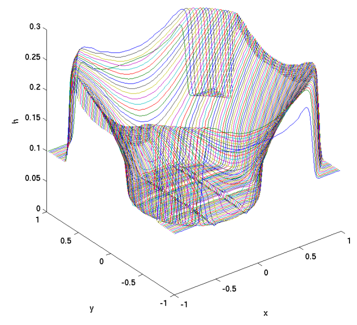


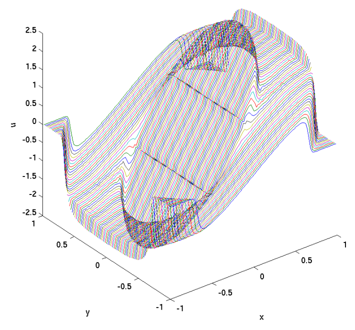
Figure 5.32: Test case 3, ShallowWater/FV scheme for different numerical fluxes, Vijayasundaram, domain size $[-1, 1] \times [-1, 1]$.



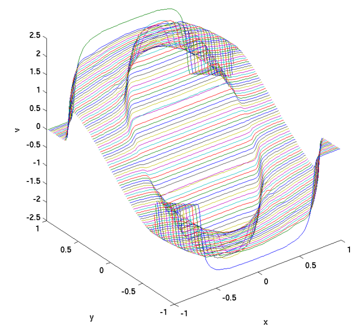
h, u, v at $y = 0$



h

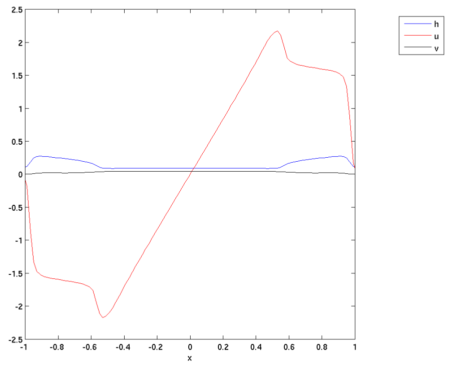


u

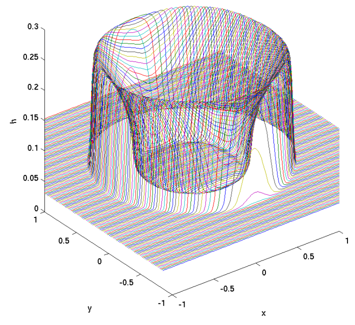


v

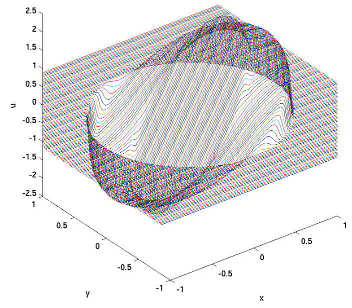
Figure 5.33: Test case 3, ShallowWater/FV scheme for different numerical fluxes, Steger-Warming, domain size $[-1, 1] \times [-1, 1]$.



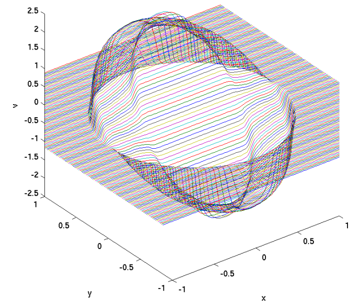
h, u, v at $y = 0$



h

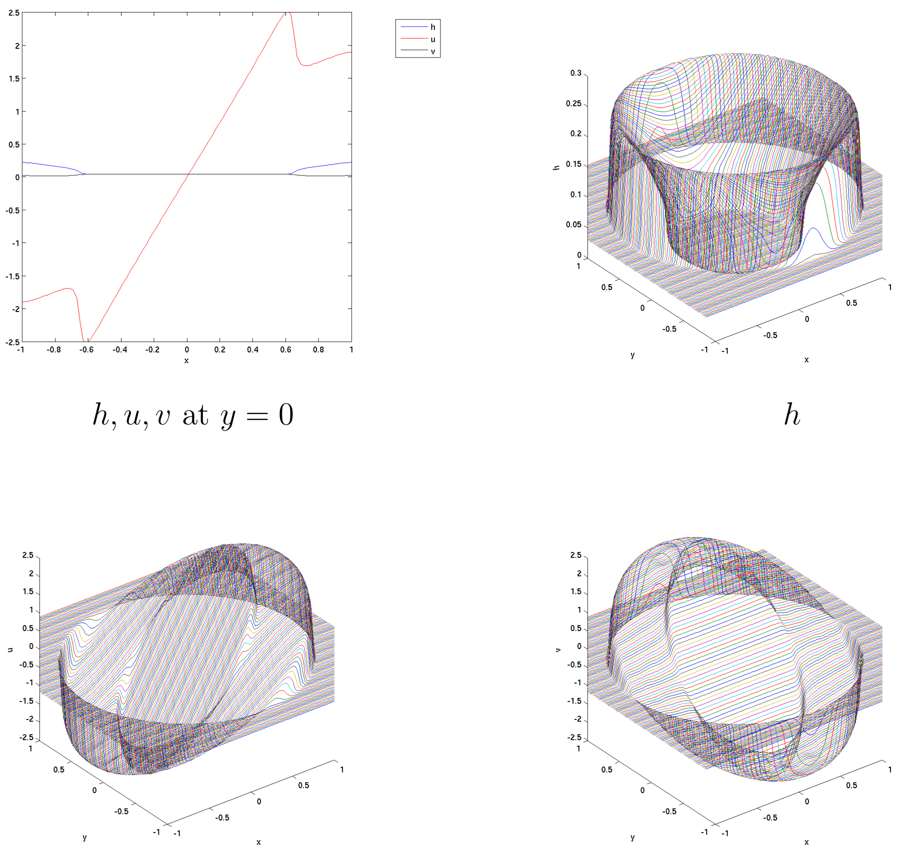


u



v

Figure 5.34: Test case 3, ShallowWater/FV scheme for different numerical fluxes, Van Leer, domain size $[-1.5, 1.5] \times [-1.5, 1.5]$.



h, u, v at $y = 0$ h

u v

Figure 5.35: Test case 3, ShallowWater/FV scheme for different numerical fluxes, Vijaya-sundaram, domain size $[-1.5, 1.5] \times [-1.5, 1.5]$.

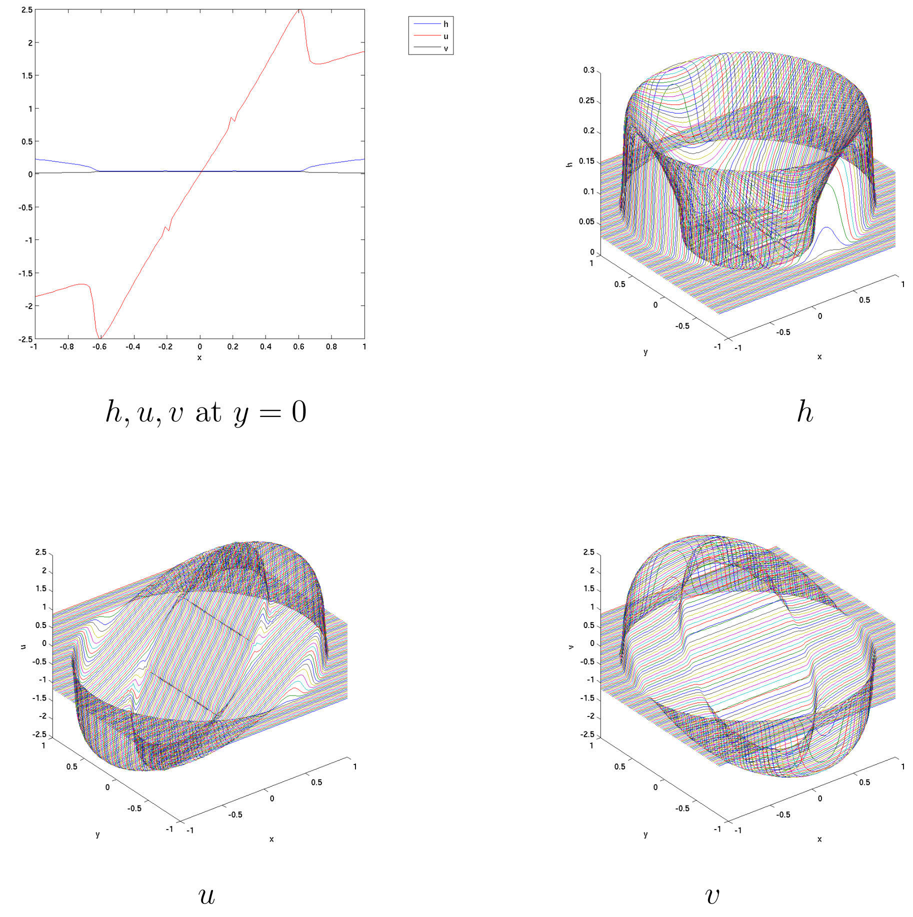


Figure 5.36: Test case 3, ShallowWater/FV scheme for different numerical fluxes, Steger-Warming, domain size $[-1.5, 1.5] \times [-1.5, 1.5]$.

$N \times N$ cells	$\left\lVert U^n_{N/2} - U^n_N \right\rVert_{L^2}$	EOC_{L^2}	$\left\lVert U^n_{N/2} - U^n_N \right\rVert_{L^1}$	EOC_{L^1}
40×40	0.008236		0.020011	
80×80	0.002323	1.825955	0.005512	1.860145
160×160	0.000608	1.933846	0.001428	1.948580
320×320	0.000154	1.981141	0.000362	1.979934

EOCs for $U = (h, hu, hv)^T$

$N \times N$ cells	$\left\lVert h^n_{N/2} - h^n_N \right\rVert_{L^2}$	EOC_{L^2}	$\left\lVert h^n_{N/2} - h^n_N \right\rVert_{L^1}$	EOC_{L^1}
40×40	0.002460		0.003842	
80×80	0.000670	1.876425	0.001042	1.882502
160×160	0.000174	1.945074	0.000270	1.948324
320×320	0.000044	1.983512	0.000069	1.968291

EOCs for h

$N \times N$ cells	$\left\lVert u^n_{N/2} - u^n_N \right\rVert_{L^2}$	EOC_{L^2}	$\left\lVert u^n_{N/2} - u^n_N \right\rVert_{L^1}$	EOC_{L^1}
40×40	0.014166		0.023777	
80×80	0.004089	1.792612	0.006688	1.829920
160×160	0.001105	1.887702	0.001776	1.912943
320×320	0.000287	1.944924	0.000458	1.955212

EOCs for u

Table 5.1: Test case T4, convergence of DG scheme, CFL = 0.4

$N \times N$ cells	$\left\|U^n_{N/2} - U^n_N\right\|_{L^2}$	EOC_{L^2}	$\left\|U^n_{N/2} - U^n_N\right\|_{L^1}$	EOC_{L^1}
40×40	0.001305		0.003137	
80×80	0.000259	2.331123	0.000637	2.299331
160×160	0.000060	2.121213	0.000147	2.119698
320×320	0.000015	2.029765	0.000036	2.042593

EOCs for $U = (h, hu, hv)^T$

$N \times N$ cells	$\left\|h^n_{N/2} - h^n_N\right\|_{L^2}$	EOC_{L^2}	$\left\|h^n_{N/2} - h^n_N\right\|_{L^1}$	EOC_{L^1}
40×40	0.000375		0.000547	
80×80	0.000068	2.469341	0.000105	2.381071
160×160	0.000014	2.306277	0.000022	2.264561
320×320	0.000003	2.136968	0.000005	2.134424

EOCs for h

$N \times N$ cells	$\left\|u^n_{N/2} - u^n_N\right\|_{L^2}$	EOC_{L^2}	$\left\|u^n_{N/2} - u^n_N\right\|_{L^1}$	EOC_{L^1}
40×40	0.002774		0.004200	
80×80	0.000670	2.048535	0.001037	2.017743
160×160	0.000164	2.033718	0.000256	2.017437
320×320	0.000040	2.021053	0.000063	2.012732

EOCs for u

Table 5.2: Test case T4, convergence of FVEG scheme, CFL = 0.4.

$N \times N$ cells	$\left\lVert U_{N/2}^n - U_N^n \right\rVert_{L^2}$	$\frac{\left\lVert U_{N/2}^n - U_N^n \right\rVert_{L^2}}{\left\lVert U_N^n - U_{2N}^n \right\rVert_{L^2}}$	$\left\lVert U_{N/2}^n - U_N^n \right\rVert_{L^1}$	$\frac{\left\lVert U_{N/2}^n - U_N^n \right\rVert_{L^1}}{\left\lVert U_N^n - U_{2N}^n \right\rVert_{L^1}}$
40×40	0.004378		0.009941	
80×80	0.001101	1.991457	0.002503	1.989733
160×160	0.000285	1.949781	0.000649	1.947368
320×320	0.000073	1.964994	0.000166	1.967035

EOCs for $U = (h, hu, hv)^T$

$N \times N$ cells	$\left\lVert h_{N/2}^n - h_N^n \right\rVert_{L^2}$	$\frac{\left\lVert h_{N/2}^n - h_N^n \right\rVert_{L^2}}{\left\lVert h_N^n - h_{2N}^n \right\rVert_{L^2}}$	$\left\lVert h_{N/2}^n - h_N^n \right\rVert_{L^1}$	$\frac{\left\lVert h_{N/2}^n - h_N^n \right\rVert_{L^1}}{\left\lVert h_N^n - h_{2N}^n \right\rVert_{L^1}}$
40×40	0.001187		0.001736	
80×80	0.000296	2.003651	0.000432	2.006664
160×160	0.000077	1.942667	0.000113	1.934709
320×320	0.000020	1.944858	0.000029	1.962198

EOCs for h

$N \times N$ cells	$\left\lVert u_{N/2}^n - u_N^n \right\rVert_{L^2}$	$\frac{\left\lVert u_{N/2}^n - u_N^n \right\rVert_{L^2}}{\left\lVert u_N^n - u_{2N}^n \right\rVert_{L^2}}$	$\left\lVert u_{N/2}^n - u_N^n \right\rVert_{L^1}$	$\frac{\left\lVert u_{N/2}^n - u_N^n \right\rVert_{L^1}}{\left\lVert u_N^n - u_{2N}^n \right\rVert_{L^1}}$
40×40	0.006433		0.009468	
80×80	0.001528	2.073847	0.002314	2.032671
160×160	0.000388	1.977516	0.000595	1.959427
320×320	0.000099	1.970556	0.000152	1.968818

EOCs for u

Table 5.3: Test case T4, convergence of FV scheme, CFL = 0.4.

Conclusions

The aim of this work was a comparison of the discontinuous Galerkin method with the traditional finite volume methods as well as with the genuinely multidimensional finite volume evolution Galerkin method, when applied to hyperbolic problems.
For the DG scheme, the construction of linear basis functions on a triangular grid proposed by Feistauer [2] formed the basis of our implementation. Apart from the implementation of this approach on a regular triangular grid, we wanted to consider the DG scheme on a rectangular grid, in order to make a better comparison with the FVEG scheme possible. An implementation of the FVEG scheme on a rectangular grid was given. Therefore, proceeding in analogy to the triangular approach, we constructed a DG scheme with piecewise bilinear basis functions on a rectangular grid. This is our original contribution. Now, the choice of quadrature rules for the volume and surface integrals needed special attention. In correspondence to the convergence results derived by Cockburn et al. [5], approximations differing from those proposed by Feistauer [2] for the triangular grid had to be used. For the discretization of the numerical fluxes several approaches have been implemented in order to examine the influence of the numerical flux function on the properties of the DG scheme. Analogously, we have implemented the traditional FV scheme with different numerical flux functions on a rectangular grid.
Our extensive numerical results show the accuracy, stability and convergence of the different schemes, when applied to the Burgers equation and to the shallow water equation system. No large differences can be seen between the approximations obtained with the different numerical schemes. However, it should be pointed out that in comparison with the DG scheme, the FVEG scheme is sharper in the resolution of multidimensional effects. The global error that we obtained when applying the DG scheme to the shallow water equation system, suggests that the DG scheme is not equal in accuracy to the finite volume schemes, either. The implementation of the DG scheme with approximate mass matrix is superior in stability to the finite volume schemes. However, for the other implementations of the DG scheme, stability is only obtained by strongly limiting the time step. Here a different choice of the time stepping method might help.
An examination of the stability of the differently implemented DG schemes on the basis of the von Neumann analysis might be an interesting objective of future study. Further comparisons between the DG scheme and the FV schemes might be of interest, too, among which could be named a comparison of the CPU time. Here, the traditional FV schemes have the advantage, that only the surface integrals have to be computed at each time level,

whereas the DG update requires the computation of the volume integrals as well as of the surface integrals. For a better examination of the influence of the different numerical flux functions on the properties of the DG scheme, hyperbolic problems which do fulfill the homogeneity condition (1.27) – such as e.g. the Euler equations – should be considered.

Bibliography

[1] M. Feistauer, *Mathematical Methods in Fluid Dynamics*, Longman Scientific and Technical, 1993.

[2] M. Feistauer, J. Felcman, I. Straškraba, *Mathematical and Computational Methods for Compressible Flow*, Clarendon press Oxford, 2003.

[3] R. J. LeVeque, *Finite Volume Methods for Hyperbolic Problems.* Cambridge University Press, 2002.

[4] R. J. LeVeque, *Numerical Methods for Conservation Laws.* Birkhäuser Verlag, 1992.

[5] B. Cockburn, G. E. Karniadakis, C.-W. Shu (Eds.), *Discontinuous Galerkin Methods.* Springer-Verlag, 2000.

[6] M. Lukáčová-Medvid'ová, K. W. Morton, and G. Warnecke, *Evolution Galerkin Methods for Hyperbolic Systems in Two Space Dimensions.* MathCom. 69(232), 2000, 1355-1384.

[7] M. Lukáčová-Medvid'ová, K. W. Morton, and G. Warnecke, *Finite Volume Evolution Galerkin Methods for Hyperbolic Systems.* SIAM J. Sci. Comp. 26(1), 2004, 1-30.

[8] M. Lukáčová-Medvid'ová, J. Saibertóva, and G. Warnecke, *Finite volume evolution Galerkin methods for nonlinear hyperbolic systems.* J. Comp. Phys. 183, 2002, 533-562.

[9] M. Lukáčová-Medvid'ová, G. Warnecke and Y. Zahaykah, *On stability of the evolution Galerkin schemes applied to a two-dimensional wave equation system*, accepted SIAM J. Num: Anal., 2006.

[10] Qurrat-ul-Ain, *Multidimensional Schemes for Hyperbolic Conservation Laws on Triangular Meshes*, Dissertation, University of Magdeburg, 2005.