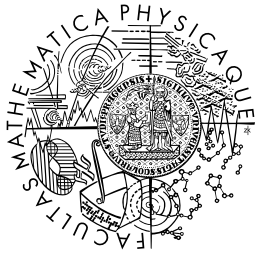


FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

MASTER THESIS

Lenka Stará

The Gibbs phenomenon in the discontinuous Galerkin method

Department of Numerical Mathematics

Supervisor of the master thesis:	doc. RNDr. Václav Kučera, Ph.D.
Study programme:	Mathematics
Study branch:	Numerical and Computational Mathematics

Prague 2018

I declare that I carried out this master thesis independently, and only with the cited sources, literature and other professional sources.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

In ..... date ..... signature of the author

Title: The Gibbs phenomenon in the discontinuous Galerkin method

Author: Lenka Stará

Department: Department of Numerical Mathematics

Supervisor: doc. RNDr. Václav Kučera, Ph.D., Department of Numerical Mathematics

Abstract: The solution of the Burgers' equation computed by the standard finite element method is degraded by oscillations, which are the manifestation of the Gibbs phenomenon. In this work we study the following numerical methods: Discontinuous Galerkin method, stable low order schemes and the flux corrected technique method in order to prevent the undesired Gibbs phenomenon. The focus is on the reduction of severe overshoots and undershoots and the preservation of the smoothness of the solution. We consider a simple 1D problem on the interval $(0, 1)$ with different initial conditions to demonstrate the properties of the presented methods. The numerical results of individual methods are provided.

Keywords:
Burgers' equation
Discontinuous Galerkin method
Low order scheme
Flux corrected technique
Overshoots and undershoots
Limiting strategies

I would like to thank to doc. RNDr. Václav Kučera, Ph.D. without whose supervising this thesis would never be written. I'm really grateful for all these afternoons we have spend over discussing the difficulties I had to face. The words I hated the most in the past year was "Try to write everything from the scratch on the paper and compare the results you have obtained from your program." But now I appreciate it because it helped me to gain the confidence that I can solve the problems by myself. Thanks for encouraging me and helping me to last until the end.

Contents

Introduction

The Gibbs phenomenon is a phenomenon in which the Fourier series of a piecewise continuously differentiable periodic function exhibits oscillations in the regions, where the function has the jump discontinuity. The partial sum in the area close to these jump discontinuities may exhibit overshoots and undershoots, i.e. the n-th partial sum of the Fourier series might grow over the function maximum. The phenomenon is well illustrated in Figure 1 from Wolfram Web Resource [see Weisstein].

In the Finite element method (FEM) the Gibbs phenomenon stands for undesired phenomena, which exhibits spurious oscillations in the discrete solution whenever we solve a problem with steep gradients or discontinuities, see Figure 2 as an example. This is the major drawback of the "traditional" approach of solving the partial differential equation using FEM. Moreover, for the solution possessing discontinuities it is not reasonable to use a continuous approximation.

To avoid the Gibbs phenomenon, we use a different numerical scheme, the *discontinuous Galerkin* (DG) method. The method is similar to the FEM method that we also use piecewise polynomial functions in the discrete formulation. The major difference between FEM and DG methods is that in DG we do not have any assumptions on the continuity between neighboring elements. The idea behind the DG method is to use some kind of weaker continuity constraints realized by penalization terms. Hence DG is more suited for approximating discontinuous function. Another advantage of the DG scheme is that if the solution provides overshoots and undershoots on some intervals, there is only limited propagation of the "bad" behavior to the neighboring elements.

Essentially, by using the DG method instead of FEM we can improve the solution and avoid the Gibbs phenomenon to a large extent. The methods itself, however, does not prevent overshoots and undershoots in the areas close to discontinuities. There are different approaches how to improve the DG method, e.g. by adding the artificial viscosity [see Persson and Peraire, 2006] or by projection limiting [see Krivodonova, 2007]. In this work we present a rather nonstandard approach. To limit the magnitude of the overshoots and undershoots we use a method called *flux corrected transport* or *flux corrected technique* (FCT).

The FCT method was developed for stabilizing the FEM and becomes rather popular because of its simplicity. The idea is to manipulate with the system matrix of the FEM scheme using only algebraic operations to provide a system with better numerical properties. If the method is well implemented, it is efficient and the solution is considerably improved. For more details about FEM-FCT refer to Kuzmin and Möller [2005] or John and Schmeyer [2008].

Combining the DG method with FCT stabilization is a promising combination for eliminating Gibbs phenomenon and minimizing overshoots and undershoots. In this work we will study several strategies based on the combination of DG and FCT, especially the behavior of these methods used for solving Burgers' equation.

The work has the following structure. In Section 1 we provide the basic definitions and notation necessary for derivation of the numerical methods. In Section 2 we derive the DG formulation of Burgers' equation step by step, at the end of this chapter we present the high order scheme with system matrix

provided. In Section 3 we introduce the low order scheme and FCT stabilization and provide a summary of some theoretical results in this field, we also discuss several algorithms how to choose limiting parameters. In the last chapter we present numerical results and compare the methods treated in Section 3.

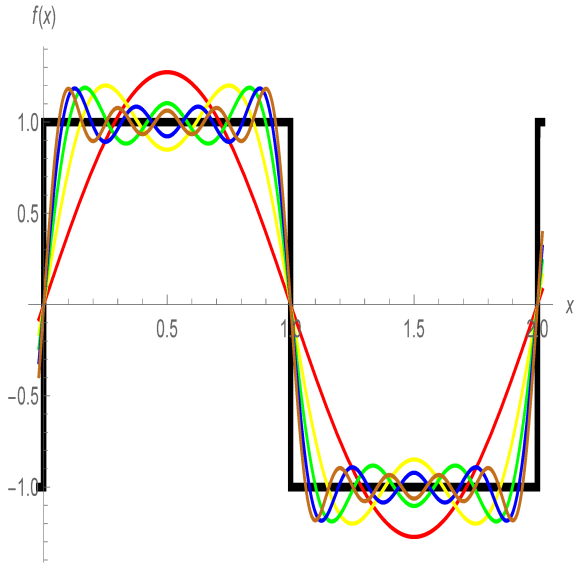


Figure 1: Gibbs phenomenon illustrated in the Fourier series of a square wave.

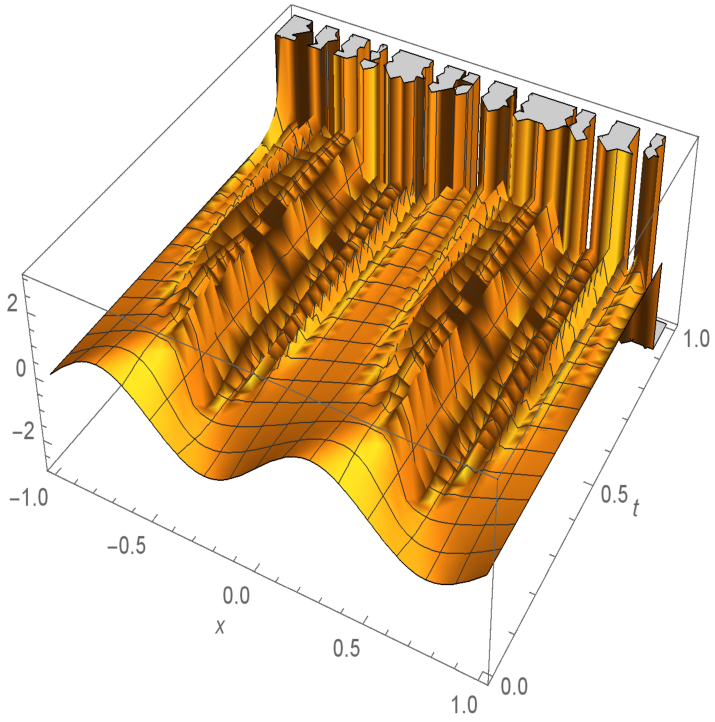


Figure 2: Gibbs phenomenon in numerical solution of Burgers' equation equipped with the initial condition $\sin(2\pi x)$ computed in software Mathematica [Wolfram Research, Inc.]. The code is available as Attachment A1.

1. Definitions and notations

Before we introduce the DG method, we need to introduce some definitions and useful notation.

Let $D_n = \{0 = x_0 < x_1 < x_2 < \cdots < x_{n-1} < x_n = 1\}$ be the set of distinct points. On these points we define the partition of the interval $[0, 1]$ to be a finite number of closed intervals I with disjoint interiors such that

$$[0, 1] = \bigcup_{I \in \mathcal{T}_h} I.$$

We will denote the partition by the symbol $\mathcal{T}_h$.

For each interval we set $h_I = x_{i+1} - x_i$ then the *norm of the partition* is given by

$$h = \max_{I \in \mathcal{T}_h} h_I.$$

Parameter h gives us the information about roughness of the division.

Let $I, J \in \mathcal{T}_h$, we say that the interval J is a neighboring element of I if their intersection is nonempty.

By $L^2(\Omega)$ we denote the standard space of Lebesgue square-integrable functions on Ω.

Definition 1 (Sobolev space)**.** *Denote the interval* $(0, 1)$ *by* Ω*. Let us define the* Sobolev space $H^1(\Omega)$ *as*

$$H^1(\Omega) = \left\{ v \in L^2(\Omega);\, v' \in L^2(\Omega) \right\}.$$

equipped with the norm

$$|v|_{H^1} = \|v\|_{H^1(\Omega)} = \left(\|v\|^2_{L^2(\Omega)} + \|v'\|^2_{L^2(\Omega)} \right)^{\frac{1}{2}}.$$

The seminorm is defined as

$$|v|_{H^1} = \|v\|_{H^1(\Omega)} = \left(\|v'\|^2_{L^2(\Omega)} \right)^{\frac{1}{2}}.$$

Definition 2 (Broken Sobolev spaces)**.** *We define the* broken Sobolev spaces *over* $\mathcal{T}_h$ *as follows*

$$H^1(\Omega, \mathcal{T}_h) = \left\{ v; v|_I \in H^1(I), \forall I \in \mathcal{T}_h \right\}.$$

equipped with the seminorm

$$|v|_{H^1(\Omega, \mathcal{T}_h)} = \left(\sum_{I \in \mathcal{T}_h} |v|_{H^1(I)} \right)^{\frac{1}{2}}.$$

Actually the functions from $H^1(\Omega, \mathcal{T}_h)$ are generally globally discontinuous but on the interior of $I \in \mathcal{T}_h$ they are H^1 functions.

Notation. For discontinuous functions from $H^1(\Omega, \mathcal{T}_h)$ it is convenient to denote

$$u_h^-(x_i) = \lim_{x \to x_i^-} u_h(x),$$

$$u_h^+(x_i) = \lim_{x \to x_i^+} u_h(x).$$

Let $v \in H^1(\Omega, \mathcal{T}_h)$ be a function which is generally discontinuous. Hence it is suitable to introduce the notation for *jump*

$$[v]_i = v(x_i)^+ - v(x_i)^-,$$

where x_i is a boundary point of $I \in \mathcal{T}_h$. Special care must be taken for the points 0 and 1, because the value $v(0)^-$ and $v(1)^+$ are not defined. We formally define the values by extrapolating from the interior of Ω

$$v(0)^- := v(0)^+, \quad v(1)^+ := v(1)^-.$$

Let $P^1(I)$ denote the space of linear polynomials on the interval I. Then we define the space of discontinuous piecewise polynomial functions

$$S_h = \left\{ v; v|_I \in P^1(I), \ \forall I \in \mathcal{T}_h \right\}.$$

In the space S_h we would like to seek the solution of the discrete problem. That means that we use piecewise linear functions to approximate a piecewise Sobolev function from $H^1(\Omega, \mathcal{T}_h)$.

Definition 3. *Let f be a real valued function from Ω to $\mathbb{R}$ then the* support *of f is defined as*

$$\operatorname{supp}(f) = \overline{\{x \in \Omega | f(x) \neq 0\}}.$$

2. Nonlinear convection equation

Let us consider the following one dimensional partial differential equation.

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = 0 \quad \text{in } (0,1) \times (0,T). \tag{2.1}$$

We seek a function $u : (0,1) \times (0,T) \to \mathbb{R}$ satisfying (2.1) under the *boundary conditions*

$$u(x,t) = u_D(x,t), \quad x \in \Gamma_D, t \in (0,T)$$

and the *initial condition*

$$u(x,0) = u_0(x), \quad 0 \leq x \leq 1,$$

where the function $u_0 : (0,1) \to \mathbb{R}$ and the Dirichlet data $u_D : \Gamma_D \times (0,T) \to \mathbb{R}$ are prescribed. The Dirichlet boundary Γ_D is a subset of $\{0,1\}$, which we will describe later. The function $f(u) \in C^1(\mathbb{R})$ is called *convective flux*. It is often assumed that the flux is globally Lipschitz continuous. However, this assumption is sometimes not satisfied, the Burgers' equation investigated in this work is Lipshitz continuous only locally.

By the choice of f we obtain equations describing different phenomena such as city traffic, behavior of electrons in semiconductors or fluid flows.

As a linear version of (2.1) we take

$$\frac{\partial u}{\partial t} + \frac{\partial a(x,t)u}{\partial x} = 0 \quad \text{in } (0,1) \times (0,T). \tag{2.2}$$

The linear equation can be very useful in understanding the behavior of the nonlinear case.

By the choice $f(u) = u^2$ we obtain

$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} = 0 \quad \text{in } (0,1) \times (0,T), \tag{2.3}$$

which is a special case of equation (2.1) and it is known as *inviscid Burgers' equation*. This equation has been studied by researchers because it is one of the simplest form of nonlinear partial differential equation which in addition simulates the physical phenomenon of shock wave.

Note that in the literature the Burgers' equation is usually stated with $f(u) = \frac{1}{2}u^2$. In order to keep the presented methods as simple as possible, we omit the factor one half.

2.1 DG formulation

We inherit the notation from Section 1. Using the usual approach, the equation (2.1) is multiplied by a test function $\phi \in H^1((0,1), \mathcal{T}_h)$, integrated over interval

$(x_i, x_{i+1}) \in \mathcal{T}_h$ and integration by parts is applied to the convective terms. The result is the equation

$$\int_{x_i}^{x_{i+1}} \frac{\partial u}{\partial t}\phi \,\mathrm{d}x + [f(u)\phi]_{x_i}^{x_{i+1}} - \int_{x_i}^{x_{i+1}} f(u)\phi' \,\mathrm{d}x = 0.$$

Now we sum over all intervals $(x_i, x_{i+1}) \in \mathcal{T}_h$

$$\int_0^1 \frac{\partial u}{\partial t}\phi \,\mathrm{d}x + \sum_{i=0}^{n-1}[f(u)\phi]_{x_i}^{x_{i+1}} - \sum_{i=0}^{n-1}\int_{x_i}^{x_{i+1}} f(u)\phi' \,\mathrm{d}x = 0.$$

We would like to replace the continuous function u by its discrete approximation $u_h \in S_h$. The problem is that the function u_h is not continuous on $(0,1)$. Loosely speaking, in point x_i the value of $u_h(x_i)$ may differ approaching the point x_i from the left and from the right. Hence the formal symbol $[f(u_h)\phi]_{x_i}^{x_{i+1}}$ does not make sense for u_h.

To provide a suitable interpretation, first rearrange the sum

$$\sum_{i=0}^{n-1}[f(u)\phi]_{x_i}^{x_{i+1}} = \sum_{x_i \in D_i} f(u)[\phi]_i.$$

We will approximate a physical flux $\sum_{x_i\in D_i} f(u)[\phi]_i$ by a numerical flux $\sum_{x_i\in D_i} H(u^-, u^+)[\phi]_i$. In the multidimensional problem, the numerical flux H depends also on a vector of an outer unit normal denoted by n. In the $1D$ case, n attains the values 1 or -1 and it is natural to choose $n = 1$ meaning that u^- is the value given in the left interval and u^+ in the right. The symbols will be properly defined later.

The above considerations allow us to define the *convective form*

$$b_h(u,\phi) = -\sum_{i=0}^{n-1}\int_{x_i}^{x_{i+1}} f(u)\phi' \,\mathrm{d}x + \sum_{i=0}^{n} H(u^-, u^+)[\phi]_i. \tag{2.4}$$

Finally we are able to formulate the discrete problem derived from (2.1).

Definition 4. *We say that u_h is a DG finite element solution of the nonlinear partial differential equation (2.1) if*

1. $u_h \in C^1([0,T]; S_h)$,
2. $\dfrac{d}{dt}(u_h(t), \phi_h) + b_h(u_h(t), \phi_h) = 0, \quad \forall \phi_h \in S_h, \forall t \in (0,T),$ (2.5)
3. $u_h(0) = (u_0)_h$,

where $(u_0)_h$ denotes an approximation of the initial condition u_0 in the space S_h.

Remark. By the notation $(\cdot,\cdot)$ we understand the standard L^2 inner product. i.e.

$$(u,v) = \int_0^1 u(x)v(x)\mathrm{d}x, \quad \forall u, v \in L^2(0,1).$$

2.2 Numerical flux

How to define the numerical flux $H(u^-, u^+)$?

Consider an arbitrary point x_i. Since u_h is not continuous, instead of a uniquely defined value in x_i, there are two values $u_h^-(x_i)$ and $u_h^+(x_i)$. It may seem natural to choose

$$f(u(x_i)) \approx \left(f(u_h^-(x_i)) + f(u_h^+(x_i))\right)/2$$

or

$$f(u(x_i)) \approx f\left((u_h^-(x_i) + u_h^+(x_i))/2\right).$$

However, such a choice of numerical flux leads to unstable schemes, for further details on this topic see Feistauer et al. [2003]. Vaguely speaking, this is caused by the fact, that averaging is natural for diffusive problems. In diffusive problems the information spreads in all directions more or less evenly. On the contrary, in the case of convective problems, the information is transported in some direction, hence it is more suitable to use *upwinding*. The proper proof is beyond the scope of this work, refer to Feistauer et al. [2003].

In the next paragraph we explain what it is mean by upwinding. It is a numerical method for solving partial differential equations, which uses an adaptive approach to numerically simulate the direction of propagation of information in a flow field. We will demonstrate the idea on the one-dimensional linear advection equation

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} = 0. \tag{2.6}$$

The simplest upwind scheme using the finite difference method for discretization is given by

$$\begin{aligned} \frac{u_i^{n+1} - u_i^n}{\delta t} + a\frac{u_i^n - u_{i-1}^n}{\Delta x} &= 0 \quad \text{for } a > 0, \\ \frac{u_i^{n+1} - u_i^n}{\delta t} + a\frac{u_{i+1}^n - u_i^n}{\Delta x} &= 0 \quad \text{for } a < 0, \end{aligned}$$

where u_i^n approximates $u(x_i, t_n)$. The second term depends on the direction of propagation. If $a > 0$, information goes from left to right and for the computation of the new solution u_i^{n+1} we use the values u_{i-1}^n. On the other hand, if $a < 0$, the information is transported from right to left and in the second term we use u_{i+1}^n to compute u_i^{n+1}.

The above considerations leads us to the following definition of the numerical flux.

Definition 5 (Upwind flux)**.**

$$H(u_h^-, u_h^+) = \begin{cases} f(u_h^-), & \text{if } A > 0, \\ f(u_h^+), & \text{if } A \leq 0, \end{cases}$$

where

$$A = f'\left((u_h^-(x_i) + u_h^+(x_i))/2\right).$$

In the simpler equation (2.2), where $f(u)$ depends linearly on u, we have

$$H(u_h^-, u_h^+) = \begin{cases} a(x,t)u_h^-, & \text{if } a(x,t) > 0, \\ a(x,t)u_h^+, & \text{if } a(x,t) \leq 0. \end{cases}$$

The convective term is given by $a(x,t)u_h$. The derivative of the convective term with respect to u is $a(x,t)$ and its sign determines the direction of propagation of information. If the direction of convection aims from the interval (x_{i-1}, x_i) to interval (x_i, x_{i+1}), we evaluate the flux using u_h^-. If the information propagates in the opposite direction, we evaluate using u_h^+. Basically we use the value lying in the upwind direction because this is the "important" information we want to use in further computations.

To give the reader a complete overview, we will introduce the flux properties in the general space $\mathbb{R}^d$. Let $H(u, v, \mathbf{n})$ be a general flux, where $\mathbf{n}$ denotes the outer unit normal. Then we assume that H has the following properties:

1. $H(u, v, \mathbf{n})$ is *consistent*:

$$H(u, u, \mathbf{n}) = \sum_{s=1}^{d} f_s(u) n_s, \quad \forall u \in \mathbb{R},\ \forall \mathbf{n} = (n_1, \ldots, n_d) \in \mathbb{R}^d : |\mathbf{n}| = 1.$$

2. $H(u, v, \mathbf{n})$ is *conservative*:

$$H(u, v, \mathbf{n}) = -H(u, v, -\mathbf{n}), \quad \forall u, v \in \mathbb{R}, \mathbf{n} \in \mathbb{R}^d : |\mathbf{n}| = 1.$$

The consistency property ensures that if we insert the exact solution u into $H(u, v, \mathbf{n})$, then the obtained discrete scheme is consistent with the continuous problem (2.1).

In the 1D case studied in this work, the outer unit normal n can achieve only two values 1 and -1. As explained above, the natural choice is $n = 1$, so it's sufficient to denote the numerical flux by $H(u, v)$ omitting the normal n.

Remark. The numerical flux presented in Definition 5 is by its definition consistent and conservative.

Numerical flux and boundary conditions

We must define the numerical flux also in the boundary points. The problem is closely related to the formulation of boundary conditions. Let Γ_D be prescribed on the *inflow* boundaries. By the term "inflow" we understand the parts of boundaries, where $f'(u)n < 0$. In other words, in the "inflow" points, the information propagates from the outside of the interval $(0, 1)$ into the interval. Hence, the value defined in point 0 from the left and in point 1 from the right must reflect the nature of the prescribed boundary condition.

A straightforward choice is

$$u^-(0) = \begin{cases} u_D, & \text{if } 0 \in \Gamma_D, \\ u^+(0), & \text{otherwise}, \end{cases}$$

$$u^+(1) = \begin{cases} u_D, & \text{if } 1 \in \Gamma_D, \\ u^-(1), & \text{otherwise}. \end{cases}$$

Remark. There are several possibilities how to set u_D. Typically u_D is defined by some function derived from the physical nature of the problem. Another option is to prescribe periodic boundary conditions. If the transport of the information goes from left to right, we prescribe $u^-(0) := u^-(1)$. In other words, if the information gets to the point 1, it appears in 0 afterwards. Another often used approach is to prescribe boundary condition from the initial condition u_0.

As it was already suggested in the beginning of this section, we often do not need to prescribe both $u^+(1)$ and $u^-(0)$. Whether we need to prescribe a boundary condition in the given end point or not depends on the propagation of the information. If the information flows from left to right, we need to set a boundary condition in 0 only. Similarly if the flow goes from right to left, we define the boundary conditions only in the point 1.

2.3 Time discretization

Further on we will be concerned with the Burgers' equation (2.3).

Let N be the dimension of the finite dimensional space S_h and $\mathcal{B} = \{\phi_1, \dots, \phi_N\}$ be its basis. The sought approximate solution $u_h \in S_h$ can be decomposed as a linear combination of basis functions ϕ_j and coefficients $u_j(t)$, $j = 1, \dots, N$

$$u_h(x,t) = \sum_{j=1}^{N} u_j(t)\phi_j(x).$$

Instead of testing (2.5) by an arbitrary function $\phi \in S_h$, we use only elements of the basis $\mathcal{B}$ and obtain a system of N ordinary differential equations for unknown $u_j, j = 1, \dots, N$.

$$\sum_{j=1}^{N}(\phi_j, \phi_i)\frac{\mathrm{d}u_j(t)}{\mathrm{d}t} + b_h(\sum_{j=1}^{N} u_j(t)\phi_j, \phi_i) = 0, \quad i = 1, \dots, N. \tag{2.7}$$

It is convenient to write the above system in the matrix form

$$\mathbb{M}_C\frac{\mathrm{d}u}{\mathrm{d}t} + B(u(t)) = 0,$$

where $\mathbb{M}_C = \{m_{ij}\}_{i,j=1}^{N}$ is the consistent mass matrix. The vector $B(u(t))$ corresponds to the discretization of the convective term and the unknown vector $u(t)$ is the vector of coefficients of the linear combination of element in $\mathcal{B}$.

Elements of the mass matrix are defined as follows

$$m_{ij} = \int_\Omega \phi_i\phi_j \,\mathrm{d}x,$$

and elements of vector $B(u(t))$ are

$$b_i = b_h\left(\sum_{j=1}^{N} u_j\phi_j, \phi_i\right).$$

In general $B(u(t))$ cannot be written as a product of matrix $\tilde{B}$ and vector $u(t)$, because neither $H(u_h^-, u_h^+)$ nor $f(u)$ depends linearly on u.

We follow by the time discretization. The simplest method is the forward Euler scheme (explicit). There are two main disadvantages, the method is only first order accurate and second, in order to preserve stability, restrictions on the choice of the time step must be applied, the topic is covered by Feistauer et al. [2003]. Usually the second problem is limiting for us, therefore we discretize the scheme using the backward Euler scheme (implicit).

Let $0 = t_0 < t_1 < \cdots < t_K = T$ be a partition of the interval $(0, T)$ and $\Delta t_{k+1} := t_{k+1} - t_k$. Let $u^k \approx u(t_k)$, then

$$\mathbb{M}_C u^{k+1} + \Delta t_{k+1} B(u^{k+1}) = \mathbb{M}_C u^k \tag{2.8}$$

is a scheme, which represents the backward Euler discretization of problem (2.1).

2.4 Linearization of the operator $B(u)$

The FCT method requires the discrete system in the form

$$(\mathbb{M}_C + \Delta t_{k+1} \mathbb{A}) u^{k+1} = \mathbb{M}_C u^k.$$

Before we proceed to the derivation of the method, we need to linearize the operator $B(u)$ corresponding to the discretization of the nonlinear convective term.

For Burgers' equation we approximate $b_h(u^{k+1}, \phi)$ by $b_h(u^{k+1}, u^k, \phi)$ defined as

$$b_h(u^{k+1}, u^k, \phi) = -\sum_{i=0}^{n-1} \int_{x_i}^{x_{i+1}} u^{k+1} u^k \phi' \,\mathrm{d}x + \sum_{i=0}^{n} \hat{H}\left((u^{k+1})^-, (u^{k+1})^+\right) [\phi]_i.$$

and approximate the numerical flux

$$\hat{H}\left((u^{k+1})^-, (u^{k+1})^+\right) = \begin{cases} (u^k)^- (u^{k+1})^-, & \text{if } A(x_i) > 0, \\ (u^k)^+ (u^{k+1})^+, & \text{if } A(x_i) \le 0, \end{cases}$$

where

$$A(x_i) = \left(u^k(x_i^-) + u^k(x_i^+)\right)/2.$$

The above linearization is sufficient for the backward Euler time discretisation, for more details refer to Feistauer and Kučera [2007], where a similar linearization was used for the compressible Euler equation. Formally $\hat{H}$ depends also on $(u^k)^-$ and $(u^k)^+$, however, we omit these arguments for simplicity.

We express u^{k+1} in terms of basis functions. Then

$$\begin{aligned} b_h(u^{k+1}, u^k, \phi) = &-\sum_{j=1}^{N} u_j^{k+1} \sum_{i=0}^{n-1} \int_{x_i}^{x_{i+1}} u^k \phi_j \phi' \,\mathrm{d}x \\ &+ \sum_{j=1}^{N} u_j^{k+1} \sum_{i=0}^{n} \left(A^+(x_i)\phi_j(x_i^-) + A^-(x_i)\phi_j(x_i^+)\right)(\phi(x_i^-) - \phi(x_i^+)). \end{aligned}$$

where we denote

$$A^+(x_i) := u^k(x_i^-)\mathcal{H}(A(x_i)), \tag{2.9}$$

$$A^-(x_i) := u^k(x_i^+)\mathcal{H}(-A(x_i)). \tag{2.10}$$

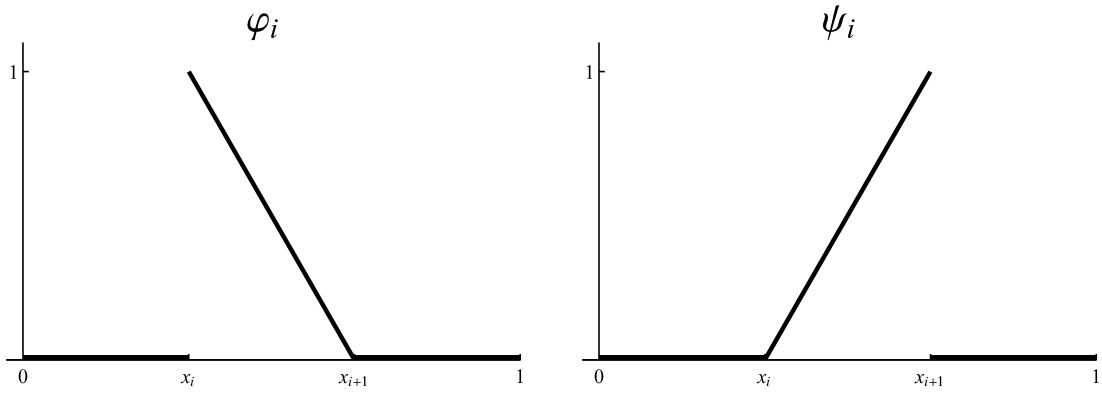


Figure 2.1: Basis functions φ_i and ψ_i respectively.

The symbol $\mathcal{H}(A(x_i))$ denotes the Heaviside function, whose value is zero for non positive argument and one for positive argument in the given point x_i.

To provide the setup which is required for the theory behind the FCT method, we use the following basis functions.

$$\varphi_i := \begin{cases} \frac{x-x_{i+1}}{x_i-x_{i+1}}, & \text{in } (x_i, x_{i+1}), \\ 0, & \text{elsewhere,} \end{cases} \tag{2.11}$$

$$\psi_i := \begin{cases} \frac{x-x_i}{x_{i+1}-x_i}, & \text{in } (x_i, x_{i+1}), \\ 0, & \text{elsewhere.} \end{cases} \tag{2.12}$$

The basis functions are chosen in such a way that the support of φ_i and ψ_i is only the element (x_i, x_{i+1}), see Figure (2.1).

For this specific basis, we will use the notation

$$u^k := \sum_{j=1}^{n} \beta_j^k \varphi_j + \gamma_j^k \psi_j.$$

Finally we can simplify the formula for b_h. First, we use the test function $\phi := \varphi_m$. Using the fact that the m-th basis function is zero outside the interval $[x_m, x_{m+1}]$ we have

$$\begin{aligned} b_h(u^{k+1}, u^k, \varphi_m) = &-\beta_m^{k+1} \int_{x_m}^{x_{m+1}} u_k \varphi_m \varphi_m' \,\mathrm{d}x - \gamma_m^{k+1} \int_{x_m}^{x_{m+1}} u_k \psi_m \varphi_m' \,\mathrm{d}x \\ &+ \sum_{j=1}^{n} \beta_j^{k+1} \sum_{i=0}^{n} \left(A^+(x_i)\varphi_j(x_i^-) + A^-(x_i)\varphi_j(x_i^+) \right) (\varphi_m(x_i^-) - \varphi_m(x_i^+)) \\ &+ \sum_{j=1}^{n} \gamma_j^{k+1} \sum_{i=0}^{n} \left(A^+(x_i)\psi_j(x_i^-) + A^-(x_i)\psi_j(x_i^+) \right) (\varphi_m(x_i^-) - \varphi_m(x_i^+)). \end{aligned}$$

Further

- $\varphi_m(x_i^-) = 0 \quad \forall i = 0, \dots, n,$
- $\varphi_m(x_i^+) = 1$ if and only if $i = m,$
- $\varphi_j(x_m^-) = 0 \quad \forall j = 1, \dots, n,$
- $\varphi_j(x_m^+) = 1$ if and only if $j = m,$
- $\psi_j(x_m^-) = 1$ if and only if $j = m - 1,$
- $\psi_j(x_m^+) = 0 \quad \forall j = 1, \dots, n.$

Denote the integrals $I_m^1 := \int_{x_m}^{x_{m+1}} u^k \varphi_m \varphi_m' \,\mathrm{d}x$ and $I_m^2 := \int_{x_m}^{x_{m+1}} u^k \psi_m \varphi_m' \,\mathrm{d}x$ and using the above considerations, we obtain

$$b_h(u^{k+1}, u^k, \varphi_m) = -\beta_m^{k+1} I_m^1 - \gamma_m^{k+1} I_m^2 - \beta_m^{k+1} A^-(x_m) - \gamma_{m-1}^{k+1} A^+(x_m).$$

Similarly, if we use the test function ψ_m we derive the form of b_h to be

$$b_h(u^{k+1}, u^k, \psi_m) = -\beta_m^{k+1} J_m^1 - \gamma_m^{k+1} J_m^2 + \beta_{m+1}^{k+1} A^-(x_{m+1}) + \gamma_m^{k+1} A^+(x_{m+1}),$$

where J_m^1 and J_m^2 denotes

$$J_m^1 = \int_{x_m}^{x_{m+1}} u^k \varphi_m \psi_m' \,\mathrm{d}x, \quad J_m^2 = \int_{x_m}^{x_{m+1}} u^k \psi_m \psi_m' \,\mathrm{d}x.$$

2.5 Discrete scheme

Let the partition of the interval $(0, 1)$ be equidistant and denote the length of any subinterval by h.

For the basis functions (2.11) and (2.12) presented in the previous subsection, we can write the explicit form of the linearized scheme (2.8).

In order to express the mass matrix $\mathbb{M}_C$, we evaluate the inner products of the basis functions

- $\int_{x_m}^{x_{m+1}} \varphi_m \varphi_m \,\mathrm{d}x = h/3,$
- $\int_{x_m}^{x_{m+1}} \varphi_m \psi_m \,\mathrm{d}x = h/6,$
- $\int_{x_m}^{x_{m+1}} \psi_m \psi_m \,\mathrm{d}x = h/3.$

Then

$$\mathbb{M}_C = \begin{bmatrix} \frac{h}{3} & \frac{h}{6} & & & & & \\ \frac{h}{6} & \frac{h}{3} & & & & 0 & \\ & & \frac{h}{3} & \frac{h}{6} & & & \\ & & \frac{h}{6} & \frac{h}{3} & & & \\ & & & & \ddots & & \\ & 0 & & & & \frac{h}{3} & \frac{h}{6} \\ & & & & & \frac{h}{6} & \frac{h}{3} \end{bmatrix}$$

The linearized convective operator b_h rewritten in the block matrix form is

$$\mathbb{A} = \begin{bmatrix} \mathbb{A}_{1,1} & \mathbb{A}_{1,2} & & 0 \\ \mathbb{A}_{2,1} & \mathbb{A}_{2,2} & \mathbb{A}_{2,3} & \\ & & \ddots & \\ 0 & & \mathbb{A}_{n,n-1} & \mathbb{A}_{n,n} \end{bmatrix}$$

with the 2x2 blocks defined as

$$\begin{aligned} \mathbb{A}_{m,m-1} &= \begin{bmatrix} 0 & A^+(x_m) \\ 0 & 0 \end{bmatrix}, \\ \mathbb{A}_{m,m} &= \begin{bmatrix} I_m^1 + A^-(x_m) & I_m^2 \\ J_m^1 & J_m^2 - A^+(x_{m+1}) \end{bmatrix}, \\ \mathbb{A}_{m,m+1} &= \begin{bmatrix} 0 & 0 \\ -A^-(x_{m+1}) & 0 \end{bmatrix}. \end{aligned}$$

Now we are able to write the discrete scheme (2.8) in the exact form with the system matrix to be block tridiagonal matrix

$$\left(\mathbb{M}_C - \Delta t_{k+1} \begin{bmatrix} \mathbb{A}_{1,1} & \mathbb{A}_{1,2} & & 0 \\ \mathbb{A}_{2,1} & \mathbb{A}_{2,2} & \mathbb{A}_{2,3} & \\ & & \ddots & \\ 0 & & \mathbb{A}_{n,n-1} & \mathbb{A}_{n,n} \end{bmatrix} \right) \begin{bmatrix} \beta_1^{k+1} \\ \gamma_1^{k+1} \\ \beta_2^{k+1} \\ \gamma_2^{k+1} \\ \vdots \\ \beta_n^{k+1} \\ \gamma_n^{k+1} \end{bmatrix} = \mathbb{M}_C \begin{bmatrix} \beta_1^{k} \\ \gamma_1^{k} \\ \beta_2^{k} \\ \gamma_2^{k} \\ \vdots \\ \beta_n^{k} \\ \gamma_n^{k} \end{bmatrix}. \tag{2.13}$$

3. Flux corrected schemes

The FEM-FCT method was developed in the early 70s by Boris and Book [1973]. The method was inspired by solving the continuity equation of gas dynamics to obtain realistic and accurate results. The goal was to preserve the mass-conserving property. For more details refer to Boris and Book [1973].

The FCT method is based on an algebraic modification of the system matrix and right hand side such that the new system possesses "better" numerical properties. More precisely, if the maximum principle holds for continuous equation (2.1), the discrete maximum principle should be inherited also for the discrete problem (2.8). Forcing the discrete maximum principle to be satisfied is the main idea behind FCT algorithm. It can be shown that for preservation of DMP it is enough that a matrix is an M-matrix [cf. Kuzmin and Möller, 2005]. In this section we use some theoretical results that are presented in Kuzmin and Möller [2005] or Kuzmin [2010] .

Definition 6 (maximum principle (MP))**.** *Let Σ denotes the sets of points, where initial or boundary conditions are prescribed, i.e. $\Sigma = \{(x,t) : x \in \Gamma_D \vee t = 0\}$. The continuous* maximum principle *holds for problem (2.1) if*

$$\min_{\Sigma} u \leq u(x,t) \leq \max_{\Sigma} u, \quad \forall (x,t) \in (0,1) \times (0,T). \tag{3.1}$$

The condition (3.1) in fact says, that no new maxima nor minima of the solution arise inside the domain as the solution evolves in time. This property is natural for the transport equation and it is often used for verification whether the numerical scheme provides a reasonable solution.

Theorem 1. *The continuous maximum principle is satisfied for Burgers' equation (2.3), i.e. it holds $\min_{\Sigma} u \leq u \leq \max_{\Sigma} u$.*

Proof. The proof is done by the method of characteristic [see Feistauer et al., 2003].

□

Definition 7 (discrete maximum principle (DMP))**.** *The discrete solution u^{k+1} of (2.8) satisfies the local discrete maximum principle if*

$$u_i^{\min} \leq u_i^{k+1} \leq u_i^{\max},$$

where

$$u_i^{\max} = \max \left\{ \max_{j \in S_i \cup \{i\}} u_j^k, \max_{l \in S_i} u_l^{k+1} \right\},$$
$$u_i^{\min} = \min \left\{ \min_{j \in S_i \cup \{i\}} u_j^k, \min_{l \in S_i} u_{l^{k+1}} \right\}$$

and S_i is the set of the nearest neighboring nodes of node x_i.

We would like to formulate the necessary conditions for DMP to be satisfied. First we need introduce some matrix properties.

By the symbol $v \geq 0$ we mean that any component of v is greater than or equal to zero.

Definition 8. *A regular matrix* $\mathbb{A} \in \mathbb{R}^{n\times n}$ *is called* monotone *if*

$$\mathbb{A}v \geq 0 \Rightarrow v \geq 0, \quad \forall v \in \mathbb{R}^n.$$

Definition 9. *A monotone matrix* $\mathbb{A}$ *with* $a_{ij} \leq 0$ *for all* $j \neq i$, *is called an* M-matrix.

Claim 2. *Let* $\mathbb{A} \in \mathbb{R}^{n\times n}$ *be a matrix with elements* a_{ij}. *Suppose that* $\mathbb{A}$ *satisfies the following properties*

1. $a_{ii} > 0 \quad \forall i = 1, \ldots, n,$
2. $a_{ij} \leq 0 \quad \forall i \neq j;\ i, j = 1, \ldots, n,$
3. $\sum_{j=1}^{n} a_{ij} > 0 \quad \forall i = 1, \ldots, n.$

Then $\mathbb{A}$ *is an* M*-matrix.*

Proof. It is enough to show that $\mathbb{A}$ is monotone.

Let $\mathbb{A}u \geq 0$ and exists i such that $u_i < 0$. Without loss of generality suppose that

$$u_1 \;=\; \min\{u_1, u_2, \ldots, u_n\}.$$

Hence $u_1 < 0$.

Choose an arbitrary $j \in \{2, \ldots, n\}$. Inequalities $u_1 \leq u_j$ and $a_{1j} \leq 0$ imply $a_{1j}u_1 \geq a_{1j}u_j$. A direct consequence of previous relation is $(a_{12} + \ldots + a_{1n})u_1 \geq a_{12}u_2 + \ldots + a_{1n}u_n$.

Reformulate assumption 3 as follows: $a_{11} > -a_{12} - \ldots - a_{1n}$. Combining the last two inequalities above we get

$$a_{11}u_1 < (-a_{12} - \ldots - a_{1n})u_1 \leq -a_{12}u_2 - \ldots - a_{1n}u_n,$$

which immediately implies $a_{11}u_1 + a_{12}u_2 + \ldots + a_{1n}u_n < 0$. This is in contradiction with $\mathbb{A}u \geq 0$. □

In words, Claim 2 says that a sufficient condition for a matrix to be an M-matrix is that all diagonal elements are positive, all off-diagonal elements are non-positive and row sums are positive.

Theorem 3. *Let* $\mathbb{A}, \mathbb{B} \in \mathbb{R}^{n\times n}$, $u, g \in \mathbb{R}^n$. *Consider a fully discrete scheme* $\mathbb{A}u = \mathbb{B}g$. *Suppose that all assumptions in Claim 2 are valid. Assume, moreover, that* $b_{ij} \geq 0$ *for all* $i, j = 1, \ldots, N$.
Then if

$$\sum_j a_{ij} = \sum_j b_{ij}, \ \forall i = 1, \ldots, N \tag{3.2}$$

the local DMP holds.

Proof. By S_i we denote the set of indexes $j \in 1, \ldots, N$ such that a_{ij} or b_{ij} are not zero. The set S_i corresponds to the set of neighboring elements of node x_i. Then we define u_i^{max} and u_i^{min} the same way as in Definition 7.

For the discrete scheme $\mathbb{A}u = \mathbb{B}g$ the general form of the i-th equation reads

$$a_{ii}u_i = b_{ii}g_i + \sum_{j\in S_i} \left(b_{ij}g_j - a_{ij}u_j\right). \tag{3.3}$$

Define $w_j := u_j - u_i^{\max}$ and $v_j := g_j - u_i^{\max}$ such that

$$w_j \leq 0, \forall j \in S_i, \quad v_j \leq 0, \forall j \in S_i \cup \{i\}.$$

Using the row sum condition (3.2), we can express the equation (3.3) as

$$a_{ii} w_i = b_{ii} v_i + \sum_{j \in S_i} \left(b_{ij} v_j - a_{ij} w_j \right).$$

By the inequalities assumptions the right-hand side of above equation is non-positive and $a_{ii} > 0$. Hence $w_i \leq 0$ or equivalently $u_i \leq u_i^{\max}$.

Similarly one can prove $u_i \geq u_i^{\min}$.

Presented proof can be found in Kuzmin [2010]. □

The conditions stated above motivate us to define the following matrices. The matrix $\mathbb{A}$ stands for the linearized operator $B(u)$ from the previous chapter. Denote

$$\begin{aligned} \mathbb{L} &= \mathbb{A} + \mathbb{D}, \\ \mathbb{D} &= (d_{ij}), \end{aligned}$$

where

$$\begin{aligned} d_{ij} &= -\max\{0, a_{ij}, a_{ji}\}, \quad i \neq j, \\ d_{ii} &= -\sum_{j=1, j \neq i}^{N} d_{ij}, \end{aligned}$$

and

$$M_L = diag(m_1, \ldots, m_N), \quad m_i = \sum_{j=1}^{N} m_{ij},$$

where N is the same as in the previous chapter and denotes the number of degrees of freedom of space V_h, a_{ij} are entries of the stiffness matrix $\mathbb{A}$ and m_{ij} are entries of the mass matrix $\mathbb{M}_C$.

In the next several Lemmas we summarize the properties of the introduced matrices. The results will be used to prove that the designed scheme,which will be presented later, satisfies the local DMP.

Lemma 4. *Matrix $\mathbb{L}$ posses the following properties*

1. *if $a_{ii} \geq 0 \quad \forall i = 1, \ldots, N$ then $l_{ii} \geq 0 \quad \forall i = 1, \ldots, N$,*
2. *$l_{ij} \leq 0 \quad \forall i \neq j, i, j = 1, \ldots, N$.*

Proof. 1. Let $i \in \{1, \ldots, N\}$. Then

$$l_{ii} = a_{ii} - \sum_{j=1, i \neq j}^{N} -\max\{0, a_{ij}, a_{ji}\} \geq a_{ii} \geq 0.$$

2. Let $i, j \in \{1, \ldots, N\}, i \neq j$. Then $l_{ij} = a_{ij} - \max\{0, a_{ij}, a_{ji}\}$.

- If $a_{ij} > 0$, $l_{ij} = a_{ij} - \max\{a_{ij}, a_{ji}\} \leq a_{ij} - a_{ij} = 0$.
- If $a_{ij} \leq 0$, $l_{ij} = a_{ij} - \max\{0, a_{ji}\} \leq a_{ij} \leq 0$.

□

Lemma 5. *The row and column sums of* $\mathbb{D}$ *are zero.*

Proof. The zero sum property is an immediate consequence of the definition of matrix $\mathbb{D}$. □

Lemma 6. *Matrix* $\mathbb{M}_L$ *is an* M*-matrix.*

Proof. Use the characterization stated in Claim 2.

Entries m_{ij} are defined as (φ_i, φ_j) where φ_i are positive functions $\forall i = 1, \ldots, N$ hence conditions 1 and 3 are satisfied. The remaining condition is satisfied trivially because $\mathbb{M}_L$ is a diagonal matrix.

□

Corollary 7. *The matrix* $\mathbb{M}_L + \Delta t_{k+1}\mathbb{L}$ *is an* M*-matrix provided that* Δt_{k+1} *is sufficiently small.*

Proof. We use Claim 2 together with Lemma 4 and Lemma 6.

1. The diagonal entries of $\mathbb{M}_L$ are positive. If $a_{ii} \geq 0$, $\forall i, \ldots, N$ then $l_{ii} \geq 0$ for all $i, \ldots, N$ hence the sum $\mathbb{M}_L + \Delta t_{k+1}\mathbb{L}$ has positive diagonal entries. If exists $i \in 1, \ldots, N$ such that $a_{ii} < 0$ then under the condition that Δt_{k+1} is sufficiently small then $(\mathbb{M}_L)_{ii} > |\Delta t_{k+1} l_{ii}|$ and hence the positivity of diagonal entries of $\mathbb{M}_L + \Delta t_{k+1}\mathbb{L}$ is ensured.

2. Off-diagonal elements of both matrices are non-positive, also the sum is non-positive.

3. We want to prove that the row sums $\sum_{j=1}^{N} (\mathbb{M}_L + \Delta t_{k+1}\mathbb{L})_{ij} > 0$ for all $i = 1, \ldots, N$. We know, that the diagonal entries of $\mathbb{M}_L$ are positive with the value approximately equal to the norm of the partition h and off-diagonal entries of $\mathbb{M}_L$ are zero. If Δt_{k+1} is sufficiently small in comparison to the norm of the partition h, the sum of non-positive off-diagonal entries of $\mathbb{L}$ is smaller that the diagonal element of $\mathbb{M}_L$. Thus the inequality is fulfilled.

□

3.1 Low order scheme

We will now proceed with derivation of FCT scheme. Instead of equation (2.13) consider

$$(\mathbb{M}_L + \Delta t_{k+1}\mathbb{L})\, u^{k+1} = \mathbb{M}_L u^k. \tag{3.4}$$

This is an algebraic representation of a stable low order scheme (LOS).

Lemma 8. *Suppose that for Burgers' equation (2.3)* u_k *is constant on the whole interval* $(0, 1)$*, then the discrete maximum principle holds for the low order scheme (3.4).*

Proof. We use Theorem 3. Proving all inequalities is straightforward as a consequence of stated properties of $\mathbb{L}, \mathbb{M}_L$ and $\mathbb{D}$. The interesting part is to prove the condition

$$\sum_j a_{ij} = \sum_j b_{ij}, \ \forall i = 1, \dots, N$$

For the low order scheme it means to prove the row sums of $\mathbb{M}_L + \Delta t_{k+1}\mathbb{L}$ equals the row sum of $\mathbb{M}_L$. This is equivalent to prove $\sum_j l_{ij} = 0, \forall i = 1, \dots, N$. The matrix is defined as $\mathbb{L} = \mathbb{A} + \mathbb{D}$ and we know from Lemma 5 that the row sums of D are zeros. Thus, it is sufficient to show that $\sum_j a_{ij} = 0, \forall i = 1, \dots, N$. Equivalently written as $\mathbb{A}(1, \dots, 1)^T = 0$. In our scheme it corresponds to multiplying the matrix $\mathbb{A}$ by the vector $u^{k+1} \equiv 1$. The sum $\sum_j a_{ij}$ corresponds to $b_h(1, u^k, \phi)$, where ϕ is the i-th basis function and $u^{k+1} \equiv 1$. Now we use the assumption that u^k is constant, i.e. $u^k \equiv c$ then

$$b_h(1, c, \phi) = -\sum_{i=0}^{n-1} \int_{x_i}^{x_{i+1}} c\phi' \, \mathrm{d}x + \sum_{i=0}^{n} (A^+(x_i) + A^-(x_i))(\phi(x_i^-) - \phi(x_i^+)).$$

Using integration by parts we have

$$b_h(1, c, \phi) = \sum_{i=0}^{n-1} \int_{x_i}^{x_{i+1}} c'\phi \, \mathrm{d}x - \sum_{i=0}^{n-1} c(\phi(x_{i+1}^-) - \phi(x_i^+)) + \sum_{i=0}^{n} c(\phi(x_i^-) - \phi(x_i^+)).$$

The derivative of the constant c is zero, hence the first term is zero. If we reorganize the sum in second term, the second and third term will be subtracted, using the fact that we had formally defined $u^k(0^-) := 0$ and $u^k(1^+) := 0$.

To sum it up, we have proven that $\sum_j a_{ij} = 0, \forall i = 1, \dots, N$. □

Originally, the scheme was developed for the transport equation and nowadays it is also used for the convection-diffusion-reaction equation

$$\frac{\partial u}{\partial t} + \nabla \cdot (\mathbf{v}u - D\nabla u) = 0$$

but restricted only to the problems with an incompressible velocity fields ($\nabla \cdot \mathbf{v} = 0$), which can be viewed as one of the drawback of the method. For further details refer to Kuzmin [2010].

The problem is that the row sum property (3.2) does not hold for the Burger's equation in general. It is a direct consequence of the fact that u^k does not need to be identically constant on the whole interval $(0, 1)$. The violation of this condition provides the solution which possesses overshoots and undershoots.

We need to enforce the condition (3.2). The first possibility is to modify u^k such that $b_h(u^{k+1}, u^k, \phi)$ equals zero. The second approach is to add a penalty term to the diagonal of matrix $\mathbb{A}$, in a way that the row sum condition is satisfied.

The advantage of low order scheme is that no new extremes arise and consequently the solution does not show spurious oscillations. However, this scheme is rather too diffusive and the obtained solution is smeared in the non-smooth regions. For this reason we will introduce the flux corrected scheme in Section 3.3.

3.1.1 Modification of the solution on previous time step

The convective form $b_h(u^{k+1}, u^k, \phi)$ consists of two terms: sum of integrals over elements and sum of numerical fluxes. We need to modify u^k so that the convective form $b_h(1, u^k, \phi)$ equals zero as required by Lemma 8.

First, in the integral, instead of $u^k|_{(x_i,x_{i+1})}$ we use the average value over the interval (x_i, x_{i+1}), thus we take

$$\overline{u^k}|_{(x_i,x_{i+1})} = \left(u^k(x_i^+) + u^k(x_{i+1}^-)\right)/2. \tag{3.5}$$

We will denote the piecewise constant function by $\overline{u^k}$.

Second, in the numerical flux, the criterion whether the information propagates from left to right or otherwise was determined by the sign of $A(x_i)$ defined as

$$A(x_i) = \left(u^k(x_i^-) + u^k(x_i^+)\right)/2.$$

Now, we do not use the values form neighboring intervals and approximate $A(x_i)$ by

$$\begin{aligned} A(x_i)|_{(x_i,x_{i+1})} &= \left(u^k(x_i^+) + u^k(x_{i+1}^-)\right)/2, \\ A(x_{i+1})|_{(x_i,x_{i+1})} &= \left(u^k(x_i^+) + u^k(x_{i+1}^-)\right)/2. \end{aligned}$$

This way we can achieve the fulfillment of the row sum condition.

We define modified convective term $\tilde{b}_h$ as follows

$$\begin{aligned} \tilde{b}_h(u^{k+1}, \overline{u^k}, \phi) = &- \sum_{i=0}^{n-1} \int_{x_i}^{x_{i+1}} u^{k+1}\overline{u^k}\phi' \,\mathrm{d}x \\ &+ \sum_{i=0}^{n} \left(u^{k+1}(x_i^-)A^+(x_i) + u^{k+1}(x_i^+)A^-(x_i)\right)\left(\phi(x_i^-) - \phi(x_i^+)\right), \end{aligned}$$

where

$$A^+(x_i) := \overline{u^k}|_{(x_i,x_{i+1})}\mathcal{H}\left(A(x_i)|_{(x_i,x_{i+1})}\right), \tag{3.6}$$

$$A^-(x_i) := \overline{u^k}|_{(x_i,x_{i+1})}\mathcal{H}\left(-A(x_i)|_{(x_i,x_{i+1})}\right). \tag{3.7}$$

We can verify that the condition (3.2) is satisfied for the convective term $\tilde{b}_h$, we proceed the same way as in the proof of Lemma 8. Set $u^{k+1} \equiv 1$:

$$\tilde{b}_h(1, \overline{u^k}, \phi) = -\sum_{i=0}^{n-1} \int_{x_i}^{x_{i+1}} \overline{u^k}\phi' \,\mathrm{d}x + \sum_{i=0}^{n} \left(A^+(x_i) + A^-(x_i)\right)\left(\phi(x_i^-) - \phi(x_i^+)\right).$$

We can further simplify the equation above by realizing that we test the equation by the base function which support is only one element. Let $\mathrm{supp}(\phi) = [x_j, x_{j+1}]$ then

$$\begin{aligned} \tilde{b}_h(1, \overline{u^k}, \phi) = &- \int_{x_j}^{x_{j+1}} \overline{u^k}\phi' \,\mathrm{d}x + \left(A^+(x_j) + A^-(x_j)\right)\Big(\underbrace{\phi(x_j^-)}_{=0} - \phi(x_j^+)\Big) \\ &+ \left(A^+(x_{j+1}) + A^-(x_{j+1})\right)\Big(\phi(x_{j+1}^-) - \underbrace{\phi(x_{j+1}^+)}_{=0}\Big). \end{aligned}$$

The idea is to replace the piecewise linear function u^k by the piecewise constant function in order to get zero in the integral term after integration by parts:

$$\tilde{b}_h(1, \overline{u^k}, \phi) = \int_{x_j}^{x_{j+1}} \underbrace{\overline{(u^k)'}}_{=0} \phi \,\mathrm{d}x - \overline{u^k}|_{(x_j,x_{j+1})}\phi(x_{j+1}^-) + \overline{u^k}|_{(x_j,x_{j+1})}\phi(x_j^+)$$
$$- \left(A^+(x_j) + A^-(x_j)\right)\phi(x_j^+) + \left(A^+(x_{j+1}) + A^-(x_{j+1})\right)\phi(x_{j+1}^-).$$

The next equality explains why we have modified the numerical flux as in 3.6 and 3.7. We take $A^+(x_j)$ and $A^-(x_j)$ with respect to the interval $[x_j, x_{j+1}]$ because we use the test function ϕ with support $[x_j, x_{j+1}]$. Then

$$\begin{aligned} A^+(x_j) + A^-(x_j) &= \overline{u^k}|_{(x_j,x_{j+1})}\left(\mathcal{H}\left(A(x_j)|_{(x_j,x_{j+1})}\right) + \mathcal{H}\left(-A(x_j)|_{(x_j,x_{j+1})}\right)\right) \\ &= \overline{u^k}|_{(x_j,x_{j+1})}. \end{aligned}$$

To conclude, the numerical fluxes are subtracted and hence $\tilde{b}_h(1, \overline{u^k}, \phi) = 0$ for any base test function ϕ.

The major disadvantage of this approach is that the communication between neighboring elements is not correct. More precisely the direction of propagation is given by the average value over a given element and it ignores the information given by the neighboring elements, thus the information propagation can completely stop in extreme cases. For example if u^k is the discontinuous function defined as 1 on $(0, 0.5)$ and 0 elsewhere, the solution on the new time level u^{k+1} should propagate the value 1 to the right with the velocity equal to 1 due to the Rankine-Hugoniot [cf. Feistauer et al., 2003] condition. However, computing the new solution u^{k+1} using the average values as presented in this subsection provides the solution $u^{k+1} = u^k$ since the elements where $u^k = 0$ will only propagate information from neighboring elements with velocity 0, even when the value on the neighboring element is equal to 1.

3.1.2 Penalty term contribution

The simplest way how to enforce the row sum condition (3.2) is to sum each row of matrix $\mathbb{A}$ and modify the diagonal element. We combine the average value modification and this straightforward approach. Again, we would like to have $b_h(u^{k+1}, u^k, \phi) = 0$. In the integral term, we use the average value over the interval (x_i, x_{i+1}) exactly the same way as in previous subsection. However we would like to preserve the numerical fluxes to be as in standard DG scheme. Hence we must subtract the excessive value from the diagonal.

The convective form $\hat{b}_h$ is

$$\hat{b}_h(u^{k+1}, \overline{u^k}, \phi) = -\sum_{i=0}^{n-1} \int_{x_i}^{x_{i+1}} u^{k+1}\overline{u^k}\phi' \,\mathrm{d}x$$
$$+ \sum_{i=0}^{n} \left(u^{k+1}(x_i^-)A^+(x_i) + u^{k+1}(x_i^+)A^-(x_i)\right)\left(\phi(x_i^-) - \phi(x_i^+)\right),$$

where $\overline{u^k}$ is defined as (3.5) but $A^+(x_i)$ and $A^-(x_i)$ is defined as in standard DG method (2.9) and (2.10).

Using the modified convective form $\hat{b}_h$ we compute a_{ij}, then

$$\tilde{a}_{ii} := a_{ii} - p_i,$$

where the penalty term is

$$p_i := \sum_{j=0}^{n} a_{ij}, \quad i = 1, \ldots, n.$$

If $\sum_{j=0}^{n} a_{ij} = 0$ then also $\sum_{j=0}^{n} l_{ij} = 0$ using that each row sum of $\mathbb{D}$ is zero, see Lemma 5, hence the row sum of $\mathbb{M}_L + \Delta t_{k+1}\mathbb{L}$ equals the row sum of $\mathbb{M}_L$.

This approach is rather engineering because we don't have any mathematical model or theory which explains what the new scheme represents. However, the solution using this scheme possesses the desired properties, no undershoots and overshoots are visible and the solution is rather smeared. Also we avoid the major drawback of using the average values in the edge terms. The penalty term scheme transports the information as expected.

3.2 DG scheme with average values

The DG method possesses undesired undershoots and overshoots. The reason can be similar to the one in the low order scheme: the row sums of $\mathbb{M}_C + \Delta t_{k+1}\mathbb{A}$ do not equal to the row sums of $\mathbb{M}_C$. So in the linearized convective term b_h we modify the solution on the previous time step u^k such that we consider instead of u^k the average values $\overline{u_k}$ precisely as in subsection 3.1.1. The only difference is that we do not assemble the matrix $\mathbb{M}_{\mathbb{L}}$ and $\mathbb{L}$ and solve the problem

$$(\mathbb{M}_{\mathbb{C}} + \Delta t_{k+1}\mathbb{A})\, u^{k+1} = \mathbb{M}_{\mathbb{C}} u^k.$$

Note that if we consider the DG scheme defined in the previous chapter 2.13 then if we use u^k on the right hand side of the scheme or if we use $\overline{u^k}$, the schemes will be equivalent. The proof is simple and will not be presented.

The inequalities from Claim 2 are generally not satisfied, hence the solution on the next time level u^{k+1} does not satisfy local discrete maximum principle however the undershoots and overshoots are considerably diminished and the computational costs do not increase.

This method was developed as a by product of the above methods but for some test data it provides surprisingly good results. However, the method has the same drawback as the modified low order scheme 3.1.1. In some cases it stops the information propagation.

3.3 Algebraic flux correction

Define the antidiffusive flux as the difference between the DG scheme and the low order scheme

$$F(u^{k+1}) = (\mathbb{M}_L - \mathbb{M}_C)\left(u^{k+1} - u^k\right) + \Delta t_{k+1}(\mathbb{L} - \mathbb{A})u^{k+1}.$$

The idea is to decompose the antidiffusive fluxes $F(u)$ into numerical fluxes and limit the magnitude of these fluxes in the non-smooth regions. In this way the method can prevent possible violation of the DMP.

Using $\mathbb{L} - \mathbb{A} = \mathbb{D}$ and the zero row sum property of $\mathbb{D}$ (Lemma 5), we can define the *raw antidiffusive fluxes* F_{ij}, $i, j = 1, \dots, N$ in the following way

$$F_{ij} = m_{ij}(u_i^{k+1} - u_j^{k+1}) - m_{ij}(u_i^k - u_j^k) - \Delta t_{k+1} d_{ij}(u_i^{k+1} - u_j^{k+1}).$$

These fluxes describes the conservative mass exchange between a pair of nodes. Hence the defined fluxes satisfy $F_{ij} = -F_{ji}$. The proof of this property is immediately seen from the definition of matrices $\mathbb{M}_C, \mathbb{M}_L$ and $\mathbb{D}$.

Then for the i-th component of the vector $F(u^{k+1})$ it holds that

$$F_i = \sum_{j=1}^{N} F_{ij}.$$

We are now able to define a flux-corrected counterpart of (3.4)

$$(\mathbb{M}_L + \Delta t_{k+1}\mathbb{L})u_{k+1} = \mathbb{M}_L u^k + F^*(u^{k+1}, u^k), \tag{3.8}$$

where the nonlinear term $F^*(u^{k+1}, u^k)$ stands for the vector with i-th component equal to

$$F_i^*(u^{k+1}, u^k) = \sum_{j=1}^{N} \alpha_{ij} F_{ij}, \quad 0 \le \alpha_{ij} \le 1.$$

If $\alpha_{ij} = 1$ we obtain the original high order scheme and for $\alpha_{ij} = 0$ we compute using the stable low order scheme. A well-designed flux limiter will produce $F_i^*(u^{k+1}, u^k) = F_i$ in the smooth regions, i.e. we use the original high order scheme in the smooth regions, and $F_i^*(u^{k+1}, u^k) = 0$ elsewhere, i.e. the stable low order scheme is used in the areas where the high order scheme provides overshoots and undershoots.

3.4 Solving the flux-corrected problem

Suppose the weights α_{ij}, $i, j = 1, \dots, N$ are given.

We would like to reformulate the equation (3.8) as a system

$$\mathbb{P}(\alpha)u^{k+1} = G(\alpha),$$

where α is a matrix with elements α_{ij}, $i, j = 1, \dots, N$.

Note that α depends on the solution u^{k+1} or u^k but for keeping the notation as simple as possible, we will omit this dependence.

First we adjust the formula of row antidifussive fluxes using elements of matrix $\mathbb{L}$ and $\mathbb{A}$ instead of $\mathbb{D}$

$$\begin{aligned} F_i^*(u^{k+1}, u^k) = \sum_{j=1}^{N} \alpha_{ij} \Big(& m_{ij}(u_i^{k+1} - u_j^{k+1}) - m_{ij}(u_i^k - u_j^k) \\ & -\Delta t_{k+1}(l_{ij} - a_{ij})(u_i^{k+1} - u_j^{k+1}) \Big). \end{aligned}$$

Rearranging the terms

$$\begin{aligned} F_i^*(u^{k+1},u^k) = &-\sum_{j=1}^{N}\alpha_{ij}\left(m_{ij}u_j^{k+1}+\Delta t_{k+1}a_{ij}u_j^{k+1}\right)\\ &+\sum_{j=1}^{N}\alpha_{ij}m_{ij}u_j^{k}+\sum_{j=1}^{N}\Delta t_{k+1}a_{ij}u_i^{k+1}\\ &+u_i^{k+1}\sum_{j=1}^{N}\alpha_{ij}m_{ij}+\sum_{j=1}^{N}\Delta t_{k+1}\alpha_{ij}l_{ij}u_j^{k+1}\\ &-u_i^{k}\sum_{j=1}^{N}\alpha_{ij}m_{ij}-u_i^{k+1}\sum_{j=1}^{N}\Delta t_{k+1}\alpha_{ij}l_{ij}. \end{aligned}$$

The first part of the above expression corresponds to the high order scheme with weights α_{ij}, in the remaining terms there are terms that corresponds to the low order scheme scaled by α_{ij}. Hence in the matrix form

$$\begin{aligned} F^*(u^{k+1},u^k) = &-\left((\mathbb{M}_C+\Delta t_{k+1}\mathbb{A})\circ\alpha\right)u^{k+1}\\ &+(\mathbb{M}_C\circ\alpha)u^k+\Delta t_{k+1}(\mathbb{A}\circ\alpha)(1,\ldots,1)^Tu^{k+1}\\ &+(\mathbb{M}_C\circ\alpha)(1,\ldots,1)^Tu^{k+1}+\Delta t_{k+1}(\alpha\circ\mathbb{L})u^{k+1}\\ &-(\mathbb{M}_C\circ\alpha)(1,\ldots,1)^Tu^{k}-\Delta t_{k+1}(\alpha\circ\mathbb{L})(1,\ldots,1)^Tu^{k+1}. \end{aligned} \tag{3.9}$$

The operator $\mathbb{A}\circ\mathbb{B}$ denotes the *Hadamard product* also known as *Schur product* and it is defined as

$$(\mathbb{A}\circ\mathbb{B})_{i,j}=\mathbb{A}_{i,j}\mathbb{B}_{i,j}.$$

Remark. The term $(\mathbb{M}_C\circ\alpha)(1,\ldots,1)^T$ is an equivalent prescription for the diagonal matrix with the elements on the diagonal equal to

$$\sum_{j=1}^{N}\alpha_{ij}m_{ij},\quad i=1,\ldots,N.$$

We will denote this matrix as $\widetilde{\mathbb{M}}_L$.

Inserting the derived formula (3.9) into flux-corrected formula (3.8) we obtain the scheme

$$\begin{aligned} \Big[\mathbb{M}_L-\widetilde{\mathbb{M}}_L+\Delta t_{k+1}\mathbb{L}\circ(\mathbb{1}-\alpha)+(\mathbb{M}_C+\Delta t_{k+1}\mathbb{A})\circ\alpha-\Delta t_{k+1}(\mathbb{A}\circ\alpha)(1,\ldots,1)^T\\ +\Delta t_{k+1}(\mathbb{L}\circ\alpha)(1,\ldots,1)^T\Big]u^{k+1}=\Big[\mathbb{M}_L-\widetilde{\mathbb{M}}_L+\mathbb{M}_C\circ\alpha\Big]u^k. \end{aligned} \tag{3.10}$$

The symbol $\mathbb{1}$ denotes the *matrix of ones*, which is the matrix with all entries equal to one.

Note that all the properties of the FCT scheme mentioned in the previous section are still valid for the scheme (3.10). If all α_{ij} are zeros then $\widetilde{\mathbb{M}}_L$ is the zero matrix and we use the original high order scheme. On the other hand if all α_{ij} equal one, using the the zero row sum property of $\mathbb{A}$ and $\mathbb{L}$ together with the fact, that $\mathbb{M}_L=\widetilde{\mathbb{M}}_L$ produce the low order scheme 3.4.

3.5 Choice of α_{ij}

The crucial part of the algorithm is to provide reasonable values for the weights α_{ij}. For $F^*_{ij} := \alpha_{ij} Fij$ we need to preserve the mass conservation property on the numerical fluxes, i.e. we need

$$F^*_{ij} = -F^*_{ji}, \quad i, j = 1, \dots, N \tag{3.11}$$

to hold.

There are advanced methods of the choice of the weights α designed for FEM-FCT, one of the algorithms is described f.e. in Zalesak [1979]. The major difference between FEM-FCT and FCT method derived from the DG scheme, further only DG-FCT, is that for DG-FCT method the solution in less influenced by the choice of the limiters thus we should provide as simple methods as possible for the choice of weights. In this work we limit ourself only to the case when α_{ij} attains the value either zero or one.

We reformulate the problem of the choice of α_{ij} to the problem of selecting the intervals (x_i, x_{i+1}) on which we compute the solution using the low-order scheme. By default we use the high order scheme, in terms of α_{ij} it means to set all weights equal to one. Then we select the i-th element as the element on which we would like to compute the low-order scheme. Then we set $\alpha_{ij},\ j = 1, \dots, N$ and $\alpha_{ji},\ j = 1, \dots, N$ equal to zero. The resulting α matrix is symmetric and the condition (3.11) is then satisfied.

We present several approaches how to choose the elements on which we use the low order scheme.

3.5.1 Minmod limiting

The idea behind the Minmod criterion of the choice of elements is based on the estimates of undershoots and overshoots. We only briefly introduce the method here, for further details refer to Shu [2009].

Let us define interval average over the element (x_i, x_{i+1})

$$\bar{u_i} = \frac{1}{h} \int_{x_i}^{x_{i+1}} u^k \, dx.$$

Further compute the differences between the end values and the average

$$\begin{aligned} \tilde{u_i} &= u^{k+1}(x^-_{i+1}) - \bar{u_i}, \\ \tilde{\tilde{u_i}} &= \bar{u_i} - u^{k+1}(x^+_i). \end{aligned}$$

There are two natural conditions required from this limiting strategy. First, it should not change the cell averages. This is the conservation property of the DG method. Second, it should not affect the accuracy of the scheme in the smooth regions. In other words, in the smooth regions this limiter does not change the solution.

To this point define

$$\begin{aligned} \tilde{u_i}^{(mod)} &= m\left(\tilde{u_i}, \Delta_+ \bar{u_i}, \Delta_- \bar{u_i}\right), \\ \tilde{\tilde{u_i}}^{(mod)} &= m\left(\tilde{\tilde{u_i}}, \Delta_+ \bar{u_i}, \Delta_- \bar{u_i}\right), \end{aligned}$$

where

$$\begin{aligned}\Delta_+\bar{u}_i &= \bar{u_{i+1}} - \bar{u}_i,\\ \Delta_-\bar{u}_i &= \bar{u}_i - \bar{u_{i-1}}.\end{aligned}$$

Here the minmod function m is defined by

$$m(a_1,\ldots,a_l) = \begin{cases} s\min\left(|a_1|,\ldots,|a_l|\right), & \text{if } s = sign(a_1) = \cdots = sign(a_l);\\ 0, & \text{otherwise.}\end{cases}$$

Finally we are able to define the modified values u_h^{mod}

$$\begin{aligned}u_h^{mod}(x_{i+1}^-) &= \bar{u}_i + \tilde{u_i}^{(mod)},\\ u_h^{mod}(x_i^+) &= \bar{u}_i - \tilde{\tilde{u}}_i^{(mod)}.\end{aligned}$$

If the solution u_h possesses overshoots and undershoots on some interval (x_i, x_{i+1}) then the limited function the absolute values $u_h^{(mod)}$ are smaller in magnitude that the original values. Moreover the old cell average values are maintained.

In our algorithm we use the criterion from the minmod limiter to identify elements on which to use the low order scheme.

Algorithm 1 Minmod limiting

1: **if** $\left|(u_h^{(mod)}(x_i^+))\right| > \left|(u_h(x_i^+))\right|$ or $\left|(u_h^{(mod)}(x_{i+1}^-))\right| > \left|(u_h(x_{i+1}^-))\right|$ **then**
2: use low order scheme on the interval (x_i, x_{i+1})
3: **else**
4: use high order scheme on the interval (x_i, x_{i+1}).
5: **end if**

It can proved for an explicit scheme that the presented limiting strategy does not affect accuracy in smooth monotone regions. For the proof see Shu [2009]. The limiter though does kill accuracy at smooth extremes. At a maximum of the solution u_h, the second and third argument of the minmod function $m\left(\tilde{u_i}, \Delta_+\bar{u}_i, \Delta_-\bar{u}_i\right)$ have different sign, hence $\tilde{u_i}^{(mod)}$ and $\tilde{\tilde{u}}_i^{(mod)}$ are zero and the modified values are

$$u_h^{mod}(x_{i+1}^-) = \bar{u}_i, \quad u_h^{mod}(x_i^+) = \bar{u}_i.$$

Therefore in practice we often use a corrected minmod limiter

$$\tilde{m}(a_1,\ldots,a_l) = \begin{cases} a_1, & \text{if } |(a_1)| \le Mh^2;\\ m(a_1,\ldots,a_l), & \text{otherwise.}\end{cases}$$

The parameter M has to be chosen adequately, often dependently on the solving equation or given data. If we lack any information about solving problem, suitable values are between 10 and 40, [see Shu, 2009].

3.5.2 Magnitude of jumps at nodal points

Minmod limiting can capture nicely overshoots and undershoots but another possibility is to check the magnitude of the difference between $u_h(x_i^-)$ and $u_h(x_i^+)$ at each node x_i.

The idea is simple, we can expect "bad" behavior at discontinuities, where large jumps in the solution occur and hence we use the low order scheme around these points. We do not test if there are overshoots and undershoots around the discontinuity in the sense of the minmod limiter, where also the signs of the jumps are taken into account. We use the low order scheme around every discontinuity even though no overshoots and undershoots arise at these points.

The biggest advantage of this method is the simplicity of implementation as well as its cheapness, as low memory cost as low computation time.

Algorithm 2 Magnitude of jumps

1: **if** $\left|(u_h(x_i^+) - u_h(x_i^-))\right| + \left|(u_h(x_{i+1}^+) - u_h(x_{i+1}^-))\right| > Ch$ **then**
2: use low order scheme on the interval (x_i, x_{i+1})
3: **else**
4: use high order scheme on the interval (x_i, x_{i+1}).
5: **end if**

The constant C is, similarly to constant M in minmod limiting, some number often dependent on the given problem. By trying different constants and evaluating the results the adequate constant for Burgers' equation was approximately 2.

3.5.3 Magnitude of derivatives

Let us consider the initial condition to be a smooth function. As the solution evolves in time, a discontinuity begins to emerge. For example if $\sin(2\pi x)$ is prescribed for the initial condition and we solve Burgers' equation, the solution moves to the right in the left half of the interval $(0, 1)$ and the the solution moves left in the second part of the interval. Hence in the midpoint a discontinuity appears.

The first two introduced strategies select the elements around the discontinuity with delay. Meaning that the choice of intervals react on the overshoots and undershoots or jumps of the solution. If the solution is smooth or the jumps are not too big, no elements are selected by the previous strategies. The solution must provide some "misbehavior" in order to identify the bad elements. The delay depends on the given tolerance, chosen constants and also on precision of computations. We have observed that the delay, however, may possibly cause severe overshoots and undershoots of the FCT solution, especially on coarse grids.

For this reason another criterion is introduced

Algorithm 3 Magnitude of derivatives

1: **if** $\left|u_h(x_{i+1}^-) - u_h(x_i^+)\right| > C_2$ **then**
2: use low order scheme on the interval (x_i, x_{i+1})
3: **else**
4: use high order scheme on the interval (x_i, x_{i+1}).
5: **end if**

This strategy can be useful if we have information about the derivative of the solution. We can obtain a useful estimate of constant C_2 from the initial and boundary conditions using the fact that DMP is satisfied, i.e. no new extremes arise inside the interval. Suppose that the extremes are attained in the initial conditions u_0, therefore we choose

$$C_2 = \max(u_0) - \min(u_0).$$

If new extremes arise from boundary conditions, the constant C_2 should be updated accordingly.

The advantage of this interval selecting method is again its simplicity. Moreover it works well in case the initial condition is continuous. The major drawback is that we need the information about the derivative.

4. Numerical results

We will study the behavior of the presented methods solving Burgers' equation

$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} = 0, \tag{4.1}$$

1. equipped with the initial condition

$$u_0(x) = \sin(2\pi x) \tag{4.2}$$

and the boundary conditions prescribed as

$$u_D(0) := 0, \quad u_D(1) := 0. \tag{4.3}$$

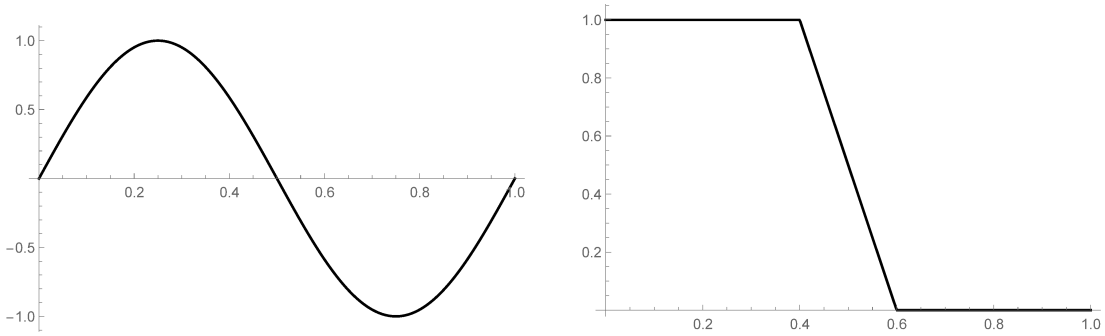
2. Or using the initial condition

$$u_0(x) = \begin{cases} 1, & \text{if } \ 0 < x < 0.4, \\ 5(0.6 - x) & \text{if } \ 0.4 \leq x < 0.6 \\ 0, & \text{if } \ 0.6 \leq x \leq 1. \end{cases} \tag{4.4}$$

and the boundary condition

$$u_D(0) := 1.$$

We do not need to define the boundary condition in point 1, because it is the outlet boundary point for (4.4).



(a) Initial condition $\sin(2\pi x)$. (b) Initial condition (4.4).

Figure 4.1: Prescribed initial conditions for Burgers' equation.

What should the solution look like?

For the second initial condition (4.4) we know that the value 1 is transported to the right by the velocity equal to 1. Hence after some time, the solution is a constant function with function value equal to one. The analytic solution is discussed in the book Feistauer et al. [2003] for the Burgers' equation with the convective term multiplied by the factor 1/2.

Concerning the initial condition equal to $\sin(2\pi x)$, we are able to say that the solution on $(0, 0.5)$ moves from left to right and the solution on $(0.5, 1)$ moves

from right to left. Hence we can say that the two parts of the solution moves against each other and we expect the discontinuity to arise at the point 0.5. The velocity of propagation is biggest at the areas where the sine function attains its extremes. After some time, these extremes cross the point 0.5 and are no longer visible, hence the extremes of the solution decrease with time and also the velocity of propagation around point 0.5 decreases.

However, as it was proven in Theorem 1, Burgers' equation satisfies the discrete maximum principle, hence no new extremes should arise in the interior of the domain. If the numerical solution has overshoots and undershoots, it is a sign that the used scheme does not provide a correct solution.

4.1 DG

Let us present the results using the DG method for both initial conditions. We divide the interval $(0, 1)$ into 501 sub-intervals. The time step is chosen to be 0.001 and the solution for the first initial condition is plotted at time $0.08, 0.1, 0.3$ and 1, respectively. We can observe considerable overshoots and undershoots, Figure 4.2. The same is true for the second initial condition (4.4), the solution plotted at time $0.1, 0.2, 0.5$ and 1, Figure 4.3. For time greater then 0.5 we have the solution constant 1, thus the overshoots and undershoots are not visible.

Note that we intentionally choose an odd number of sub-intervals so that there is an element with the rising discontinuity in the interior of this element and thus we can observe overshoots and undershoots. If we have an even number of sub-intervals, for selected initial conditions no undershoots and overshoots are formed.

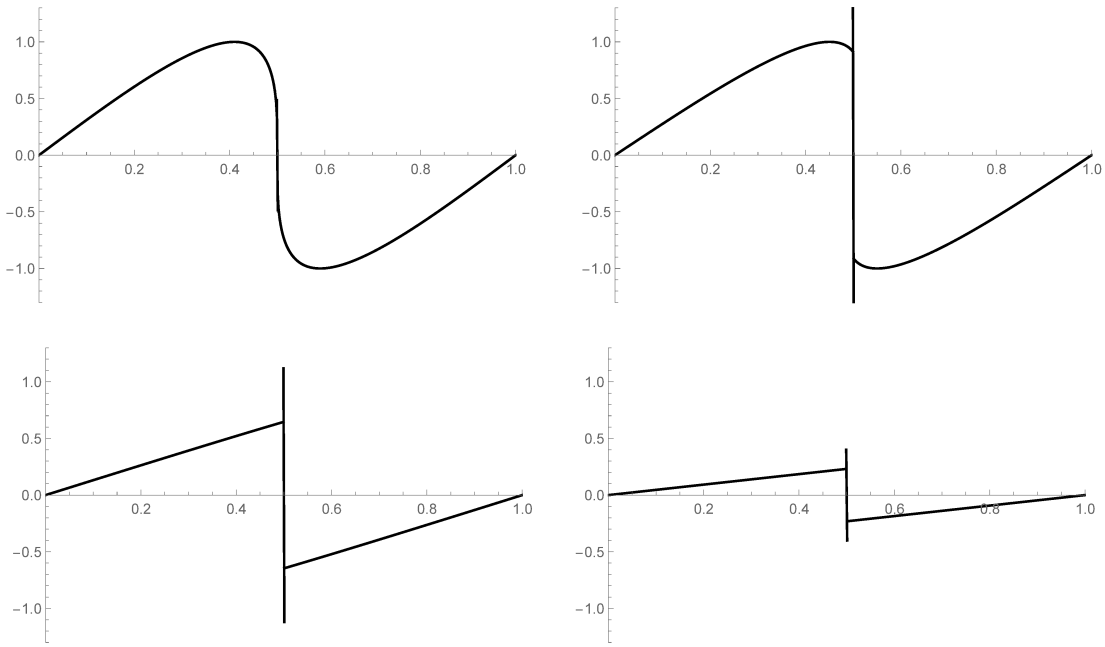


Figure 4.2: Burgers' equation with initial condition $\sin(2\pi x)$ computed by the DG method, grid with 501 elements, time step set to 0.001, figures show the solution at time $0.08, 0.1, 0.3$ and 1, respectively.

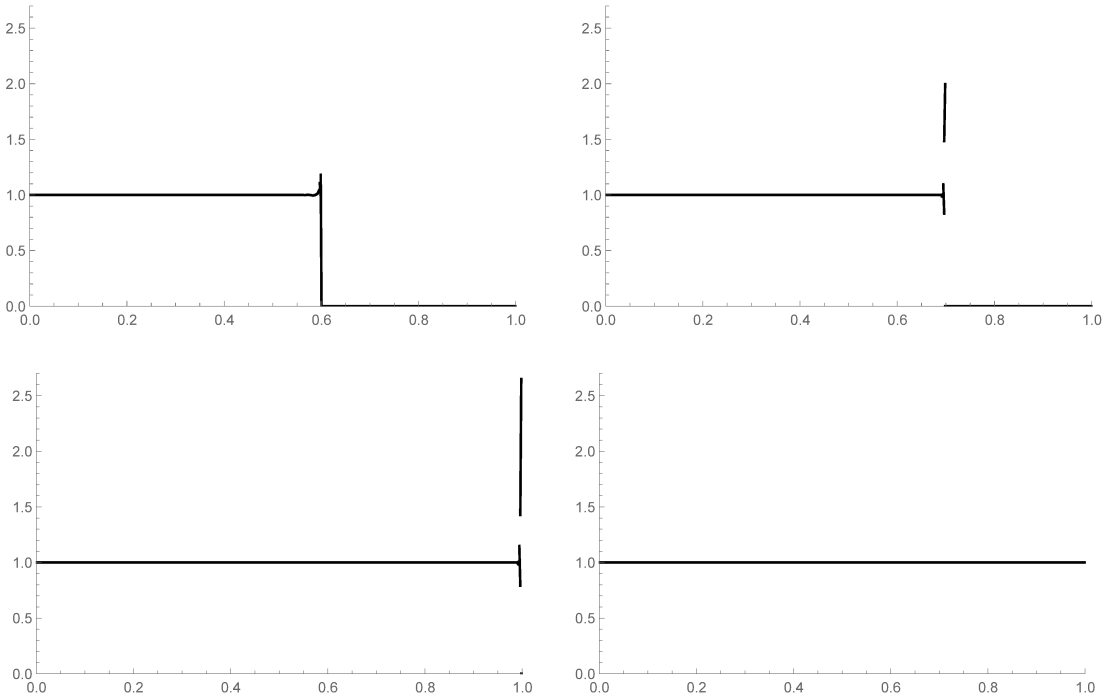


Figure 4.3: Burgers' equation with initial condition (4.4) computed by the DG method, grid with 501 elements, time step set to 0.001, figures show the solution at time $0.1, 0.2, 0.5$ and 1, respectively.

4.2 LOS

First we consider the low order method with modification of the solution on the previous time step as described in Section 3.1.1. Let the initial condition be $\sin(2\pi x)$, time step 0.001. The solution computed on the coarse grid using 19 elements is visibly not continuous anywhere, we can observe "stairs" or "terraces". It is a property of low order scheme, the slopes of linear function on each subinterval are smeared, Figure 4.4.

The results of the same problem, i.e. initial condition $\sin(2\pi x)$, time step 0.001, provided on fine grid with 501 elements gives surprisingly good results. The terracing effect is not visible, the solution looks like it is smooth, Figure 4.5.

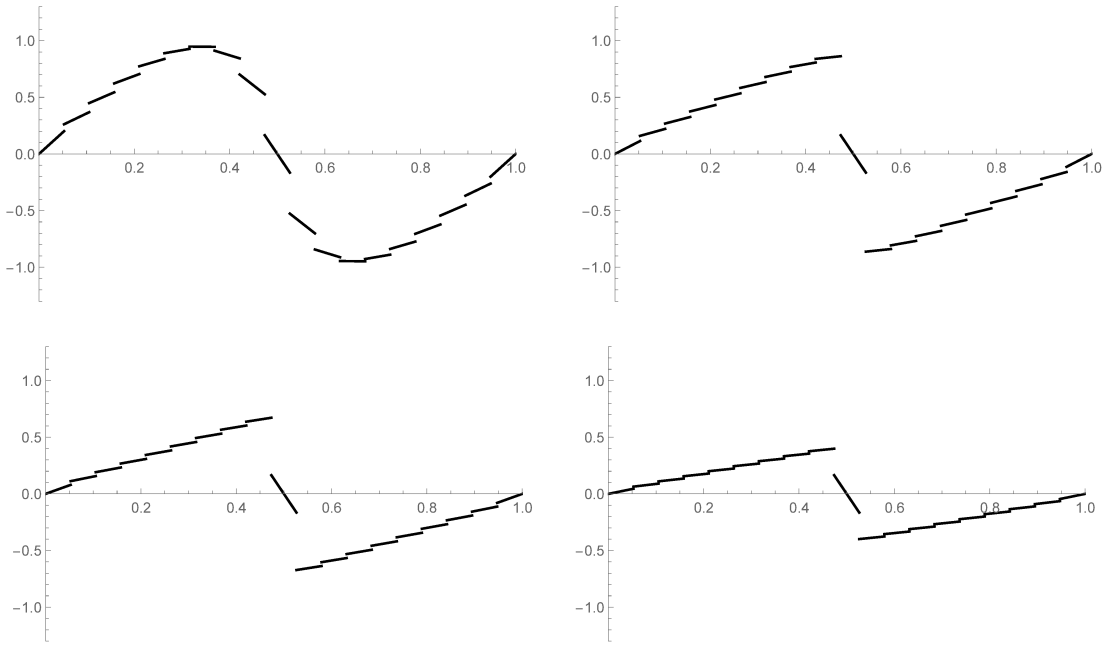


Figure 4.4: Burgers' equation with initial condition $\sin(2\pi x)$ computed by the low order method with modification of the solution on previous time step, grid with 19 elements, time step set to 0.001, figures show the solution at time $0.1, 0.3, 0.5$ and 1, respectively.

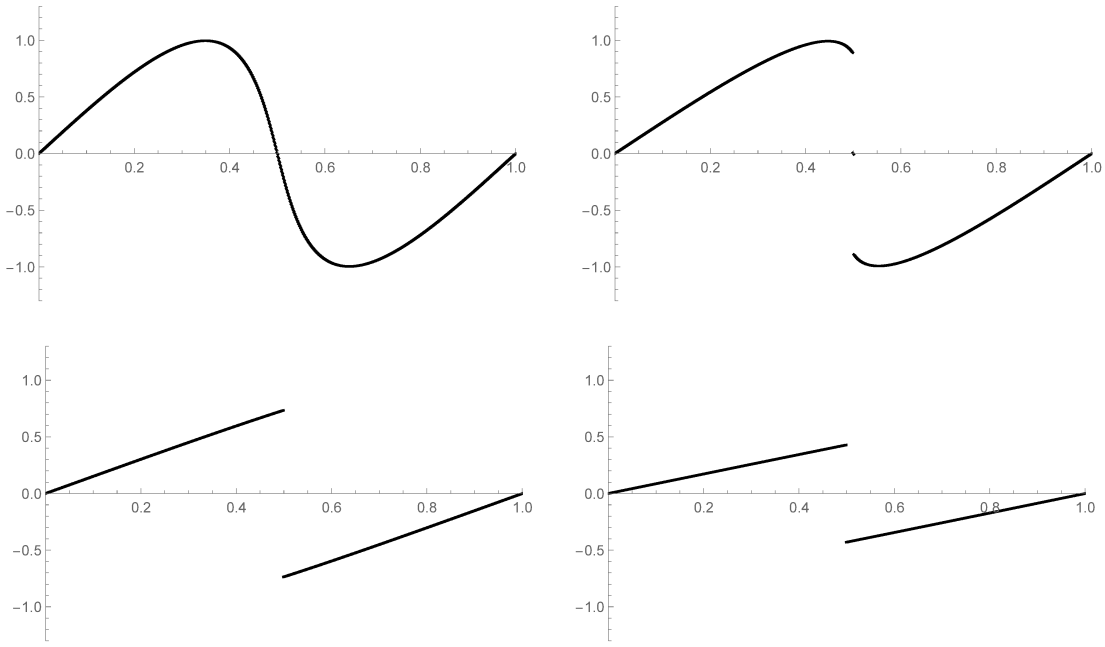


Figure 4.5: Burgers' equation with initial condition $\sin(2\pi x)$ computed by the low order method with modification of the solution on previous time step, grid with 501 elements, time step set to 0.001, figures show the solution at time $0.1, 0.2, 0.5$ and 1, respectively.

The problem of this method is that the solution of Burgers' equation with initial condition (4.4) is wrong. The information is not propagated from left to right correctly because of the limited communication between neighboring elements as explained in the subsection 3.1.1. Provided results were computed on the grid consisting of 501 elements, with time step 0.001, the plots capture the solution at time $0.08, 0.16, 0.2$ and 0.3, the solution does not evolve after time 0.3. The solution becomes stationary, piecewise constant, 1 on $(0, 0.6)$ and 0 elsewhere, see Figure 4.6.

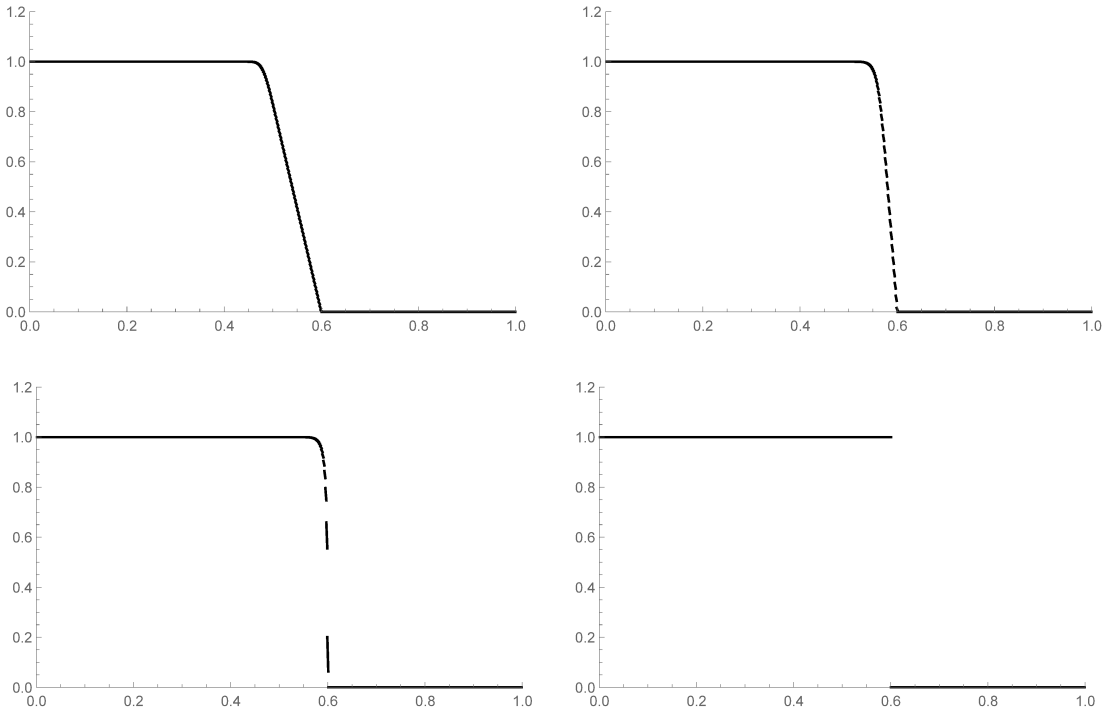


Figure 4.6: Burgers' equation with initial condition (4.4) computed by the low order method with modification of the solution on previous time step, grid with 501 elements, time step set to 0.001, figures show the solution at time $0.08, 0.16, 0.2$ and 0.3, respectively.

We will now consider the low order method with enforcing row sum condition (3.2) by penalty terms. As was suggested in the subsection 3.1.2, the method works well, the provided solution satisfies the local discrete maximum principle. With a coarse grid we can also observe terracing effect. We will provide results for both initial conditions, first $\sin(2\pi x)$ - Figure 4.7, then (4.4) - Figure 4.8, the time step remains the same 0.001, the number of elements is 501.

In the Figure 4.8 we can observe that the solution moves from left to right but the velocity is smaller then in the DG solution. We cannot explain satisfactorily why there is a delay in propagation of the solution nor can we answer what the velocity of propagation in this scheme is. The same behavior is observed in Figure 4.7 for the initial condition $sin(2\pi x)$ but the delay is not so apparent. The comparison of the low order scheme with penalty terms and the DG solution for the initial condition (4.4) is in Figure 4.9

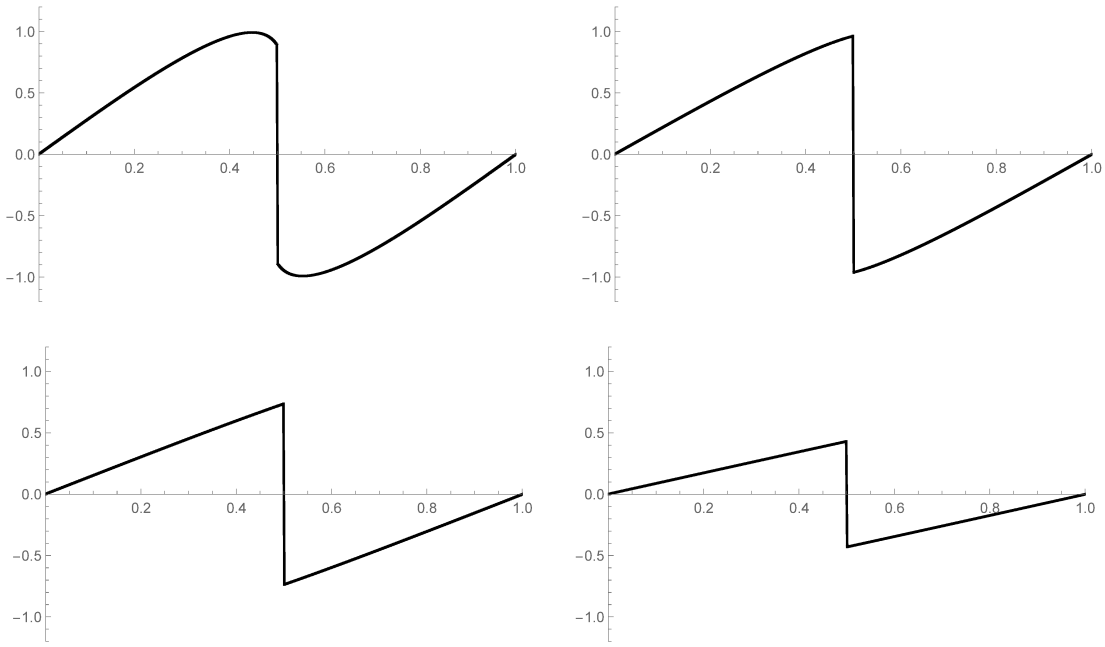


Figure 4.7: Burgers' equation with initial condition $\sin(2\pi x)$ computed by the low order method with penalty terms, grid with 501 elements, time step set to 0.001, figures show the solution at time $0.2, 0.3, 0.5$ and 1, respectively.

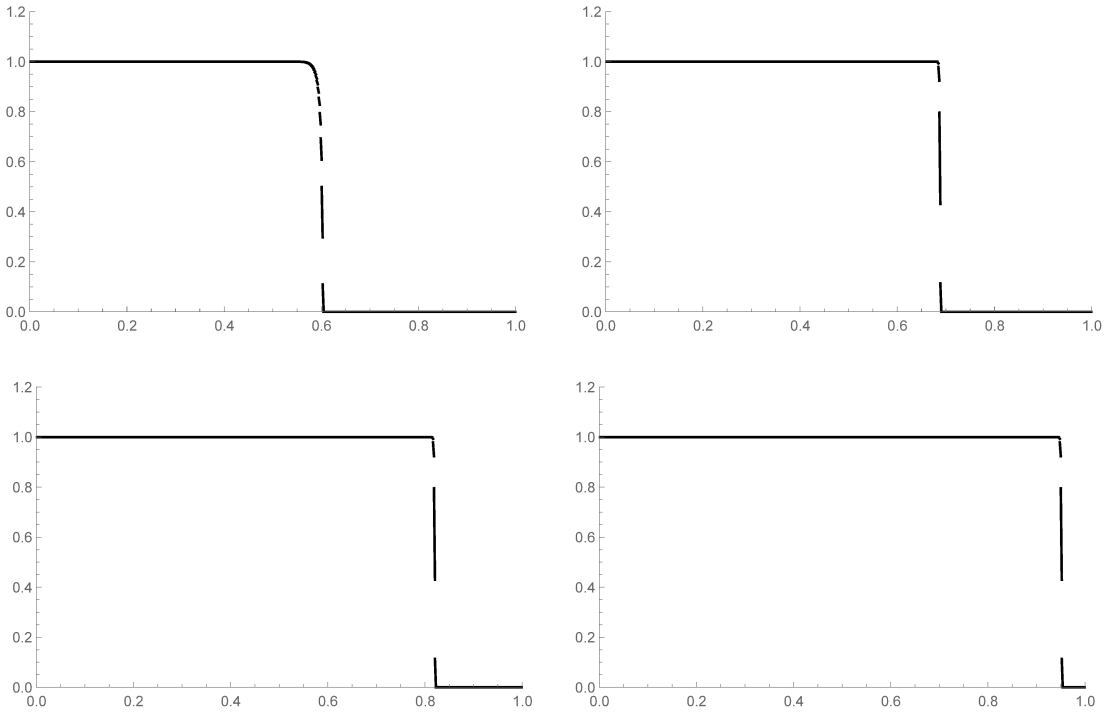


Figure 4.8: Burgers' equation with initial condition (4.4) computed by the low order method with penalty terms, grid with 501 elements, time step set to 0.001, figures show the solution at time $0.2, 0.4, 0.7$ and 1, respectively.

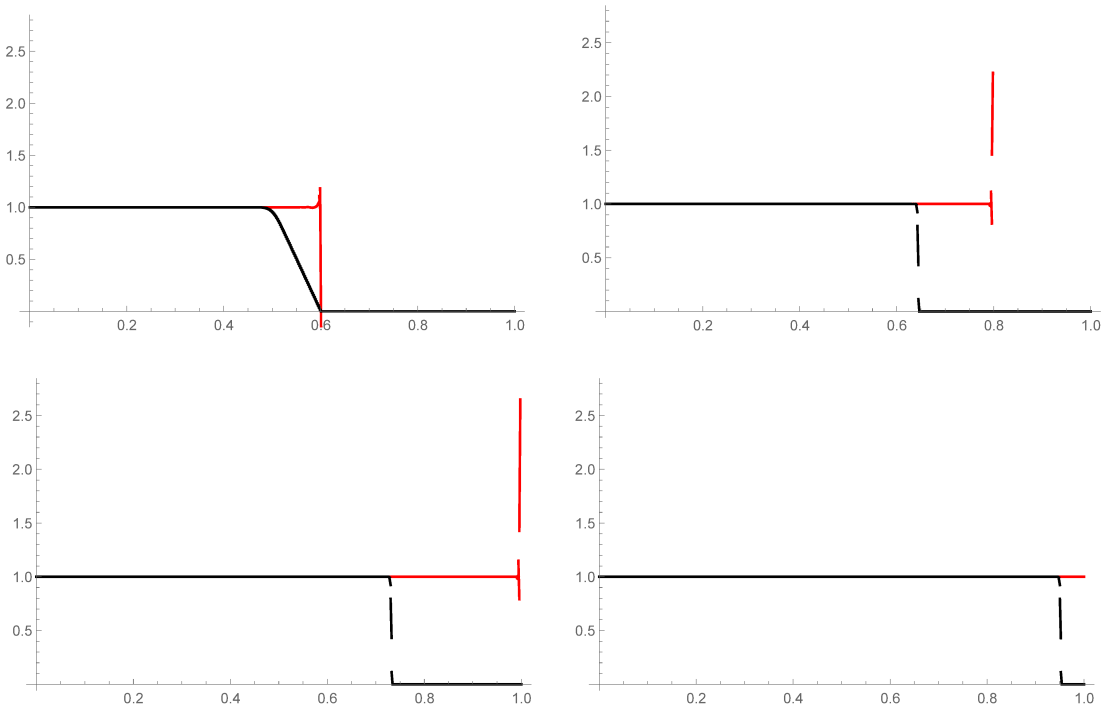


Figure 4.9: Burgers' equation with initial condition (4.4) computed by the low order method with penalty terms compared with solution obtained by using DG method, grid with 501 elements, time step set to 0.001, figures show the solution at time $0.1, 0.3, 0.5$ and 1, respectively.

4.3 DG scheme with average values

We introduce also the results for the DG method which uses average values as described in Section 3.2. As it was mentioned, the results obtained by this method are similar to the results computed by the low order method with penalty terms. The advantages of the method are that the solution shows only small overshoots and undershoots if any and we do not need to assemble matrices $\mathbb{M}_L$ and $\mathbb{L}$, hence it is as expensive as the DG method measured by the number of operations. The major disadvantages are that the information does not propagate correctly for the second initial condition 4.4 and the local DMP need not be satisfied. In the Figure 4.10 there is the solution for the first initial condition 4.2. In the Figure 4.11 there it is the solution for the second initial condition 4.4, the solution does not evolve after time 0.3.

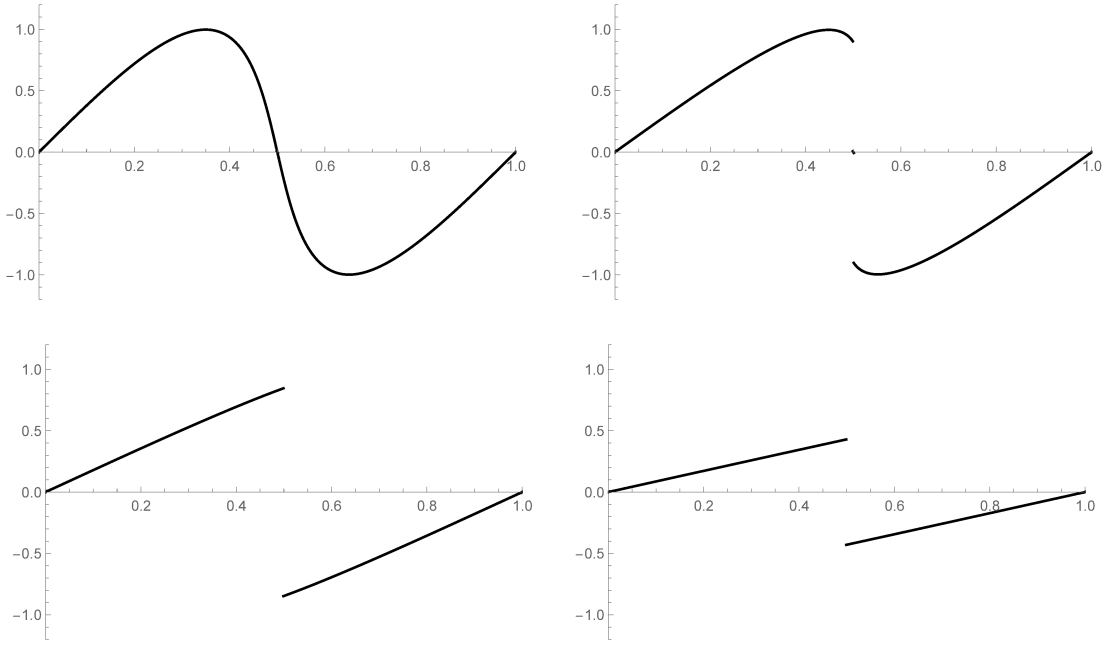


Figure 4.10: Burgers' equation with initial condition $\sin(2\pi x)$ computed by the DG method with average values, grid with 501 elements, time step set to 0.001, figures show the solution at time $0.1, 0.2, 0.4$ and 1, respectively.

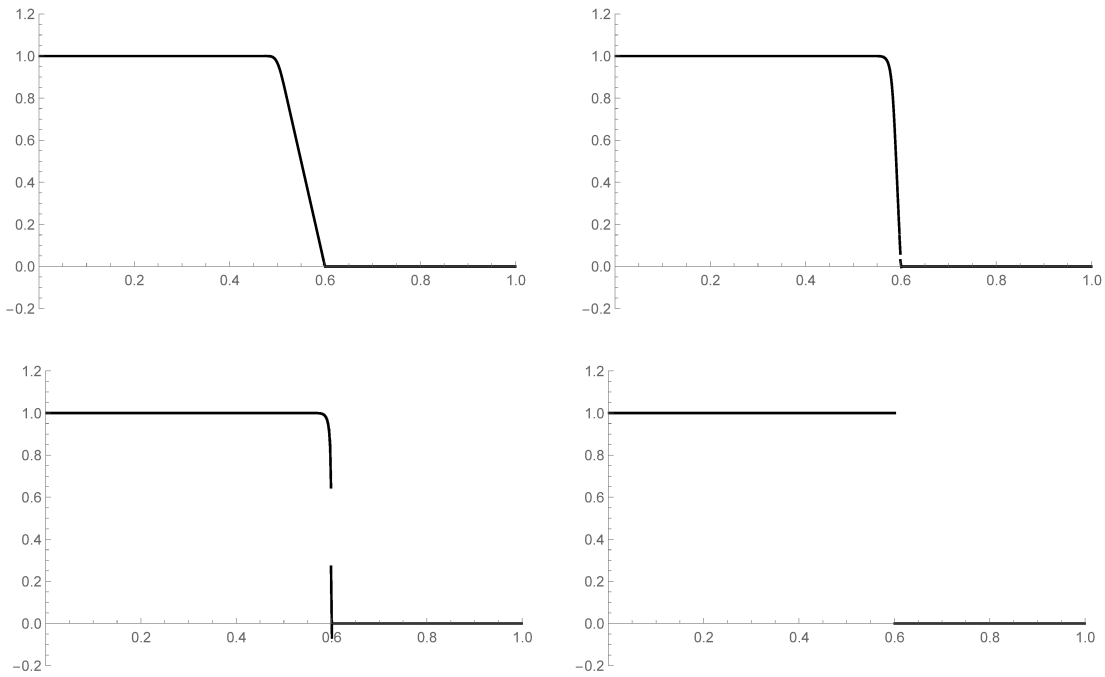


Figure 4.11: Burgers' equation with initial condition 4.4 computed by the DG method with average values, grid with 501 elements, time step set to 0.001, figures show the solution at time $0.1, 0.18, 0.2$ and 0.3, respectively.

4.4 FCT

This method is a combination of the low order scheme with the DG method. As it was mentioned in Section 3.3, the method depends on the choice of the weights α_{ij}. Hence, before we proceed to the presentation of individual approaches of the choice of α_{ij}, we show the results computed by the FCT method for the first initial condition 4.2 provided the weights are set manually with the knowledge of where the discontinuity arises. Let $\sin(2\pi x)$ be the initial condition, the number of elements is set to 39. The discontinuity is located at the element number 20. First, we use the low order scheme only at the 20-th interval, Figure 4.12. Second we set the low order scheme at intervals $18, 19, 20, 21$ and 22, Figure 4.13. The second solution is worse than the first one because the solution at elements $18, 19, 21$ and 22 has larger jumps. However, Figure 4.13 well-illustrates the behavior of the FCT method and how important is the choice of the weights α_{ij}.

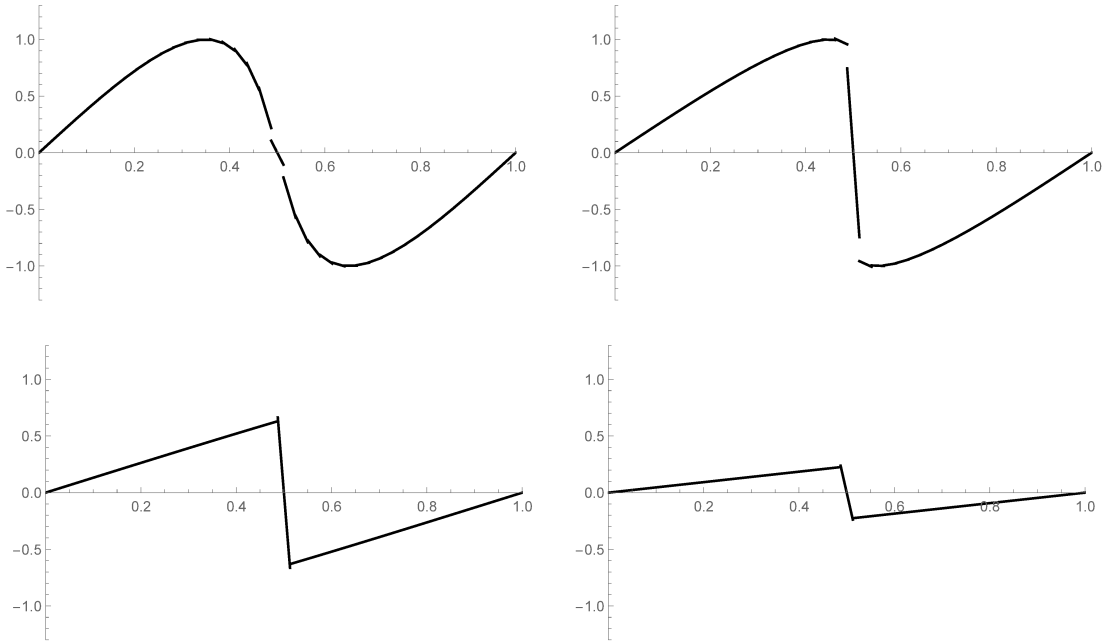


Figure 4.12: Burgers' equation with initial condition $\sin(2\pi x)$ computed by the FCT method on the grid with 39 elements. The low order scheme with penalty terms is used on the 20-th interval. Time step set to 0.001, figures show the solution at time $0.05, 0.1, 0.3$ and 1, respectively.

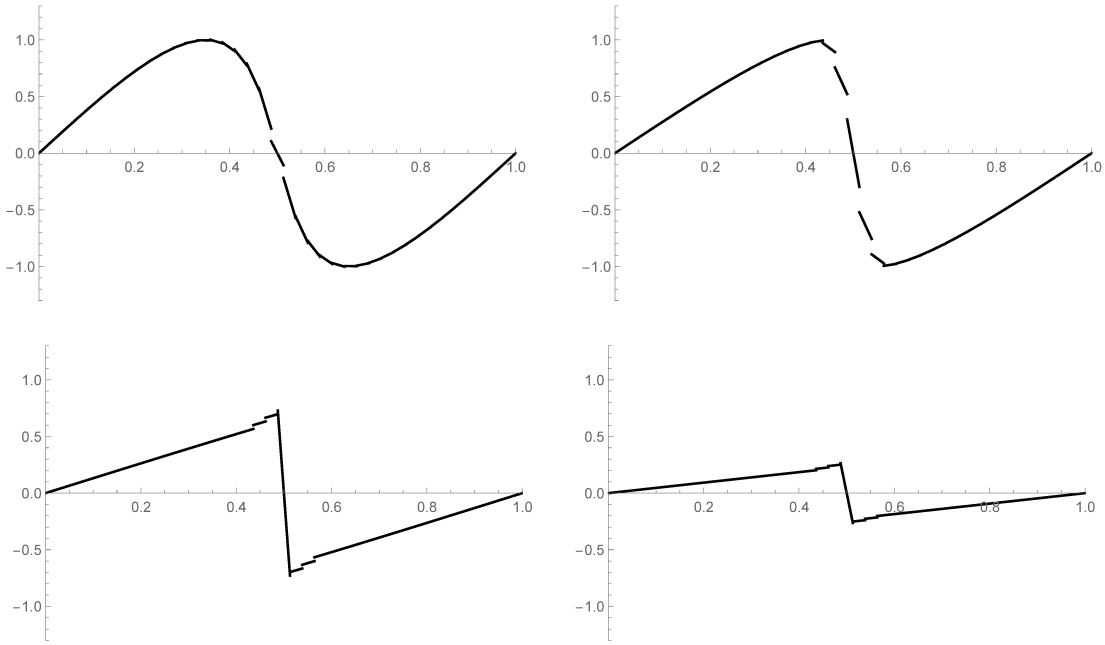


Figure 4.13: Burgers' equation with initial condition $\sin(2\pi x)$ computed by the FCT method on the grid with 39 elements. The low order scheme with penalty terms is used on the intervals $18, 19, 20, 21, 22$. Time step set to 0.001, figures show the solution at time $0.05, 0.1, 0.3$ and 1, respectively.

We can observe that the magnitude of overshoots and undershoots depends on the norm of the partition. For coarse grids we have larger overshoots and undershoots, for fine grids the overshoots and undershoots are less invisible, see Figure 4.14. But for fine grids we encounter the difficulty of finite precision computations. Therefore we focus on presenting the results on fine grids. However at the end of this section we also show selected solutions on coarse grids and we discuss the choice of time step.

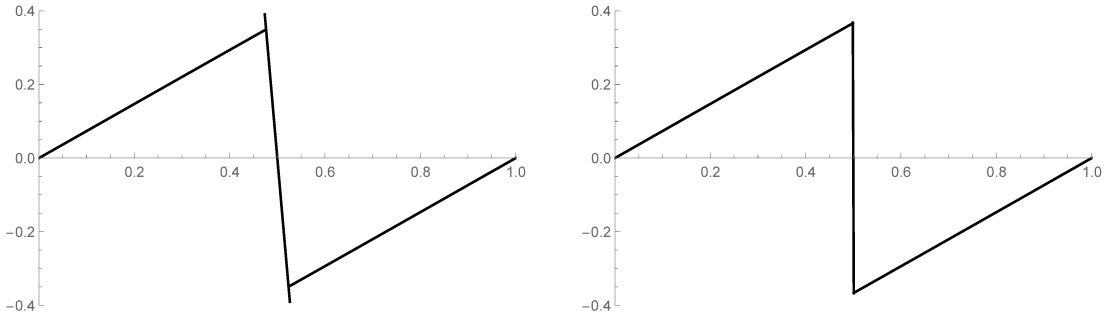


Figure 4.14: Burgers' equation with initial condition $\sin(2\pi x)$ computed by the FCT method. The solution in the left figure uses the grid with 19 elements, in the right figure there is a grid with 501 elements. The low order scheme with penalty terms is used on the interval 20. Time step set to 0.001, figures show the solution at time 0.6.

We now progress to the presentation of the individual strategies how to choose the weights α_{ij} described in Section 3.5. Note, that the FCT method uses a low order method as part of the algorithm, hence for each strategy of the choice of weights we need to specify what low order method was used.

4.4.1 Minmod limiting

Let us choose the parameter M from Section 3.5.1 to be 40. We start with the FCT method using the low order scheme with modification of the solution on the previous time level. For the initial condition $\sin(2\pi x)$ the solution does not show large overshoots and undershoots, on the middle element it has small derivative but the average over the interval is not zero as expected. This is well seen at time 0.8 and 1, where the average over the interval changes the sign and we can observe small overshoots and undershoots. This behavior is caused by the non symmetric choice of the intervals, where the low order scheme is applied. More precisely, the solution is symmetric around point 0.5, denote the middle interval by m, then if the method chooses the interval $m - i$, it should choose also the interval $m + i$. In practice, due to rounding errors, the method sometimes chooses the intervals nonsymmetrically thus the computed solution on the next time level looses the symmetric property. If the symmetry is corrupted, the Minmod limiting method increases the number of selected nonsymmetric intervals in each time step and the error accumulates which may results in the behavior presented in Figure 4.15.

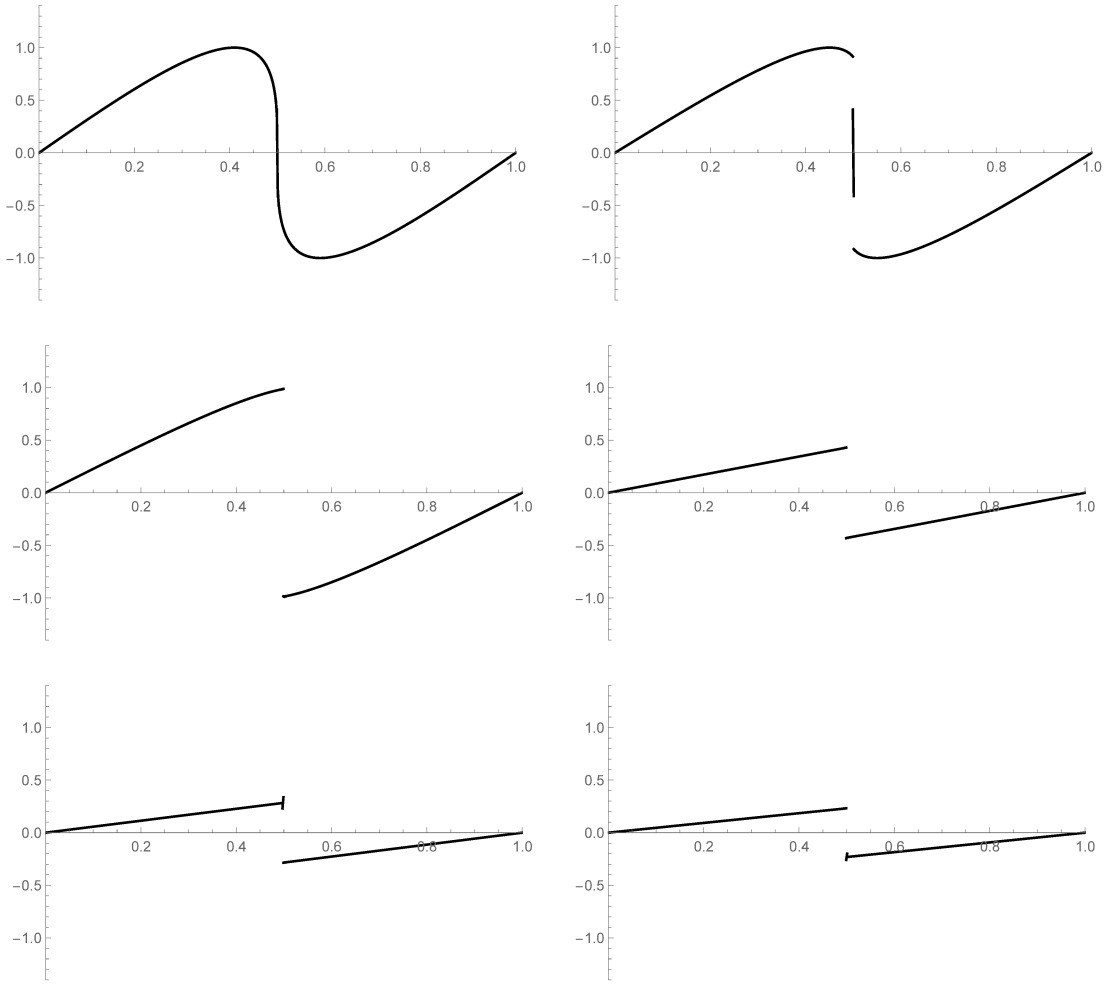


Figure 4.15: Burgers' equation with initial condition $\sin(2\pi x)$ computed by the FCT method using minmod limiting on the grid with 501 elements. The low order scheme with modification of the solution on previous level. Time step set to 0.001, figures show the solution at time $0.08, 0.1, 0.14, 0.5, 0.8$ and 1, respectively.

For the initial condition (4.4) we observe the same problem as for the low order scheme with modification of the solution on the previous time step. The solution stabilizes after some time as 1 to the interval $(0, 0.6)$ and zero elsewhere instead of transporting the value 1 on the whole interval $(0, 1)$, see Figure 4.16.

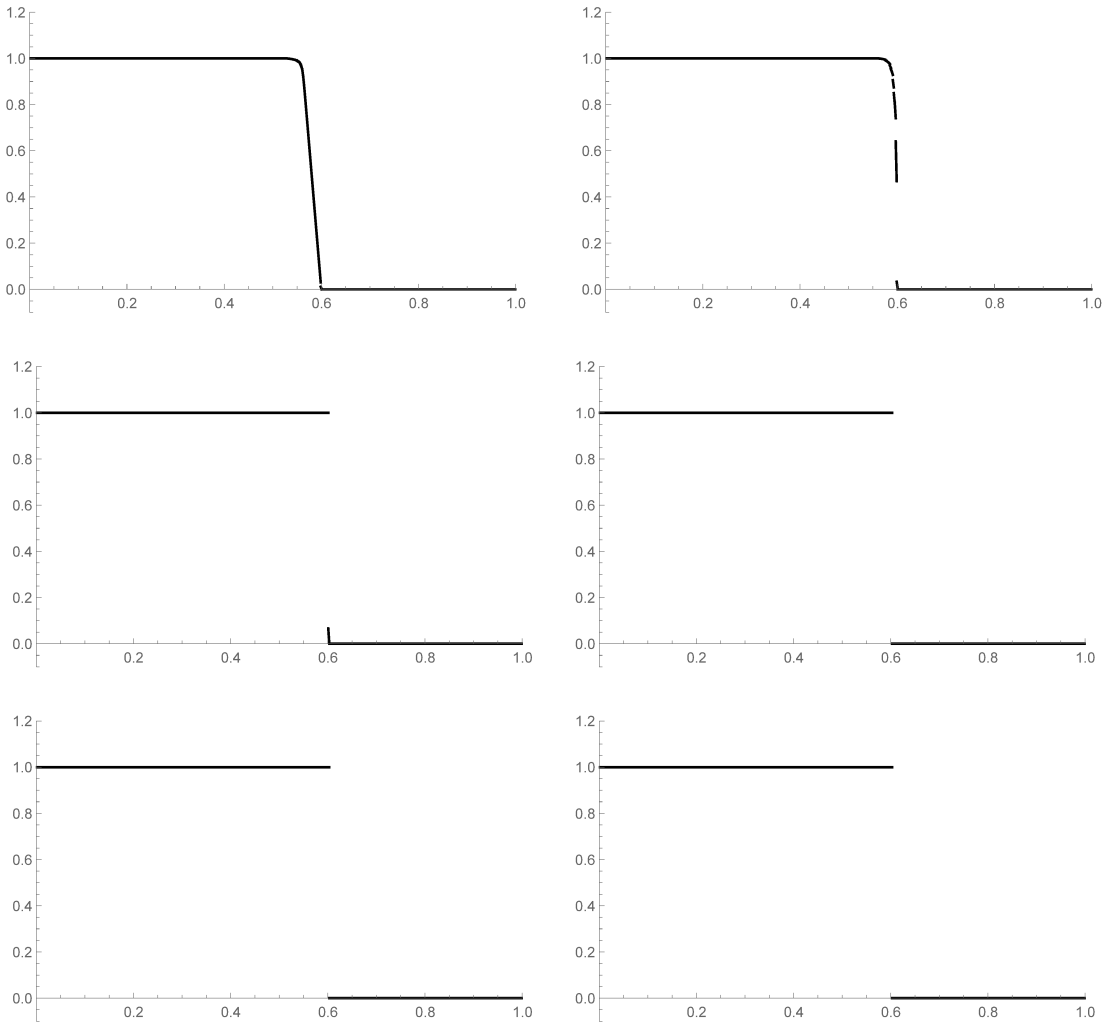


Figure 4.16: Burgers' equation with initial condition (4.4) computed by the FCT method using minmod limiting on the grid with 501 elements. The low order scheme with modification of the solution on previous level. Time step set to 0.001, figures show the solution at time $0.08, 0.1, 0.14, 0.2, 0.5$ and 1, respectively.

Consider now the FCT method using penalty terms. For the initial condition $\sin(2\pi x)$ the computed solution is the best from all presented solutions. The overshoots and undershoots, if any, are not visible. The solution looks continuous, Figure 4.17.

For the initial condition (4.4) the computed solution is also good, see Figure 4.18. No undershoots and overshoots are visible, the signal is transported from left to right, hence after some time the value 1 is extended on the whole interval $(0, 1)$. The major drawback is the velocity of signal propagation. In comparison to the DG method, in the FCT method the signal is transported about half of the speed of the DG method.

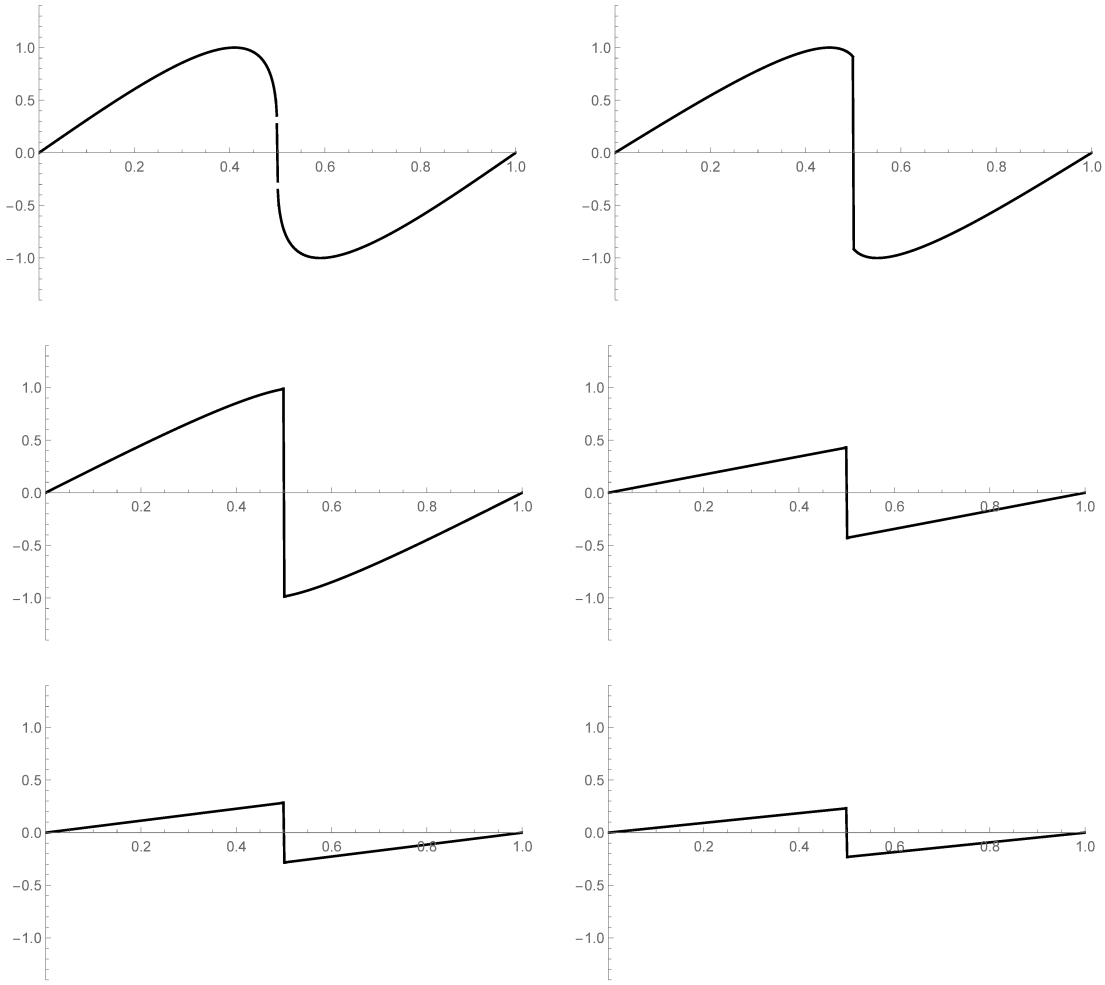


Figure 4.17: Burgers' equation with initial condition $\sin(2\pi x)$ computed by the FCT method using minmod limiting on the grid with 501 elements. The low order scheme with penalty terms. Time step set to 0.001, figures show the solution at time $0.08, 0.1, 0.14, 0.5, 0.8$ and 1, respectively.

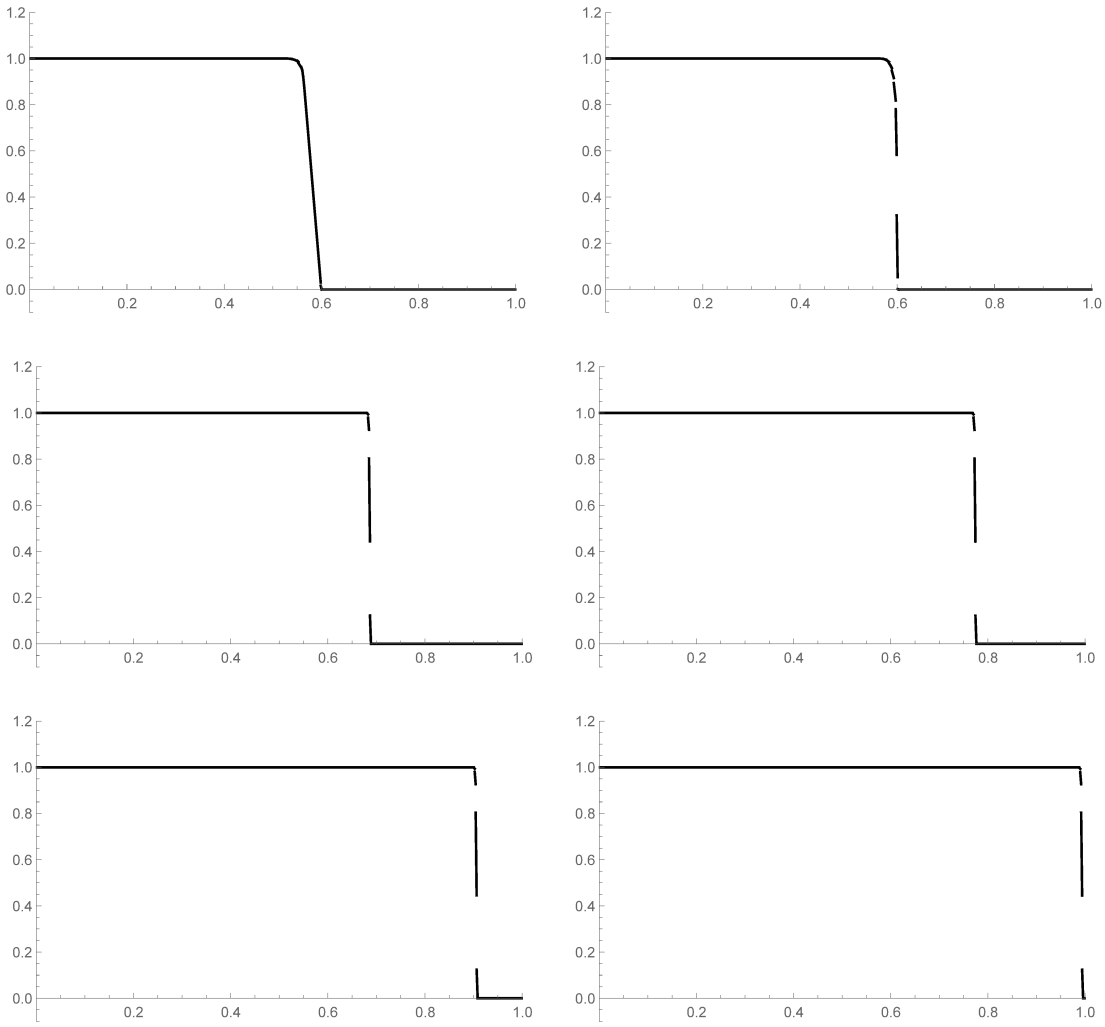


Figure 4.18: Burgers' equation with initial condition (4.4) computed by the FCT method using minmod limiting on the grid with 501 elements. The low order scheme with penalty terms. Time step set to 0.001, figures show the solution at time $0.08, 0.1, 0.3, 0.5, 0.8$ and 1, respectively.

4.4.2 Magnitude of jumps at nodal points

Let the parameter C from section 3.5.2 be equal to 2. The motivation for this method was to limit not only the elements where undershoots and overshoots arise but also the elements where the solution is discontinuous at any of the endpoints. For the FCT method using the low order scheme with modification of the solution on the previous time step and the initial condition $\sin(2\pi x)$ we can observe the improvement of the solution comparing to the minmod limiting strategy. At times $0.08, 0.1$ and 0.14 the average value over the middle interval is zero as expected. At time around 0.16 the average value over the middle interval becomes negative but it does not oscillate with time between negative and positive values, it remains negative. In the Figure 4.19, we can't see the undershoots and overshoots caused by the oscillating of the solution on the middle interval, hence from the figure we do not see, what the average value over the middle interval is. The reason why the average value over the middle interval is not zero is the same as for the minmod limiting, i.e. the strategy of selecting intervals on which we use low order scheme does not choose the intervals symmetrically. The results for the second initial condition (4.4) are almost identical to the solution computed by the FCT method using minmod limiting. No visible improvement is achieved using the limiting strategy based on magnitudes of jumps at nodal points hence the results will not be presented.

The solution for the FCT method using penalty terms and the limiting strategy based on magnitudes of jumps at nodal points does not differ much from the solution computed by the FCT method using penalty terms and the minmod limiting. Hence we do not provide as detailed results as in Figure 4.17 and 4.18. For the solution of Burgers' equation computed by the FCT method using the low order scheme with penalty terms and the limiting strategy based of magnitudes of jumps at nodal points as described in 3.5.2 equipped with the initial condition $\sin(2\pi x)$ see Figure 4.20, for the second initial condition (4.4) see Figure 4.21.

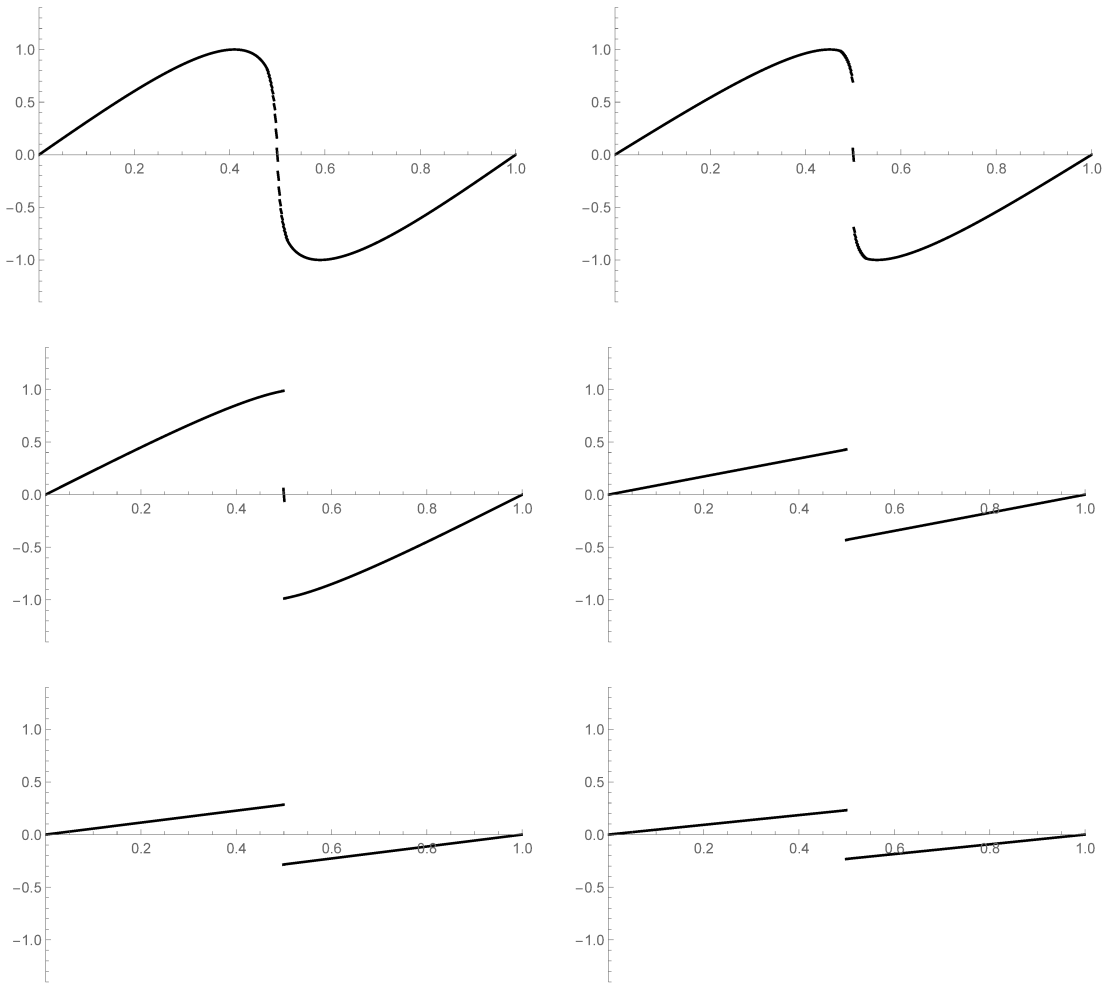


Figure 4.19: Burgers' equation with initial condition $\sin(2\pi x)$ computed by the FCT method using the limiting strategy based on the magnitudes of jumps at nodal points on the grid with 501 elements. The low order scheme with modification of the solution on previous level. Time step set to 0.001, figures show the solution at time $0.08, 0.1, 0.14, 0.5, 0.8$ and 1, respectively.

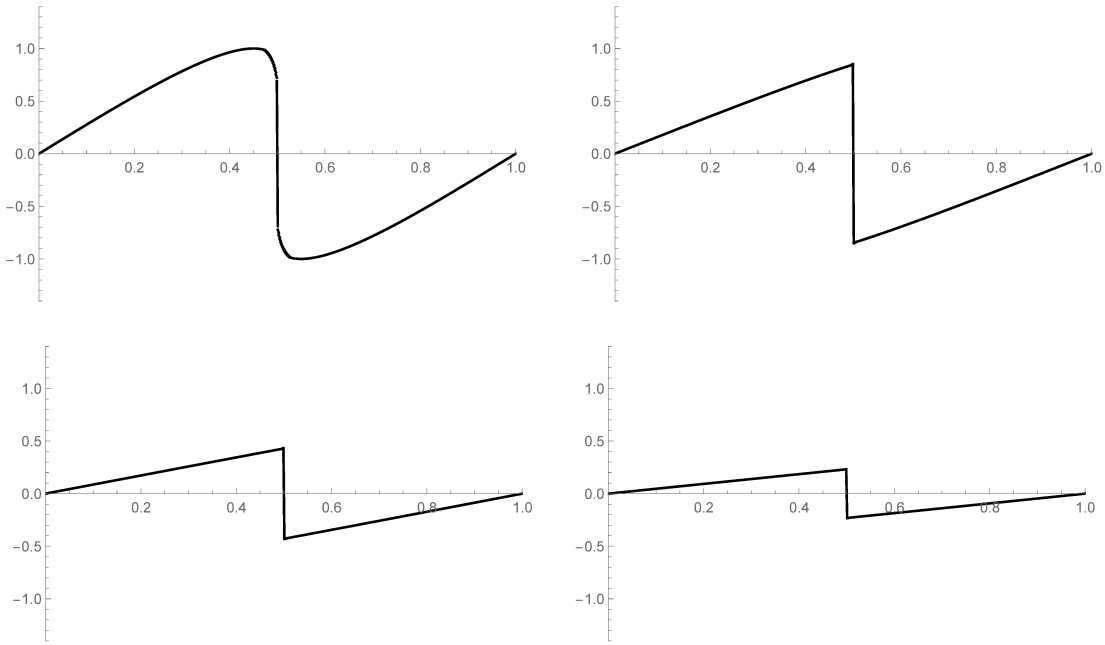


Figure 4.20: Burgers' equation with initial condition $\sin(2\pi x)$ computed by the FCT method, using the limiting strategy based on magnitudes of the jumps at nodal points, on the grid with 501 elements. The low order scheme with penalty terms. Time step set to 0.001, figures show the solution at time $0.1, 0.2, 0.5$ and 1, respectively.

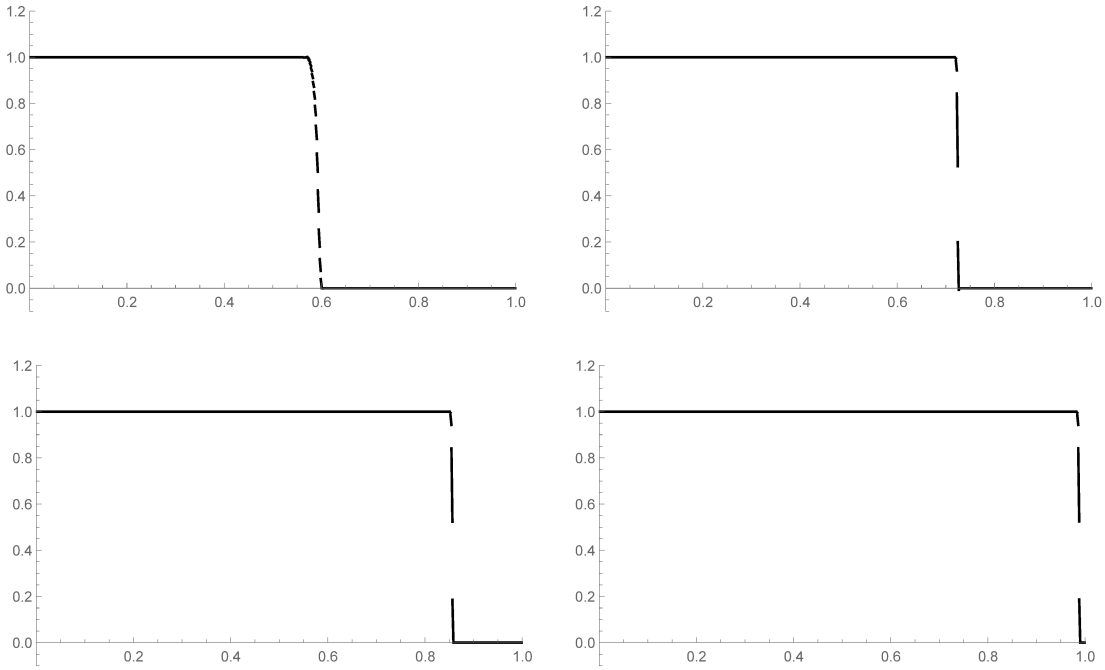


Figure 4.21: Burgers' equation with initial condition $\sin(2\pi x)$ computed by the FCT method, using the limiting strategy based on magnitudes of the jumps at nodal points, on the grid with 501 elements. The low order scheme with penalty terms. Time step set to 0.001, figures show the solution at time $0.1, 0.4, 0.7$ and 1, respectively.

4.4.3 Magnitude of derivatives

Let us set the parameter C_2 to $7 * h$, where h is the norm of the partition. The reasoning for this strategy was that the previous strategies choose the limiting elements after some "bad" behavior appears in the solution. We have spoken about the delay in elements selection. The limiting strategy based on magnitudes of derivatives should prevent the delay and select the elements on which we use the low order scheme just before the overshoots and undershoots or jumps appears. The consequence of the delay are the overshoots and undershoots which will be presented later. For small time steps, in comparison to the norm of the partition, we do not observe the impact of the delay on the solution.

First we show the results for the FCT method using the low order scheme with modification of the solution of the previous time step for the initial condition to be $\sin(2\pi x)$. The top left figure in Figure 4.22 is interesting because in the area, where the function decreases, the low order scheme is not used on all elements, there are several groups of following intervals where the solution is computed by the DG method. This causes the unexpected shape of the function. In addition we can observe the same behavior as in Figure 4.15 and Figure 4.19. The average value over the middle interval is zero up to time 0.22 then again the solution on this interval changes the average to be negative and it is connected to the solution in the right part of the domain $(0, 1)$. Though this combination of low order method and limiting strategy is not recommended because the derivative of the solution on the middle element can be small, comparing e.g. to the derivative of the solution computed by the FCT method using low order scheme with penalty terms, and hence the limiting method may not capture the middle interval.

The FCT method using the low order scheme with penalty terms combined with the limiting strategy based on magnitudes of derivatives provides also good results. To a lesser extent we can observe that the limiting method selects for the majority of intervals, where the solution decreases, to use the low order scheme but some group of following intervals are skipped and the DG method is used on them. Otherwise the solution does not show visible undershoots and overshoots for the $\sin(2\pi x)$ initial condition and has similar properties as the solutions with the two previous limiting strategies, Figure 4.23.

For the initial condition (4.4) the combination of the low order scheme with penalty terms and limiting strategy based on magnitudes of derivatives does not work well. At time 0.1 a small overshot and bigger undershot is visible, see Figure 4.24. Especially the undershot is undesired because the analytic solution is definitely non negative and we would like to preserve this property.

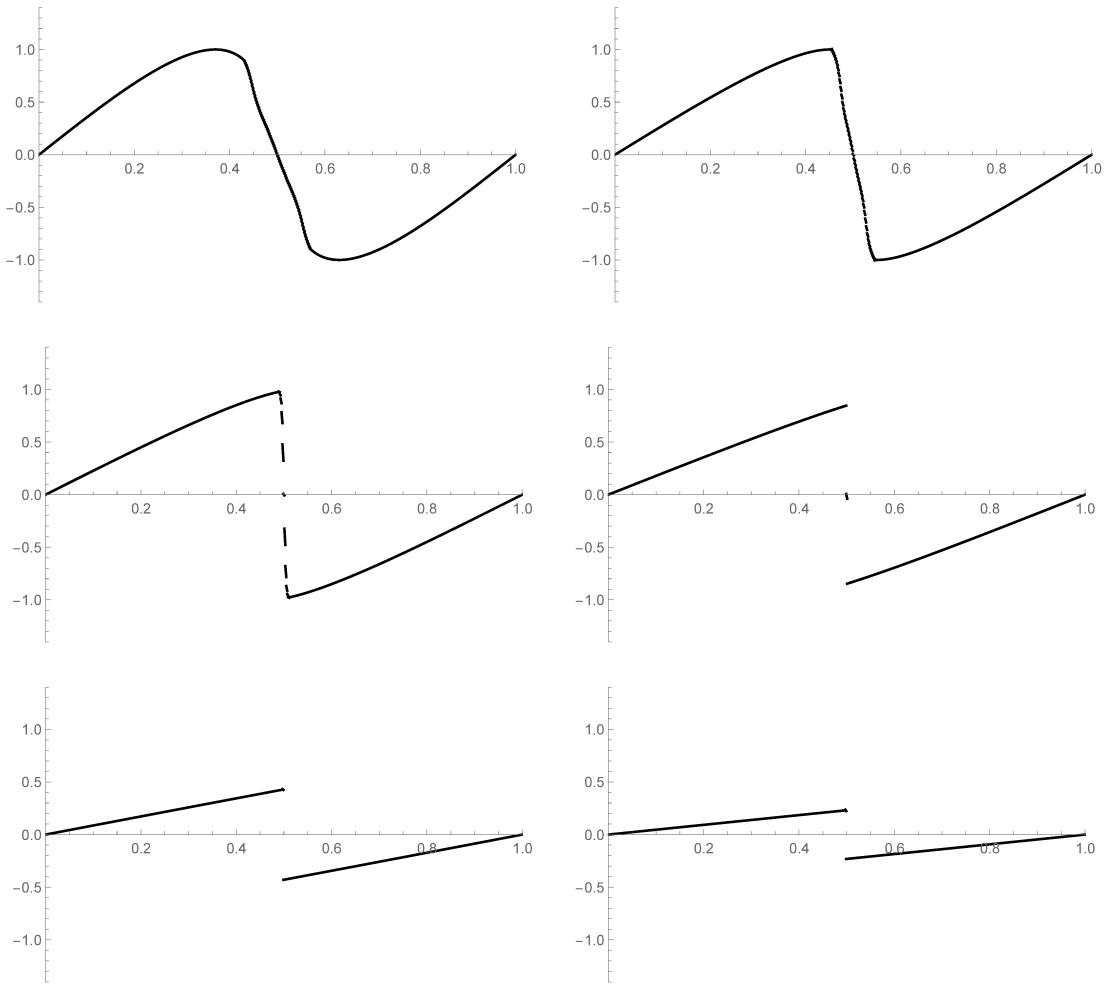


Figure 4.22: Burgers' equation with initial condition $\sin(2\pi x)$ computed by the FCT method, using the limiting strategy based on magnitude of derivatives, on the grid with 501 elements. The low order scheme with modification of the solution on previous level. Time step set to 0.001, figures show the solution at time $0.06, 0.1, 0.14, 0.2, 0.5$ and 1, respectively.

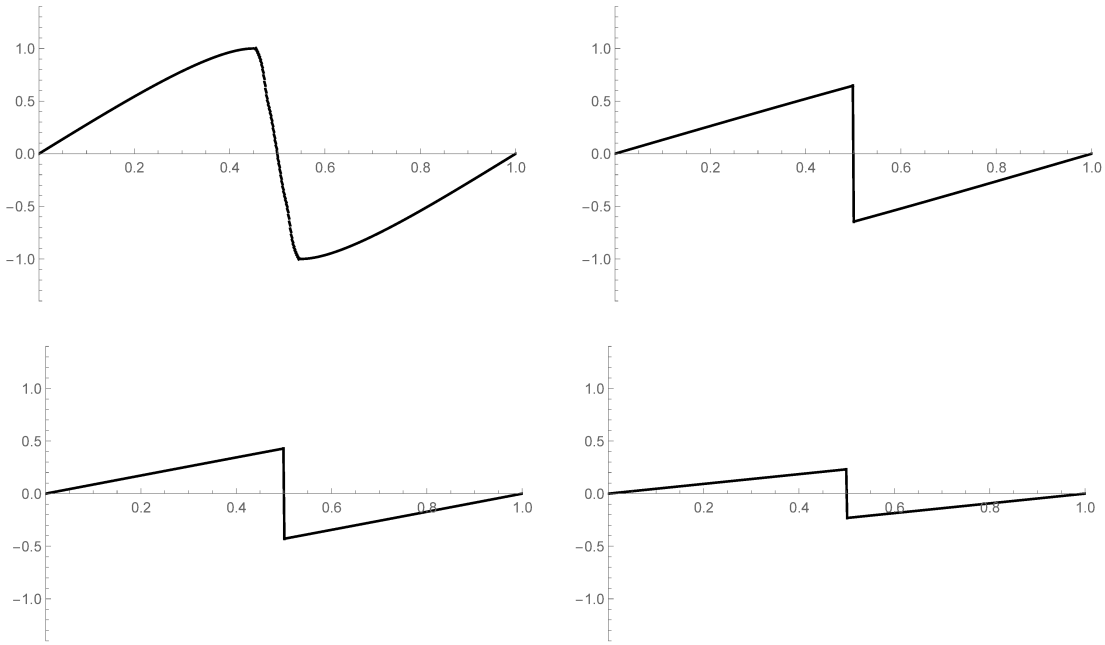


Figure 4.23: Burgers' equation with initial condition $\sin(2\pi x)$ computed by the FCT method, using the limiting strategy based on magnitude of derivatives, on the grid with 501 elements. The low order scheme with penalty terms. Time step set to 0.001, figures show the solution at time $0.1, 0.3, 0.5$ and 1, respectively.

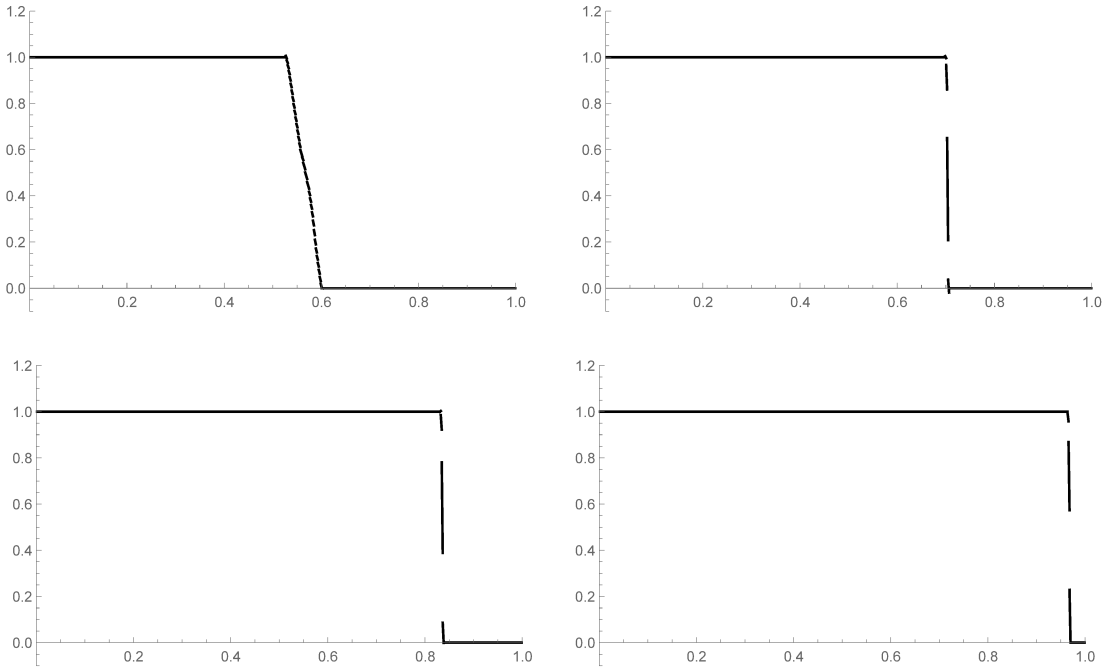


Figure 4.24: Burgers' equation with initial condition $\sin(2\pi x)$ computed by the FCT method, using the limiting strategy based on magnitude of derivatives, on the grid with 501 elements. The low order scheme with penalty terms. Time step set to 0.001, figures show the solution at time $0.1, 0.4, 0.7$ and 1, respectively.

4.4.4 Computing on coarse grids

In this section we do not provide a complete overview of all methods. The purpose of this subsection is to show the major differences when computing on coarse grids. The grid consists of 39 elements. Otherwise the settings remains the same as above. The time step is 0.001.

The FCT method using the low order scheme with penalty terms works as well as for fine grids. As it was mentioned, the overshoots and undershoots are more visible but do not increase too much. Figure 4.25 shows the solution of Burgers' equation equipped with the $\sin(2\pi x)$ initial condition computed with the FCT method, the low order scheme with penalty terms and minmod limiting strategy.

Figure 4.26 shows the results for the initial condition (4.4), the solution computed by the FCT method using the low order scheme with penalty terms and minmod limiting strategy. We can see that the velocity of propagation is a bit larger than for fine grids. The same is true for the others limiting strategies. Otherwise the solutions has the same properties as for fine grids.

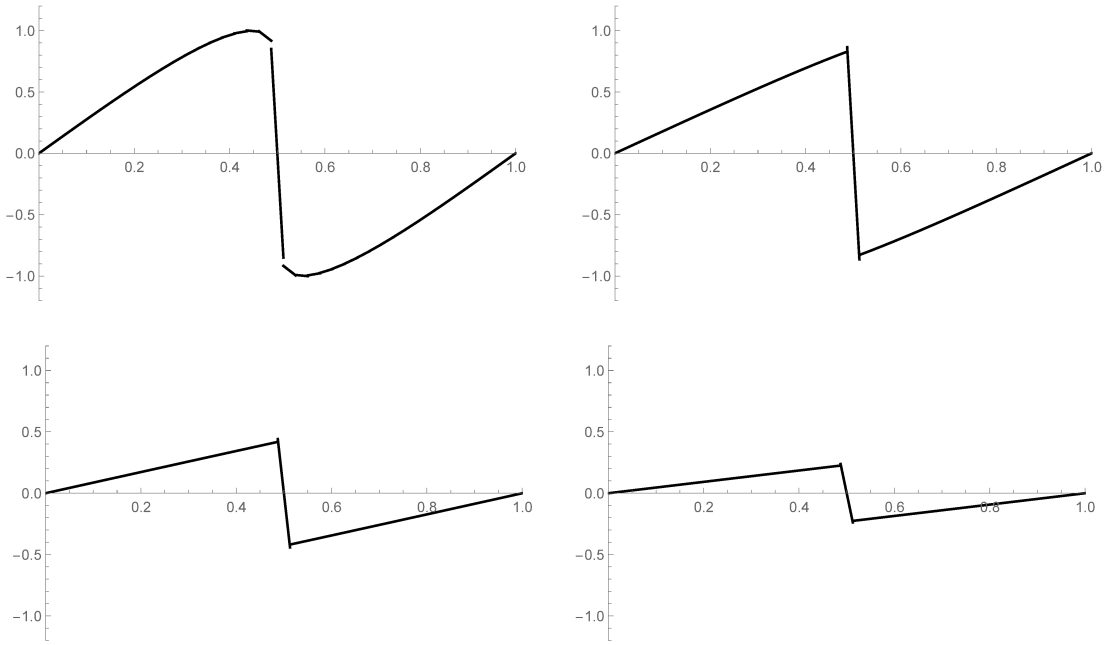


Figure 4.25: Burgers' equation with initial condition $\sin(2\pi x)$ computed by the FCT method, using minmod limiting, on the grid with 39 elements. The low order scheme with penalty terms. Time step set to 0.001, figures show the solution at time $0.1, 0.2, 0.5$ and 1, respectively.

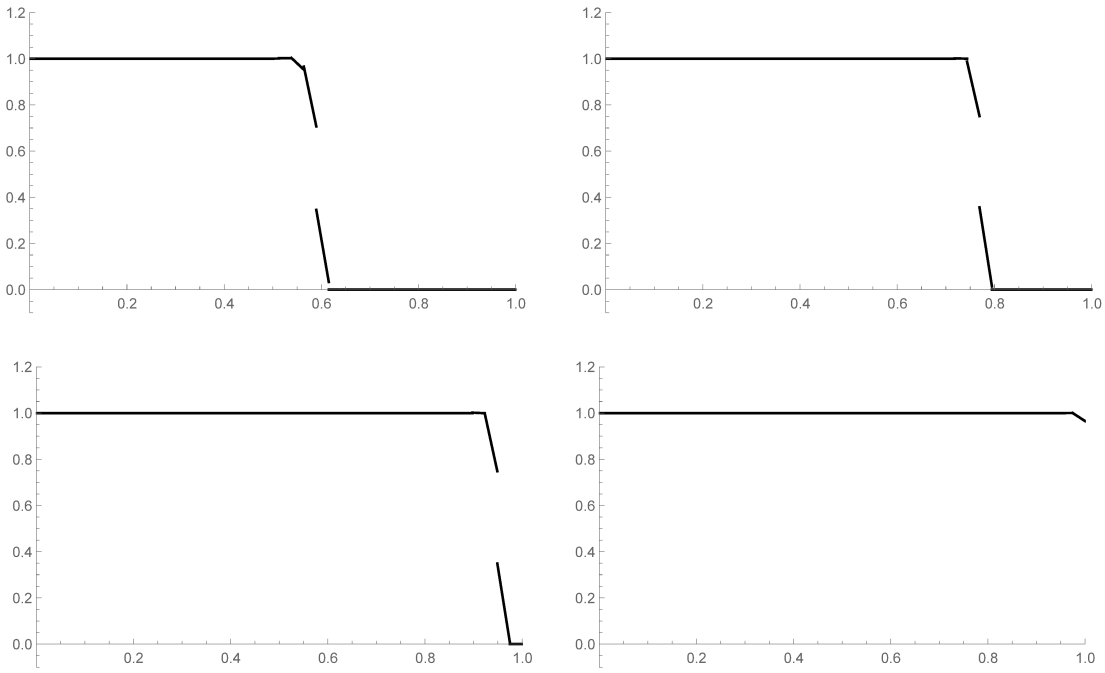


Figure 4.26: Burgers' equation with initial condition (4.4) computed by the FCT method, using minmod limiting, on the grid with 39 elements. The low order scheme with penalty terms. Time step set to 0.001, figures show the solution at time $0.1, 0.4, 0.7$ and 0.8, respectively.

The solution is improved when computed by the FCT method combined the low order scheme with modification of the solution on the previous time step and the limiting strategy based on the magnitudes of derivatives. We can observe that the average value over the middle interval remains zero for all times, the limited intervals are selected symmetrically, Figure 4.27.

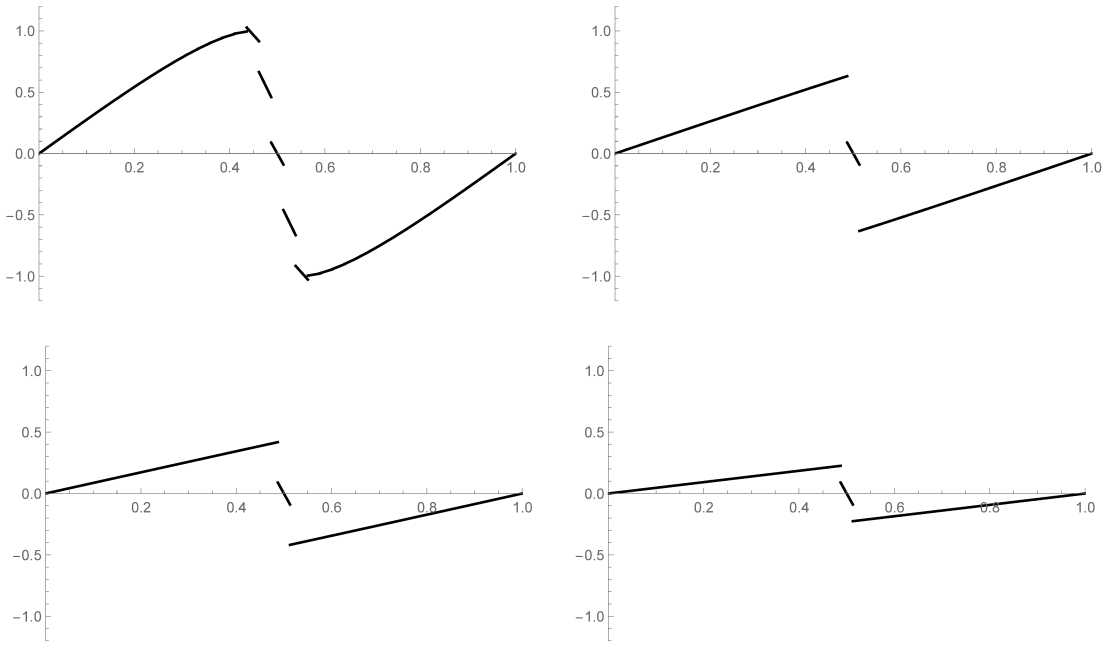


Figure 4.27: Burgers' equation with initial condition $\sin(2\pi x)$ computed by the FCT method, using the limiting strategy based on magnitudes of derivatives, on the grid with 39 elements. The low order scheme with modification of the solution on the previous time step. Time step set to 0.001, figures show the solution at time $0.1, 0.3, 0.5$ and 1, respectively.

What is also surprising is that for the intial condition (4.4), the solution computed by the FCT method using the low order scheme with modification of the solution on the previous time step together with the minmod limiting strategy propagates the signal, see Figure 4.28. The velocity though is very small. From the numerical experiments we can judge that it depends on the norm of the partition. The signal transportation is also visible when the limiting strategy based on magnitude of derivatives is used.

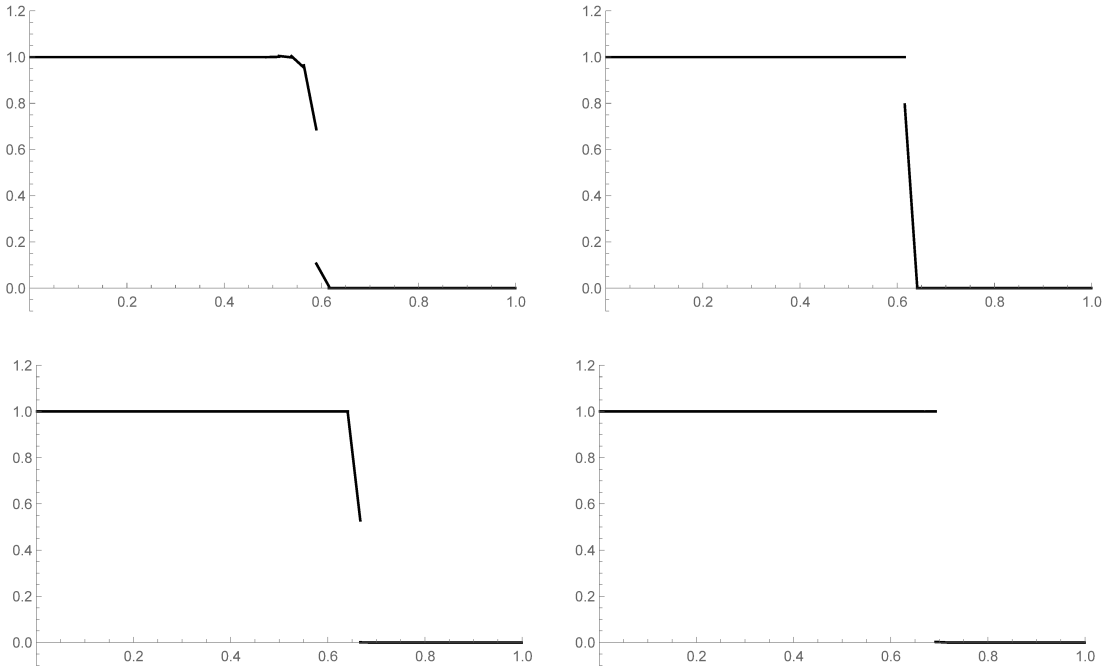


Figure 4.28: Burgers' equation with initial condition (4.4) computed by the FCT method, using the minmod limiting strategy, on the grid with 39 elements. The low order scheme with modification of the solution on the previous time step. Time step set to 0.001, figures show the solution at time $0.1, 0.4, 0.7$ and 1, respectively.

The FCT method using the low order scheme with the modification of the solution on the previous time step together with minmod limiting strategy provides the solution that is as bad as the DG method or even worse, Figure 4.29. The solution shows big overshoots an undershoots, it is not symmetric at time 0.5 and at time 0.6 the average over the middle element is not zero. The "bad" behavior is caused by the delay in the selecting of limited elements. Moreover, the later the method select the element on which the solution is computed by the low order method, the bigger the overshoots and undershoots are. We demonstrate this assertion in Figure 4.30. We use the FCT method and the low order scheme with the modification of the solution on the previous time step. From time 0.1 further on we select the 20-th element on which we use the low order scheme, the solution is similar to the one in Figure 4.29. Then we compute the solution with the same settings but we use the low order scheme on the 20-th element from time 0.07 further on. The second solution shows only small overshoots and undershoots.

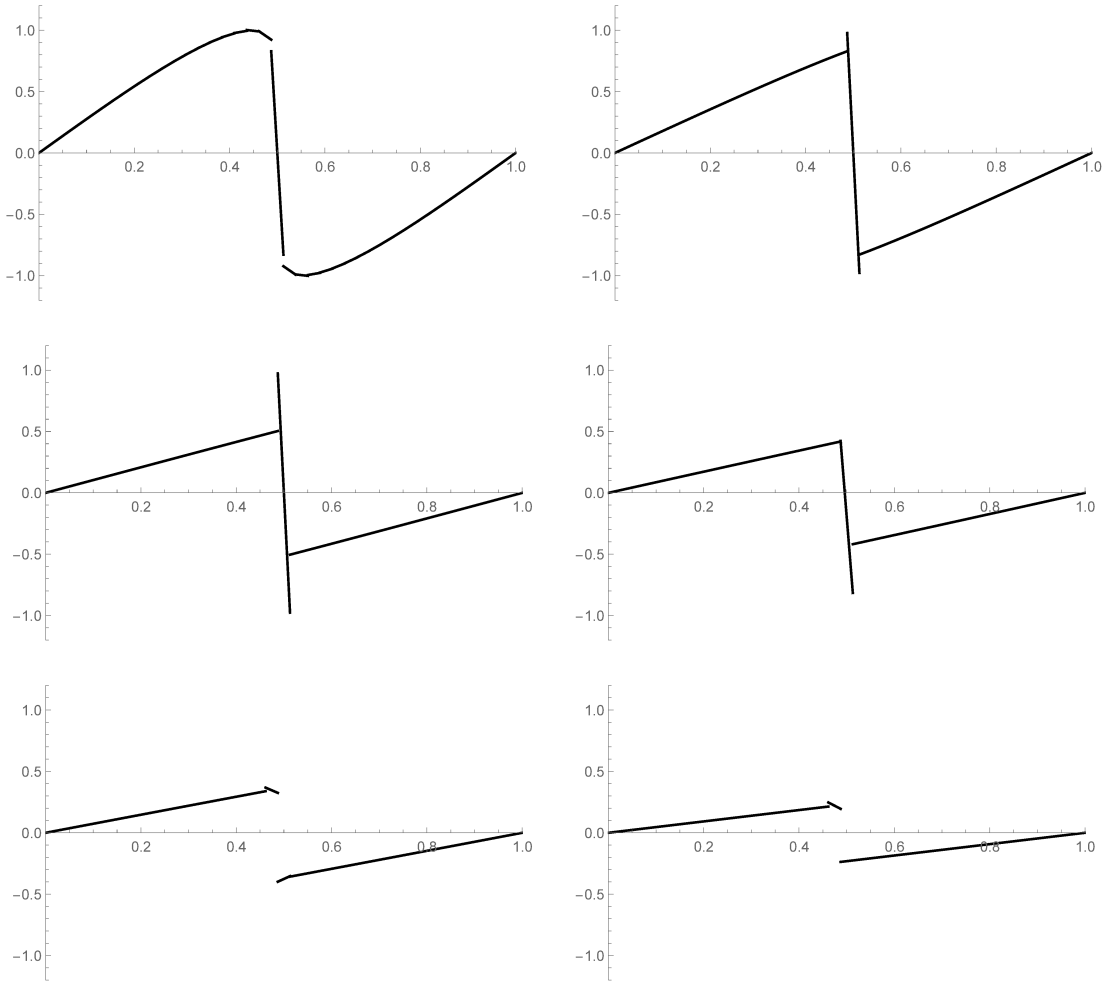


Figure 4.29: Burgers' equation with initial condition $\sin(2\pi x)$ computed by the FCT method using minmod limiting on the grid with 39 elements. The low order scheme with modification of the solution on previous level. Time step set to 0.001, figures show the solution at time $0.1, 0.2, 0.4, 0.5, 0.6$ and 1, respectively.

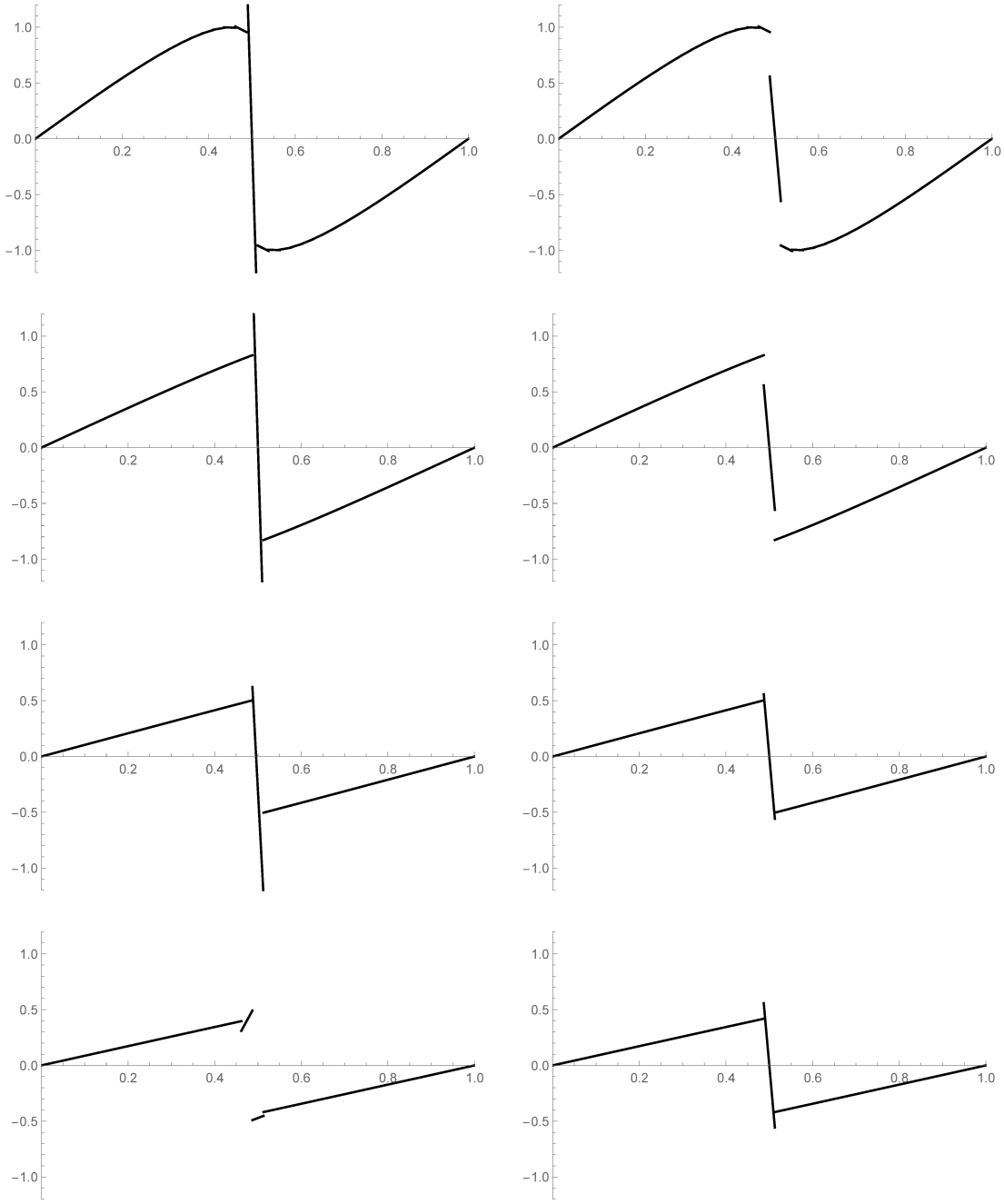


Figure 4.30: Burgers' equation with initial condition $\sin(2\pi x)$ computed by the FCT method on the grid with 39 elements. The low order scheme with modification of the solution on previous level is used for the 20-th element, the left column figures show the solution with limiting switch on in time 0.1, the right column provides the solution with limiting switch on in time 0.07 . Time step set to 0.001. Figures show the solution at time $0.1, 0.2, 0.4$ and 0.5, respectively.

4.4.5 The choice of time step

In the proof of Corollary 7 we have shown that we need sufficiently small time step in comparison to the norm of partition in order to the matrix $\mathbb{M}_L + \Delta t_k \mathbb{L}$ be an M-matrix. If we choose the time step to be bigger by an order than the norm of the partition, the oscillations arise around the discontinuity, Figure 4.31. If we increase the time step too much, eventually we can observe the blow up of the solution. Providing the Figure 4.32 for the time step to be in order twice as big as the norm of the partition.

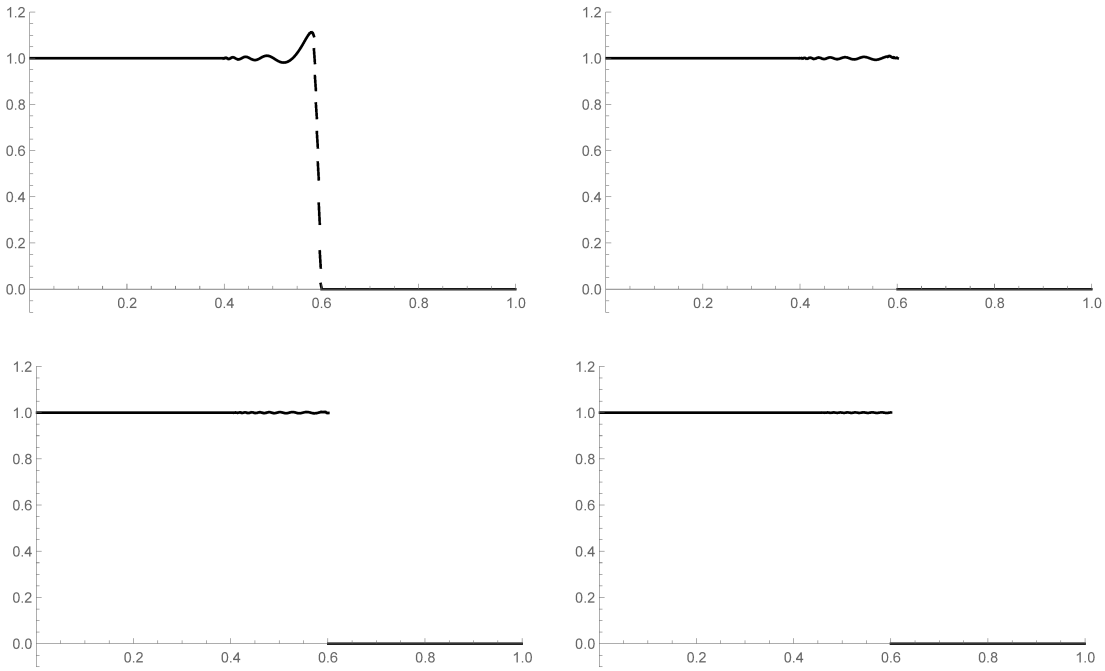


Figure 4.31: Burgers' equation with initial condition (4.4) computed by the FCT method, using the limiting strategy based on the magnitude of jumps at nodal points, on the grid with 501 elements. The low order scheme with modification of the solution on the previous time step. Time step set to 0.01, figures show the solution at time $0.1, 0.2, 0.4$ and 1, respectively.

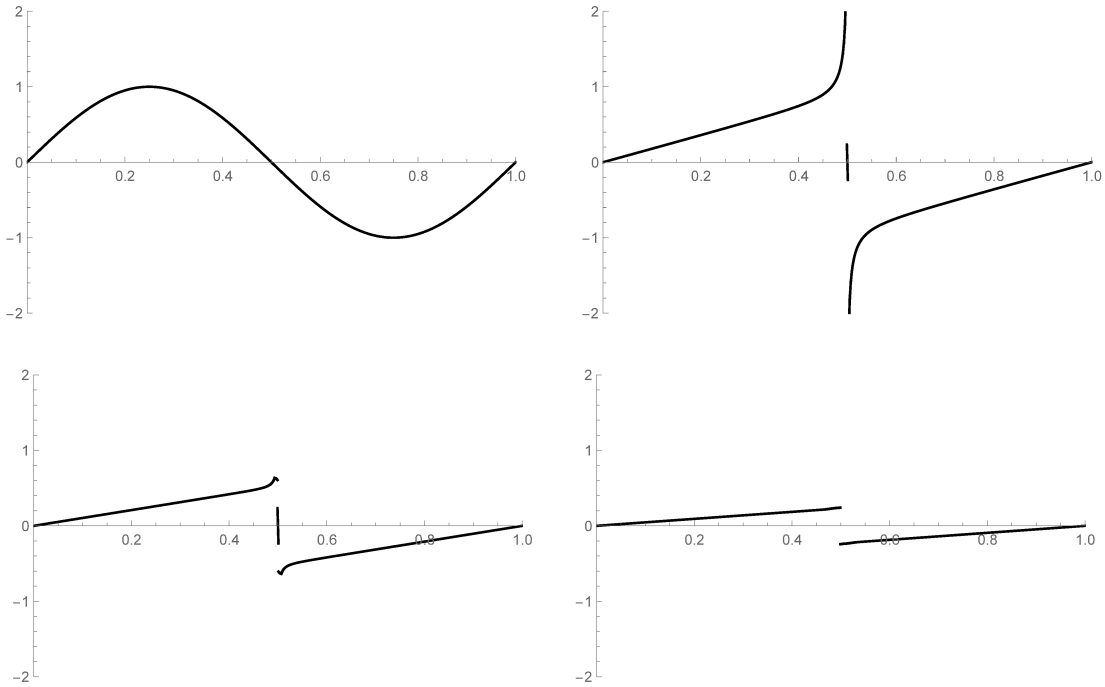


Figure 4.32: Burgers' equation with initial condition $sin(2\pi x)$ computed by the FCT method, using the minmod limiting strategy, on the grid with 501 elements. The low order scheme with modification of the solution on the previous time step. Time step set to 0.1, figures show the solution at time $0, 0.1, 0.2$ and 0.5, respectively.

4.5 Comparing of the DG, LOS and FCT method

Finally, we can compare the solutions computed by the three major methods: the DG method, the low order method with penalty terms and the FCT method using the low order method with penalty terms with minmod limiting strategy. From the results presented earlier we can judge that these methods provides the most reasonable results from all the methods discussed in this work.

First, for the initial condition $\sin(2\pi x)$ see Figure 4.33. The DG method shows overshoots and undershoot. The low order scheme decreases its function values too slowly. We can say that the low order scheme is delayed compared to the DG method. And the FCT method chooses the best from both methods. It function values goes to zero as fast as the DG method but it does not posses overshoots and undershoot.

We make the similar comparison for the initial condition (4.4), Figure 4.34. The DG method shows overshoots and undershoots. The low order method with penalty terms transports the signal too slowly. We can see that the major drawback of the low order method, that is the velocity of signal propagation, is not improved much by the FCT method.

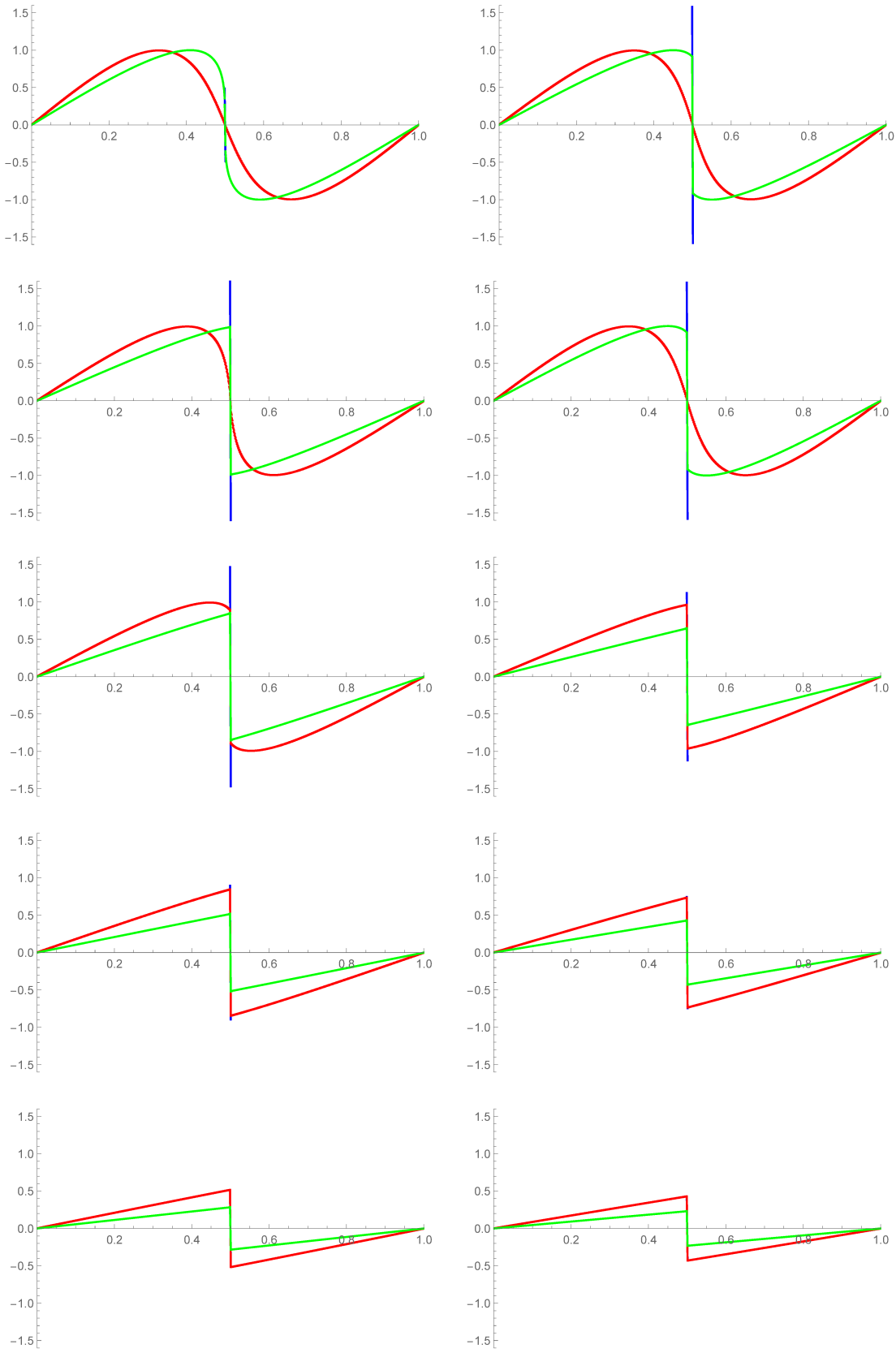


Figure 4.33: Burgers' equation with initial condition $sin(2\pi x)$ computed by the DG method(blue line), LOS with penalty terms (red line) and the FCT method, using the minmod limiting strategy (green line), on the grid with 501 elements. Time step set to 0.001, figures show the solution at time $0.1, 0.3, 0.5$ and 1, respectively.

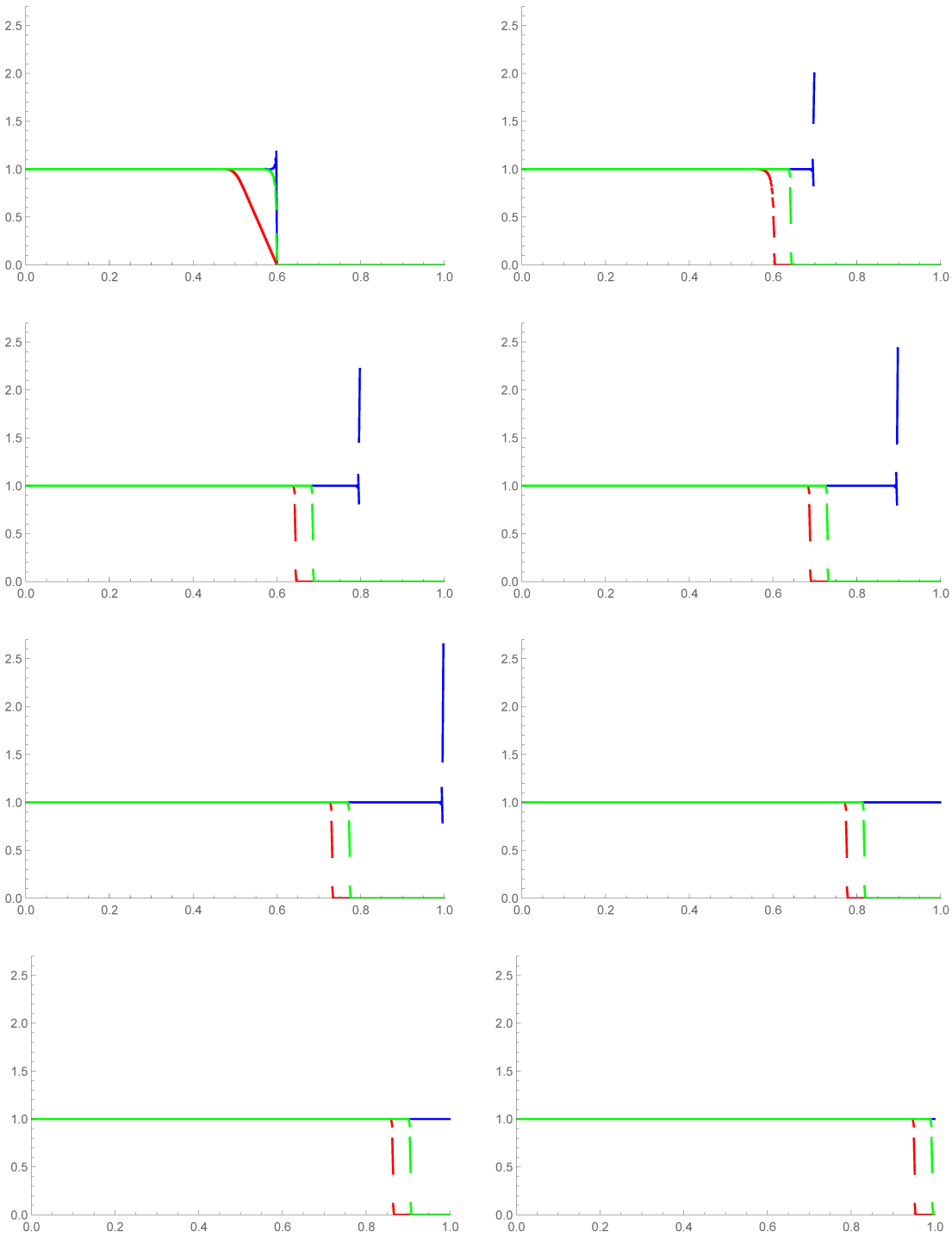


Figure 4.34: Burgers' equation with initial condition (4.4) computed by the DG method(blue line), LOS with penalty terms (red line) and the FCT method, using the minmod limiting strategy (green line), on the grid with 501 elements. Time step set to 0.001, figures show the solution at time $0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.8$ and 1, respectively.

Conclusion

The main goal of this work was to solve Burgers' equation while avoiding the undesired Gibbs phenomenon. The idea was to use discontinuous Galerkin method instead of the standard finite element method. The solution computed by the DG method does not remove completely the oscillations, which are the manifestation of the Gibbs phenomenon. As a remnant of the oscillations, we can observe overshoots and undershoots in the solution.

In order to reduce the overshoots and undershoots, we applied the low order methods. The major obstacle in using the stable low order method was that the method was developed for the convection-diffusion-reaction equation under the condition of incompressibility of the velocity field. Burgers' equation has slightly different properties hence we needed to enforce equality of the row sums of system matrices, condition (3.2).

We suggested two approaches. The first one, that modifies the solution on the previous time step such that the modified solution imitates constant solution, does not show overshoots and undershoots. However, the communication between neighboring element is limited and for the initial condition (4.4) it gives wrong results. The second approach was rather "engineering". The row sum condition was enforced by adding appropriate constants to each diagonal element. We call this method the low order scheme with penalty terms. This method works well, no overshoots and undershoots arise. The disadvantage of both low order methods is that the evolution of the solution is slowed down. It shows up as small velocity of propagation of the signal. Another problem with the low order scheme is that the scheme is too diffusive. The solution becomes discontinuous as it evolves in time and this phenomenon is often called terracing.

The FCT method is the method which combines the DG method and the low order method. The idea is to use the DG method on the areas where the solution is smooth and does not show overshoots and undershoots. On the rest of the domain it uses the low order method and thus it prevents the manifestation of the Gibbs phenomenon. The major advantage of the FCT method that it uses only the algebraic operations, hence it can be relatively easily implemented. Also the computational costs are low, if combined with an iterative solver.

The major question of the FCT method is how to choose the areas, where the low order method should be applied. This works present three strategies. The first one is the minmod limiting, inspired by the work of C.-W. Shu refer to Shu [2009]. The original method seeks overshoots and undershoots of an explicit scheme and corrects the values. So the minmod limiting strategy uses the part that captures the overshoots and undershoots and selects the intervals accordingly. The method works good but it reacts to the "bad" behavior of the solution. We speak about the delay of capturing the intervals.

The second strategy based on magnitudes of jumps at the nodal point was suggested by Václav Kučera. The method should capture also the intervals such that the significant discontinuity is located at any endpoint of this interval. The solution obtained by this choice of limiting strategy is similar to the solution obtained by the minmod limiting. The drawback remains the same, the method selects the interval after the solution is corrupted.

We needed a limiting strategy that can predict where the problem appears. The overshoots and undershoots arise around discontinuities and in the areas where the solution changes significantly. Hence we choose the elements based on the magnitude of the derivatives. This strategy produces in some cases the best results. However, the method requires the knowledge about the derivative of computed solution, which we do not have.

To summarize, we found the methods that can successfully suppress the Gibbs phenomenon. If we do not have any information about the computed solution, we suggest to use the FCT method using the low order scheme with penalty terms and minmod limiting. Concerning this method, there is an open question, if there is a satisfying mathematical explanation of the penalty terms. Other extension of this work is providing the quantitative comparison of presented methods.

Attachments

```
(*Define the initial condition equal to sin(2π x)*)
Clear[m, m0];
m[x_?NumericQ] := m0 /. FindRoot[m0 == Sin[2*Pi*x], {m0, x}];

(*Solve Burgers' equation for the given initial condition*)
{sol} = NDSolve[{D[u[t, x], t] + u[t, x] D[u[t, x], x] == 0,
   u[0, x] == m[x],
    u[t, -1] == 0, u[t, 1] == 0}, u, {t, 0, 1}, {x, -1, 1}]

(*Plot the results*)
Plot3D[u[x, t] /. sol, {x, -1, 1}, {t, 0, 1}, AxesLabel -> Automatic]
```

Listing A1: Mathematica code for solving Burgers' equation

Bibliography

J. P. Boris and D. L. Book. Flux-corrected transport. I. SHASTA, a fluid transport algorithm that works. *Journal of Computational Physics*, 11:38–69, 1973.

M. Feistauer and V. Kučera. On a robust discontinuous galerkin technique for the solution of compressible flow. *Journal of Computational Physics*, 224:208–221, 2007.

M. Feistauer, J. Felcman, and I. Straškraba. *Mathematical and computational methods for compressible flow*. Oxford University Press, 2003.

V. John and E. Schmeyer. Finite element methods for time-dependent convection–diffusion–reaction equations with small diffusion. *Computer methods in applied mechanics and engineering*, 198(3-4):475–494, 2008.

L. Krivodonova. Limiters for high-order discontinuous galerkin methods. *Journal of Computational Physics*, 226(1):879–896, 2007.

D. Kuzmin. *A Guide to Numerical Methods for Transport Equations*. University Erlangen-Nuremberg, 2010. http://www.mathematik.uni-dortmund.de/~kuzmin/Transport.pdf.

D. Kuzmin and M. Möller. Algebraic flux correction I. scalar conservation laws. In D. Kuzmin, R. Löhner, and S. Turek, editors, *Flux-Corrected Transport: Principles, Algorithms and Applications*, pages 155–206. Springer, 2005.

P. O. Persson and J. Peraire. Sub-cell shock capturing for discontinuous galerkin methods. In *44th AIAA Aerospace Sciences Meeting and Exhibit*, page 112, 2006.

C.-W. Shu. Discontinuous galerkin methods: general approach and stability. *Numerical solutions of partial differential equations*, 201, 2009.

E. W. Weisstein. Gibbs phenomenon. From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/GibbsPhenomenon.html, Visited on 16/06/18.

Wolfram Research, Inc. Mathematica 11.0. Champaign, IL, 2018.

S. T. Zalesak. Fully multidimensional flux-corrected transport algorithms for fluids. *Journal of Computational Physics*, 31:335–362, 1979.