Universität des Saarlandes

Fachrichtung 6.1 – Mathematik

Preprint Nr. 258

Fuzzy flux limiter schemes for hyperbolic conservation laws

Michael Breuß and Dominik Dietrich

Saarbrücken 2010

Fachrichtung 6.1 – Mathematik
Universität des Saarlandes

Preprint No. 258
submitted: March 5, 2010

Fuzzy flux limiter schemes for hyperbolic conservation laws

Michael Breuß

Saarland University
Faculty of Mathematics and Computer Science
P.O. Box 15 11 50
66041 Saarbrücken
Germany
breuss@mia.uni-saarland.de

Dominik Dietrich

DFKI Bremen
Cartesium 2.057
Enrique-Schmidt-Str. 5
28359 Bremen
Germany
Dominik.Dietrich@dfki.de

Edited by
FR 6.1 – Mathematik
Universität des Saarlandes
Postfach 15 11 50
66041 Saarbrücken
Germany

Fax: + 49 681 302 4443
e-Mail: preprint@math.uni-sb.de
WWW: http://www.math.uni-sb.de/

Abstract

A classic strategy to obtain high-quality discretisations of hyperbolic partial differential equations (PDEs) is to employ a non-linear mixture of two types of approximations. The building blocks for this are a monotone first-order scheme that deals with discontinuous solution features and a higher-order method for approximating the smooth solution parts. The blending is performed by the so-called flux limiter function.

In this paper we introduce a novel approach to flux limiter methods. We show that fuzzy logic (FL) is a useful tool to understand and formulate the limiter functions. After introducing the set-up, we verify that a variety of classic flux limiters can easily be interpreted via FL. Then we show how one can improve limiters making use of the developed FL framework. This is done for initial data and PDEs that describe characteristic settings for hyperbolic problems.

Our work shows for the first time in the literature that the two different fields of fuzzy logic and numerical methods for PDEs can be brought together with benefit.

1 Introduction

Fuzzy logic (FL) is an important tool in computer science [11, 16, 36] and engineering [12, 18, 29, 34]. It allows the control of complex processes based on a small number of expert rules, representing explicit knowledge of the behaviour of the considered system.
In this paper, we consider a novel field of application for FL: The construction of numerical schemes for partial differential equations (PDEs). More specifically, we deal with the approximation of hyperbolic conservation laws (HCLs). Such equations arise in many disciplines, e.g., in gas dynamics, acoustics, geophysics, or astrophysics, cf. [20, 22, 35] for an overview. They describe wave propagation and transport phenomena, in classic PDEs for instance the evolution of mass, momentum, or energy. Concerning numerical methods for HCLs it is well known that the class of so-called flux limiter schemes gives favourable results. In this paper we show for the first time in the literature how to use FL for the construction of flux limiter methods.

Numerical methods for hyperbolic conservation laws. For the construction of numerical methods for HCLs, it is adequate to consider the one-dimensional scalar initial value problem for the unknown $u(x,t)$

$$\begin{cases} \frac{\partial}{\partial t}u(x,t) & = & -\frac{\partial}{\partial x}f(u(x,t)) \quad \text{for } x \in \mathbb{R},\, t > 0 \\ u(x,t=0) & = & u_0(x) \end{cases} \tag{1}$$

Thereby, $f \in C^2$ is the so-called flux function, and $u_0(x)$ is assumed to be a piecewise smooth function of compact support. Usually, x and t are associated with space and time, respectively.
The difficulty in dealing with HCLs numerically stems from the fact that their solutions involve the formation of discontinuities, see e.g. [8, 35]. It is well known that only the

class of monotone schemes approximates discontinuous solutions correctly. They converge to the physically relevant entropy solution, a terminology borrowed from gas dynamics. Unfortunately, monotone schemes are only of first-order accuracy which may limit their practical usefulness. As a means to obtain nevertheless numerical solutions of a reasonable quality, the *high-resolution (HR) schemes* have been developed. The idea of the HR schemes is to blend a monotone method used at 'critical' regions, i.e. at discontinuities and data extrema, with a higher-order scheme that gives good approximations of smooth solution parts. Supplemented by the so-called *total variation diminishing (TVD)* stability notion, this approach has been one of the most successful strategies in the construction of numerical schemes for HCLs over the last decades. For detailed accounts of HR-TVD schemes and the underlying theory, see e.g. the textbooks [19, 20, 30].

In our paper, we deal with the construction of HR-TVD schemes by using *flux limiters*, cf. [27]. For describing the set-up, let us cover the computational domain $\mathbb{R} \times \mathbb{R}_0^+$ uniformly by cells $C_i^n := [x_{i-1/2}, x_{i+1/2}] \times [t^n, t^{n+1}]$, where we employ for some index k the notations $x_k := k \cdot h$, and $t^k := k \cdot \tau$. Thereby, the parameters h and τ are the spatial and temporal mesh width, respectively. Over the cells C_i^n, the unknown $u(x,t)$ is then given in terms of cell averages $U_i^n := \frac{1}{h} \int_{x_{i-1/2}}^{x_{i+1/2}} u(x, t^n)\, \mathrm{d}x$. Then a general explicit numerical scheme reads as

$$\frac{U_i^{n+1} - U_i^n}{\tau} + \frac{F_{i+1/2} - F_{i-1/2}}{h} = 0 \tag{2}$$

The *numerical flux function* F is supposed to be Lipschitz continuous and consistent with the flux f. By the indices $i \pm 1/2$ it is indicated that the numerical fluxes $F_{i\pm 1/2}$ shall approximate the true flux f at the left and right boundary of the cell C_i^n, respectively. For an explicit method the numerical fluxes are evaluated at time level n. The formula (2) is evaluated for each spatial point i of the computational domain. At the end points of the finite computational domain in space one needs numerical boundary conditions. The process begins at the time level $n = 0$, building up a solution by iterating from time level to time level until a prescribed stopping time.

Making use of a low-order numerical flux F^l and a higher-order numerical flux F^h, one can formulate a *flux limiter method* as

$$F_{i-1/2} := F_{i-1/2}^l + \varphi_{i-1/2} \left(F_{i-1/2}^h - F_{i-1/2}^l \right) \tag{3}$$

where φ is the flux limiter. Let us stress that besides the order, the numerical fluxes F^l and F^h used as components of $F_{i\pm 1/2}$ have very different structural properties, for instance in terms of stability. Thus, the properties of the hybrid flux limiter method are controlled by the limiter function φ.

A variety of limiters has been proposed in the literature, cf. [30] for a detailed exposition. The classic limiters are sophisticated non-linear functions. It is somewhat unsatisfactory that their definition is at first glance rather complicated, and an intuition is difficult to develop. Also, a fine tuning or modification of a given limiter is difficult.

Fuzzy flux limiter methods. The main *expert rules* used for the construction of flux limiter methods are given by the basic idea:

(a) Use a monotone scheme at discontinuities.

(b) Use a higher-order scheme at smooth solution parts.

Our aim is to show how a meaningful blending process of the expert rules can be formalised using FL with benefit *(i)* for a deeper understanding of the limiter models, and *(ii)* for the optimisation of classic limiters resulting in novel FL-based flux limiter schemes.
To our best knowledge, we employ FL for the first time in the literature for the construction of numerical schemes for PDEs. In detail, we provide the following innovations:

1. The foundation for the construction of FL-based flux limiters is given, and the key points in the construction of corresponding fuzzy controllers are discussed.
2. Some important classic flux limiters are shown to be the result of relatively simple fuzzy constructions. Thus they can easily be interpreted and modified.
3. We show how to fine tune existing FL-based limiters for specific applications, leading to a superior accuracy of the approximation.

Related work. This work represents a significant extension of our conference paper [2]. There, we gave a brief account of the first point formulated above, and we elaborated on numerical tests concerned with fine tuning FL-based approximations of a linear advection equation. As indicated, to our knowledge no other work has dealt before with the use of FL for constructing numerical schemes for PDEs.

Structure of the paper. In Section 2, we briefly review the classic flux limiter approach for hyperbolic CLs. The tools from FL that we need are introduced in Section 3. This is followed by an exposition on the construction of fuzzy flux limiters in Section 4. Then we show in Section 5 how classic limiters are constructed using the developed framework. In Section 6, we elaborate on the optimisation of the standard limiters, leading to novel fuzzy flux limiter schemes. The paper is finished by a summary and conclusion.

2 Flux Limiter Methods for HCLs

In this section, we briefly review the classic theory for TVD flux limiter methods, cf. the concise textbooks [19, 20, 30]. For relevant original works, see especially [14, 15, 27].
The main problems when dealing with HCLs are discontinuous solution features. For nonlinear equations, these arise in general after small time, even for arbitrarily smooth initial data. Then a solution is not defined in a classic sense anymore, so that one looks for weak (or distributional) solutions of HCLs. While this gives a meaning to discontinuous solutions, in general an infinite number of mathematically admissible weak solutions arise at a discontinuity. Thus, an additional uniqueness condition is imposed, which selects the unique, physically relevant entropy solution out of the set of weak solutions, cf. [1, 25, 26].
Dealing with numerical schemes for HCLs, a fundamental observation on schemes of second or higher order is that they develop oscillations when approximating a discontinuous solution feature. On the other hand, first-order schemes do not show this behaviour, but introduce a significant blurring of discontinuities. Only for a sub-class of the first-order

schemes, the monotone schemes, convergence to the entropy solution can be shown when the mesh is refined [7].
High-resolution flux limiter methods perform a blending of a monotone scheme with a higher-order method. Thereby, the monotone scheme gives the correct approximation of discontinuities, while the higher-order method gives a good resolution of smooth solution parts. The use of a monotone scheme also at data extrema enables the validity of a useful stability property, namely the total variation (TV) stability. This ensures that the numerical solution is in the compact space of functions of bounded variation over a compact space-time domain $\mathcal{BV}([a,b] \times [0,T])$. Then one can prove convergence of numerical solutions when refining the mesh making use of a compactness argument, cf. [19, 20] for more details.

2.1 TVD Flux Limiter Schemes

We recall the basic set-up for an explicit, conservative numerical scheme:

$$U_i^{n+1} = U_i^n - \frac{\tau}{h}\left(F_{i+1/2} - F_{i-1/2}\right) \tag{4}$$

see (2), where F is a consistent numerical flux. A flux limiter method employs

$$F_{i-1/2} := F^l_{i-1/2} + \varphi_{i-1/2}\left(F^h_{i-1/2} - F^l_{i-1/2}\right) \tag{5}$$

with the *limiter* φ, and where F^l, F^h are the low-order and the higher-order numerical flux, respectively, cf. (3). In our study, we consider the classic combination of the monotone upwind method and the second-order Lax-Wendroff scheme. For simplifying the choice of the upwind stencil that depends on the direction of the flow, we generally assume that $f'(\cdot) \geq 0$ holds. Then the corresponding numerical fluxes F^l and F^h are given as

$$F^l_{i-1/2} := F^{upw}(U_{i-1}, U_i) = f(U_{i-1}) \tag{6}$$

$$F^h_{i-1/2} := F^{LW}(U_{i-1}, U_i) = \frac{f(U_{i-1}) + f(U_i)}{2} - \frac{a\left(U_{i-1/2}\right)\tau}{2h}\left(f(U_i) - f(U_{i-1})\right) \tag{7}$$

For the latter formula, we define as usual $a(u) := (\partial f/\partial u)(u)$ and

$$a\left(U_{i-1/2}\right) := \frac{1}{2}\left(a(U_{i-1}) + a(U_i)\right). \tag{8}$$

In order to switch effectively between those schemes at discontinuities and smooth solution parts, we also need at $i \pm 1/2$ and time level n a *smoothness measure* that we denote by $\Theta_{i\pm1/2}$. This is usually computed making use of the ratio of consecutive slopes, and these are chosen in dependence on the flow direction. In our set-up the flow is from left to right, and the smoothness measure is computed as

$$\Theta_{i+1/2} := \frac{U^n_{i+1} - U^n_i}{U^n_i - U^n_{i-1}}. \tag{9}$$

The limiter φ evaluates Θ, and thus it controls the blending of F^l and F^h at hand of the smoothness of the approximation.
Let us turn to the desired properties of the limiter function. Concerning flux limiter schemes, the objective to have *TV* stability leads by the Theorem of Harten [14] to the notion of the *TVD region*. This is a part of the $(\theta, \varphi(\theta))$-domain which must contain the limiter so that the flux limiter scheme is TVD, see Figure 1 (left). An important sub-part

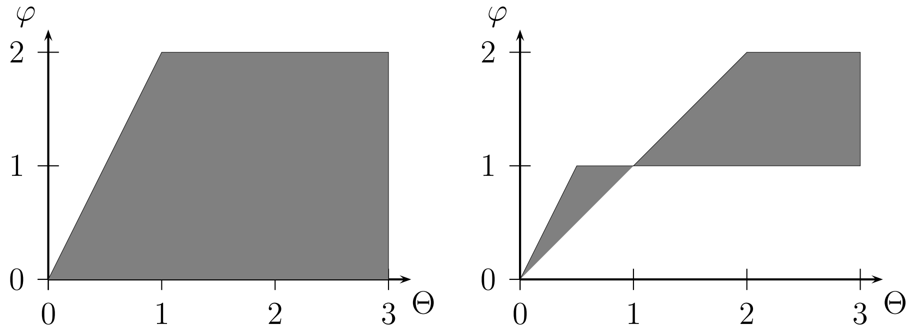


Figure 1: **Left.** TVD region. **Right.** Sweby TVD region for second-order accuracy.

of the TVD region is the *second-order TVD region* identified by Sweby [27], cf. Figure 1 (right). As the notion suggests, if the graph of the limiter lies in this domain, the overall scheme is formally of second order away from discontinuities and extrema.

2.2 Examples of flux limiters

We now briefly review three important examples of flux limiters, namely the Minmod limiter, the Superbee limiter and the MC limiter, see Figure 2. For a thorough discussion and other limiters see e.g. [20, 30] and the references therein.

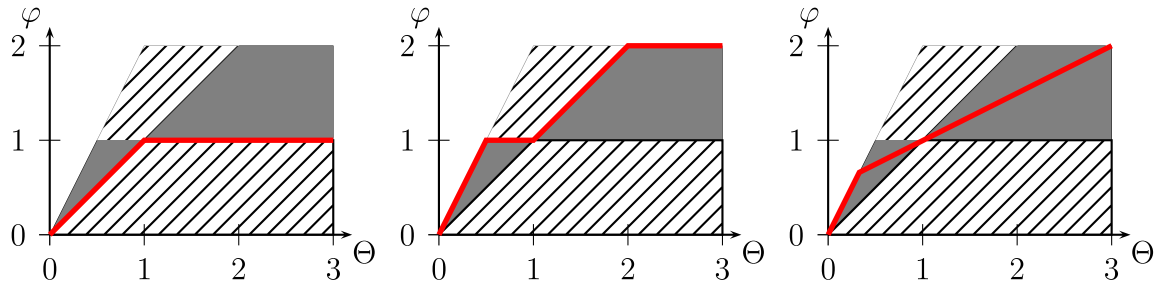


Figure 2: Flux limiters (strong line) and the Sweby TVD region (shaded). **Left.** Minmod limiter. **Middle.** Superbee limiter. **Right.** MC limiter.

The Minmod flux limiter. The *Minmod limiter* is defined via

$$\Phi(\Theta) = \text{minmod}(1, \Theta) \tag{10}$$

where the minmod function is defined as follows:

$$\operatorname{minmod}(a,b) = \begin{cases} a & \text{if } |a| < |b| \text{ and } ab > 0\,, \\ b & \text{if } |b| < |a| \text{ and } ab > 0\,, \\ 0 & \text{if } ab \leq 0\,. \end{cases} \tag{11}$$

Thus, if a and b have the same sign, the minmod function selects the one with smaller modulus, otherwise it returns 0. We sometimes refer here to the resulting flux limiter method as the *Minmod scheme*.

The Superbee flux limiter. It is well known that the Minmod scheme is, despite the improvement compared to the uncombined basic schemes, quite dissipative at discontinuities. To improve there the accuracy of the Minmod scheme, it is possible to choose a higher modulus near discontinuities. The resulting *Superbee scheme* is defined by using

$$\Phi(\Theta) \;=\; \max(0, \min(1, 2\Theta), \min(2, \Theta)) \;=\; \begin{cases} 0 & \text{if } \Theta \leq 0 \\ 2\Theta & \text{if } 0 < \Theta \leq \frac{1}{2} \\ 1 & \text{if } \frac{1}{2} < \Theta \leq 1 \\ \Theta & \text{if } 1 < \Theta < 2 \\ 2 & \text{else} \end{cases} \tag{12}$$

However, a problem with the Superbee method is that a smooth hump can be sharpened, such that stair-like artefacts are introduced into the numerical solution.

The MC flux limiter. Another possibility is to choose the so-called *monotonised central-difference limiter* (MC limiter), which compares the second-order central difference with twice the forward and twice the backward difference. It is defined as follows:

$$\Phi(\Theta) \;=\; \max(0, \min(1+\Theta)/2, 2, 2\Theta) \;=\; \begin{cases} 0 & \text{if } \Theta \leq 0 \\ 2\Theta & \text{if } 0 < \Theta \leq \frac{1}{3} \\ \frac{1+\Theta}{2} & \text{if } \frac{1}{3} \leq \Theta < 3 \\ 2 & \text{else} \end{cases} \tag{13}$$

The resulting *MC scheme* is known to be a reasonable default choice for HR-TVD schemes that does not introduce much blurring at discontinuities, nor stair-like artefacts in smooth solution parts, cf. the discussion in [20].

3 Fuzzy Logic

In this section we review the notions from FL that we need for constructing FL-based flux limiters. For comprehensive textbooks on FL, see e.g. [12, 29]. The objective of this section is in particular to describe the building blocks of a fuzzy system as shown in Figure 3. It consists of four components: A knowledge base, a fuzzifier, an inference engine, and a defuzzifier. The *knowledge base* contains a set of *fuzzy rules*, which are production rules

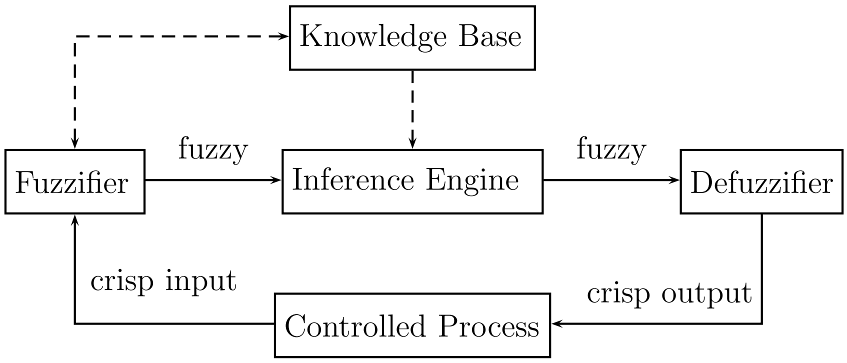


Figure 3: Generic model of a fuzzy controller.

of the form "IF A then B". A convenient way is to represent fuzzy sets by a (set of) parameterised function. Typical examples are monotonic decreasing functions, piecewise linear functions, exponential functions, or symmetric triangular functions. Moreover, it is common to attach a name to each fuzzy set and to identify the set by its name, resulting in more natural rules. The *fuzzifier* gets input values from the process to be controlled and maps them to fuzzy sets, thereby activating the rules. The *inference engine* evaluates the active rules and handles the way in which rules are combined, resulting in a fuzzy output set. Finally, the *defuzzifier* transforms the fuzzy output set to a crisp output value.

Several types of fuzzy systems that follow this general model have been proposed in the literature, among which the most important are: (i) *Mamdani* controller [32], (ii) *Takagi and Sugeno* controller [33], and (iii) *Tsukamoto* controller [31]. The differences lie in the consequences of the fuzzy rules, and thus their aggregation and defuzzification procedures differ accordingly. Within our setting, we will follow Mamdani's approach, which is the most intuitive one and most commonly used.

Fuzzy rules can either be formulated by an expert or be extracted from given data. Once all parameters have been fixed, a fuzzy system defines a nonlinear mapping, called *control function*, from the input domain to the output domain. A given control function can be modified by changing the parameters of the underlying fuzzy sets, or by modifying the rules of the underlying knowledge base. Indeed, a typical step in the development of a fuzzy system consists of *tuning* or optimising the parameters of the controller after a first prototype was specified. One particularly simple but powerful adaptation consists of applying *hedges* or *modifiers*, which are operations on fuzzy sets which change their shape while preserving their main characteristics. Within this paper we consider the standard hedges given by the *contrast operator*, the *dilation operator*, and the *concentration operator*, as specified below.

3.1 Fuzzy Sets

Fuzzy logic is directly derived from fuzzy set theory. The general idea is to allow not only for full membership or no membership of an element in a set, but also for *partial membership*. The membership of an element x in a set can be described by using a *characteristic* *function* $\mu(x)$. Then the concept of FL means that instead of allowing for a given x only full membership $\mu(x) = 1$ or no membership $\mu(x) = 0$, to also allow for partial membership i.e. $\mu(x) \in [0, 1]$. A set represented by such a membership function is called *fuzzy set*:

Definition 1 (Fuzzy Set) *Let G be a (classic) set, and μ_A be a (classic) function mapping the set G to the interval $[0, 1]$:*

$$\mu_A : G \rightarrow [0, 1] \tag{14}$$

The set A of all pairs $(x, \mu_A(x))$ (where x in G varies and $\mu_A(x)$ is defined) is then called fuzzy set *over G*

$$A = \{(x, \mu_A(x)) \mid x \in G\} \tag{15}$$

μ_A *is called* membership function. *G is called* reference set *of the fuzzy set.*

We observe that a classic *crisp set* with $\mu(x) \in \{0, 1\}$ can be represented by a particular fuzzy set. Note that μ_A already determines A completely. If we understand μ_A as a set of pairs in the set theoretic sense, we formally have $\mu_A = A$. In the sequel we always denote the membership function of a fuzzy set A with μ_A, and will write μ_A instead of $\{(x, \mu_A(x))|x \in G\}$ when it is comfortable.
Given a fuzzy set, those individuals whose membership value is greater than zero and those whose membership value is equal to one are of particular interest:

Definition 2 (Support, Tolerance) *Let A be a fuzzy set over the reference set G. Then*

$$supp(A) := \{x \mid x \in G \text{ and } \mu_A(x) > 0\} \tag{16}$$

is called support *of the fuzzy set. The set*

$$\{x \mid x \in G \text{ and } \mu_A(x) = 1\} \tag{17}$$

is called tolerance *of the fuzzy set.*

Usually, the support set of a fuzzy set is an interval $[m_1, m_2]$ or a single point $\{m\}$. In the latter case, the fuzzy set is also called *singleton*. A fuzzy set whose support is empty is called *empty fuzzy set*.
A convenient way to represent the fuzzy set $(x, \mu_A(x))$ is to use a set of parameterised functions. The idea of the so-called LR representation is to have a basic representation for very simple fuzzy sets.

Definition 3 (LR Fuzzy Set) *Let A be a fuzzy set over G. A is called of type* LR*, if there exist reference functions L, R over $\mathbb{R}_0^+$ satisfying the following properties:*

- $L, R : \mathbb{R}_0^+ \rightarrow [0, 1]$
- $L(0) = R(0) = 1$

- L, R *are monotone decreasing on* $\mathbb{R}_0^+$

such that $\mu_A(x)$ *can be described as follows:*

$$\mu_A(x) = \begin{cases} L\left(\frac{m_1 - x}{\alpha}\right) & \text{for } x \in [a, m_1] \\ 1 & \text{for } x \in [m_1, m_2] \\ R\left(\frac{x - m_2}{\beta}\right) & \text{for } x \in [m_2, b] \\ 0 & \text{else} \end{cases} \tag{18}$$

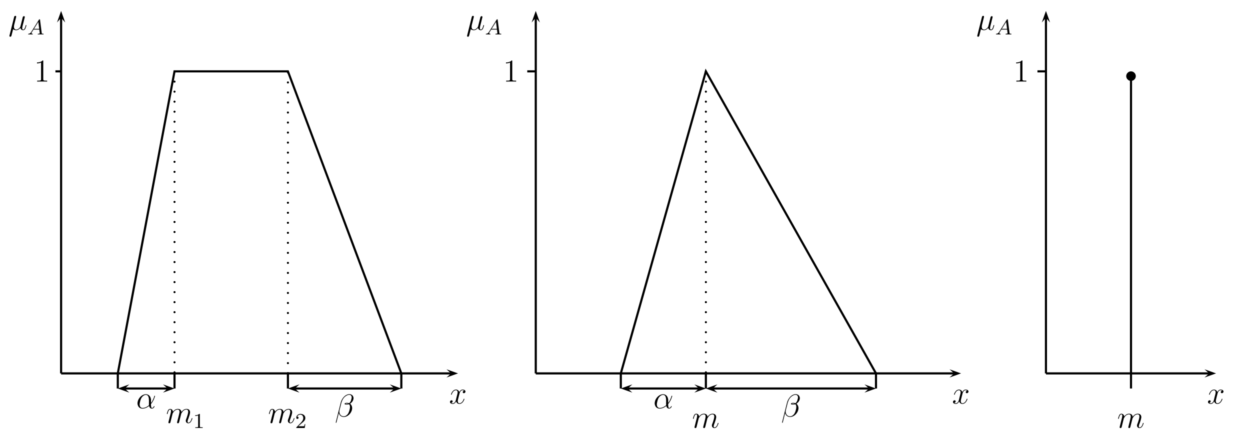


Figure 4: Examples for Fuzzy Sets. **Left.** A trapezoid fuzzy set. **Middle.** Triangular fuzzy set. **Right.** Singleton.

Typical representatives of LR fuzzy sets are shown in Figure 4. In what follows we will use:

- trapezoid(a, b, c, d) to represent a trapezoid fuzzy set with support $[a, d]$, $L(x) = R(x) = \max(0, 1 - x)$, $m_1 = b$, $m_2 = c$, $\alpha = b - a$, and $\beta = d - c$; cf. left-hand side of Figure 4.

- triangle(a, b, c) to represent a triangular fuzzy set with support $[a, c]$, $L(x) = R(x) = max(0,\ 1 - x)$, $m_1 = m_2 = b$, $\alpha = b - a$ and $\beta = c - b$; cf. middle of Figure 4).

- singleton(a) to represent a fuzzy set with support $\{a\}$ and which is 0 otherwise; cf. right-hand side of Figure 4.

3.2 Fuzzy Linguistic, Hedges and Fuzzy Rules

A general concept for classifying several concepts for one given parameter is provided by the fuzzy linguistic. Its basic elements are *linguistic variables* that describe a parameter of interest and *linguistic terms* that describe corresponding values:

Definition 4 (Linguistic Variable, Linguistic Term) *Let G be a reference set, label be the name of a fuzzy set and A be a fuzzy set over G. Then the pair $\langle label, A\rangle =: LT$ is called* linguistic term *over G.*
Let G be a parameter of interest with reference set G, label be the name of a fuzzy set, and $LT_1, \ldots, LT_n$ be linguistic terms over G. Then $\langle label, \{LT_1, \ldots, LT_n\}\rangle =: LV$ is called linguistic variable *with linguistic terms $LT_1, \ldots, LT_n$.*

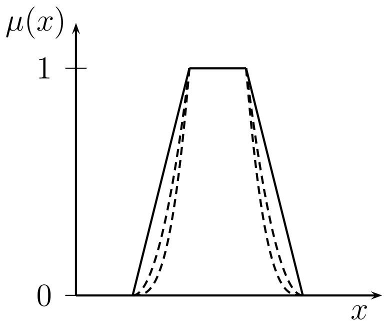
We write label(LV) and label(LT) to denote the label of a linguistic variable and a linguistic term, respectively. Linguistic terms and variables allow the identification of a fuzzy set by means of its label and results in more natural fuzzy rules.

Hedges. A *linguistic hedge* is an operation that modifies the meaning of a linguistic term by changing the membership function of the corresponding fuzzy set. More precisely, hedges can be seen as functions operating on membership functions of fuzzy sets. For our purpose, the following three hedges or modifier are of importance, cf. Figure 5. For their formal definition, let $A = \{(x, \mu_A(x)) \mid x \in G\}$ be again a fuzzy set over the set G:

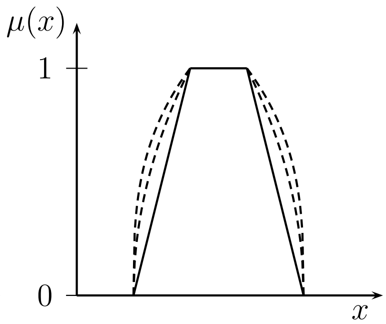
$$\text{Concentration operator} \;:\; CON_n(A) \;=\; \{(x, [\mu(x)]^n) \mid x \in U\} \tag{19}$$

$$\text{Dilation operator} \;:\; DIL_n(A) \;=\; \{(x, \sqrt[n]{\mu(x)} \mid x \in U)\} \tag{20}$$

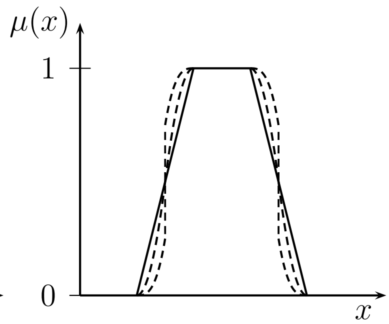
$$\text{Contrast operator} \;:\; INT_n(\mu(x)) \;=\; \begin{cases} 2\mu(x)^n \text{ if } \mu(x) < 0.5 \\ 1 - 2(1-\mu(x))^n \text{ else} \end{cases} \tag{21}$$



(a) Concentration Operator



(b) Dilation Operator



(c) Contrast Operator

Figure 5: Hedges for fuzzy sets.

Fuzzy Rules. Analogously to classic rule based systems, a fuzzy rule is represented as a production rule "If A then B", where A is a conjunction of linguistic terms over the input sets, possibly modified by a hedge, and B is a linguistic term over the output set, also possibly modified. Thus, by relying on fuzzy linguistic we derive the following definition:

Definition 5 (Fuzzy Rule, Fuzzy Knowledge Base) *Let $LV^{(1)}, \ldots, LV^{(n)}$ be linguistic variables with linguistic terms $LT_1^{(i)}, \ldots, LT_{m_i}^{(i)}$ over the reference set X_i, $1 \leq i \leq n$, characterising the input parameters. Moreover, let $LV^{(n+1)}$ be a linguistic variable with*

linguistic terms $LT_1^{(n+1)}, \ldots, LT_{m_{n+1}}^{(n+1)}$ *over the reference set* Y*, characterising the output parameter.*
Then an expression of the following form is called fuzzy rule*:*

$$R : IF\ LV^{(1)}\ is\ LT_{j_1}^{(1)}\ and\ \ldots\ and\ LV^{(n)}\ is\ LT_{j_n}^{(n)}\ THEN\ LV^{(n+1)}\ is\ LT_{j_{n+1}}^{(n+1)} \tag{22}$$

where $1 \leq j_i \leq m_i$ *selects a linguistic term of the linguistic variable* $LV^{(i)}$ *for each* $1 \leq i \leq n+1$.
Let $R_1, \ldots, R_n$ *be fuzzy rules. A finite set of fuzzy rules* $\mathcal{R} = R_1, \ldots, R_n$ *is called* fuzzy knowledge base*.* $\mathcal{R}$ *is called* consistent*, if there are no rules in* $\mathcal{R}$ *with the same antecedent but different succedents.*

Note that we only allow the combination of linguistic terms in the premises of the rules by means of the conjunction operator. This is not a restriction, because we can split a rule containing an disjunction in two separate rules. This simple restriction, however, structures the knowledge base and allows the evaluation of a rule just by applying the *and* operator for all premises.

3.3 Fuzzification and Fuzzy Inference

The inference mechanism of a fuzzy system operates on a set of inference rules and fuzzified input data. It works by first evaluating each single inference rule (*fuzzy inference*), and then combining their results (*aggregation*). The underlying idea of fuzzy inference is to understand a fuzzy rule "If A then B" as a fuzzy relation $R_{A\to B}$, and to use the so-called *compositional rule of inference* to map a given input fuzzy for a premise via the the relation to a corresponding fuzzy output set (the conclusion of the rule): $B' = A' \circ R_{A\to B}$.

Fuzzification. To be able to use fuzzy inference, the discrete input value x provided by the underlying application is used to obtain the modified input fuzzy set A', called *fuzzification.* This works by computing all membership values of the linguistic term corresponding to the linguistic variable of the input parameter. For a linguistic variable $LV^{(i)}$ with m_i linguistic terms we thus obtain a vector $\vec{v} \in [0, 1]^{m_i}$ of membership values.

Compositional Rule of Inference. In the classical case, a relation over A and B is a set of pairs $R \subset A \times B$ and the image of a set A under a relation R is defined as

$$R(A) = \{y \in B \mid \exists x\ :\ x \in A \wedge R(x, y)\} \tag{23}$$

Similarly, a fuzzy relation is defined as

Definition 6 (Fuzzy Relation) *Let* G_1, G_2 *be classical sets and* μ_A *be a function*

$$\mu_A : G_1 \times G_2 \to [0, 1] \tag{24}$$

The set

$$A = \{((x, y), \mu_A(x, y))\ \mid x \in G_1, y \in G_2\} \tag{25}$$

is called binary fuzzy relation *over* $G_1 \times G_2$*. As usual we write* $\mu_A(x, y)$ *to represent the fuzzy relation.*

In analogy to the image of a set under a relation, it is possible to define the image of a fuzzy set over a fuzzy relation. Moreover, by picking the maximal value for a fixed y, we can define a mapping. The $\wedge$ operator from the classical case is replaced by the intersection operator min. As the intersection operator is defined only for fuzzy sets over the same reference set, $\mu_A(x)$ is extended to $\mu_A^*(x,y)$, known as *extension principle*.

Definition 7 (Compositional rule of inference) *Let μ_R be a fuzzy relation over the reference sets G_1 and G_2 and let A be a fuzzy set over G_1. Then the image of R under A can be determined by the* compositional rule of inference*:*

$$R(A) = \sup_{x \in G_1} \{\min(\mu_A^*(x,y), \mu_R(x,y))\} \tag{26}$$

Thus, given an input value x' and a fuzzy rule of the form (22), the resulting fuzzy set is obtained by the following formula:

$$\begin{aligned} \mu_B(y) &= \sup_{x \in X} \min(\mu_A(x), R(\mu_A(x), \mu_B(y))) \qquad (27) \\ &= \min(\mu_A(x'), \mu_B(y)) \qquad (28) \end{aligned}$$

Aggregation. The compositional rule of inference allows the evaluation of a single fuzzy rule. To take several rules into accout, the standard approach is to individually evaluate the rules and to combine their results. This process is called *aggregation*.
Given r rules $R_1, \ldots, R_r$, for each j with $1 \leq j \leq r$ we evaluate R_j and obtain an individual fuzzy set μ_{R_j}. The resulting fuzzy set is computed by building the union of all individual fuzzy sets, which can be done with the max operator:

$$\mu_{res} = \max(\mu_{R_1}, \ldots, \mu_{R_r}) \tag{29}$$

3.4 Defuzzification

The result of applying all rules of the knowledge base is an aggregated fuzzy set $\mu_{res}(y)$ over the output reference set. The process to convert this into a crisp value is called *defuzzification*. The most commonly used method is the *centroid* or *centre of gravity* technique, which determines the centre of gravity of all resulting fuzzy output values. For a continuous aggregated fuzzy set, the center of gravity is given by the following formula

$$y' = \frac{\int\limits_S y \mu_B(y) dy}{\int\limits_S \mu_B(y) dy} \tag{30}$$

where S denotes the support of μ_B. In practice, the formula is often approximated. The centroid method can be specialised for the situation that the output fuzzy sets are singletons. For each rule with activation degree h_i the corresponding output value is multiplied by h_i. The products $h_i y_i$ are added together and divided by the sum of all activation degrees, formally

$$y_{res} = \frac{\sum_{i=1}^{m} h_i y_i}{\sum_{i=1}^{m} h_i} \tag{31}$$

4 Constructing Flux Limiters with Fuzzy Logic

HR schemes are described by a flux limiter function $\Phi : \mathbb{R} \to [0, 2]$ taking the smoothness as input and returning a limiter value. All the steps that process the input and finally give the limiter value are now modeled using the fuzzy controller. Following the described work flow, the following steps need to be specified in order to define the fuzzy flux limiter:

(F1) Choosing the input and output parameter.

(F2) Determination of the domain of the input and output parameter, also called *scaling of the linguistic variables.*

(F3) Partitioning of the linguistic variables by defining the linguistic terms together with their membership functions.

(F4) Specification of the knowledge base.

Let us stress that the control function is not formulated as a mathematical formula, but implicitly defined by the parameters of the fuzzy controller and the underlying rule base. Therefore, there is no need to write down any code during the experiments, as all computations are covered by the FL framework.
In the remainder of this section we briefly discuss the items above with the exception of (F3) and (F6) that are already clarified in the previous section.

(F1) and (F2): Input and Output Values and Scaling As a single input value we take as usual the smoothness Θ, cf. (9). Note that we can restrict the domain of the input variable to a subinterval of $\mathbb{R}$ that covers only the "interesting" parts of the input variable, i.e. those parts in which the membership functions of the linguistic terms change. Values outside of the input domain are mapped to the corresponding boundary values.

Linguistic terms. We classify the smoothness with respect to three descriptors, centered around the smoothness measure 1, which stands for a smooth solution. The second linguistic term we use identifies the situation where the data are extremal, given by a negative smoothness measure. Finally, we introduce a third linguistic term classifying situations in which the data are steep.

Input domain. As we do not want to classify the input data further, we can restrict the input domain to an interval $[a, b]$, where a is smaller than zero and b is greater than 1.

Note that there is no need to restrict the input domain to such an interval; however, this will ease the presentation.

Output domain. As output parameter we choose the limiter value Φ which we want to compute. For the schemes we want to consider we know already that a domain $[0, 2]$ for Φ is reasonable.

(F3): Partitioning of the Linguistic Variables Using the technique of expert rules means that an expert provides a classification of input and output parameters into concepts. In our scenario, for the output variable we have two basic methods to choose from, namely the upwind method and the Lax-Wendroff method. Each basic method is modeled by means of a linguistic variable, which we call "UP" and "LW". For the input variable, we define three linguistic variables "bumpy", "smooth", and "excursive". For the linguistic term "smooth", we know already that the concept is fully compatible for a smoothness measure $\Theta = 1$. Moreover, "bumpy" is thought to classify situations in which the data have a maximum or minimum, given by a negative smoothness measure. Finally, we want to use "excursive" to classify situations where the data are steep.
For the border fuzzy sets we choose trapezoid fuzzy sets, and as we are interested in a combination of the numerical methods we choose neighboring fuzzy sets to maximally overlap. In these situations, several rules will fire and we will thus obtain a mixture of both methods as overall result.
As this is a key issue in our work, let us illustrate at this point our proceeding at hand of an example where all three proposed linguistic terms are used.

Example 1 *In the framework of flux limiters the upwind method is characterised by a flux limiter value of* $\Phi = 0$*, and the Lax-Wendroff method is characterised by setting the limiter value* Φ *to* 1*. Consequently we design two linguistic terms "UP" and "LW" for the output parameter "flux limiter", where "UP" attains* 1 *for* $\Phi = 0$*, and "LW" for* $\Phi = 1$*.*
A possible choice for the parameter "flux limiter" giving a maximal overlapping of neighbouring fuzzy sets is shown in Figure 6.

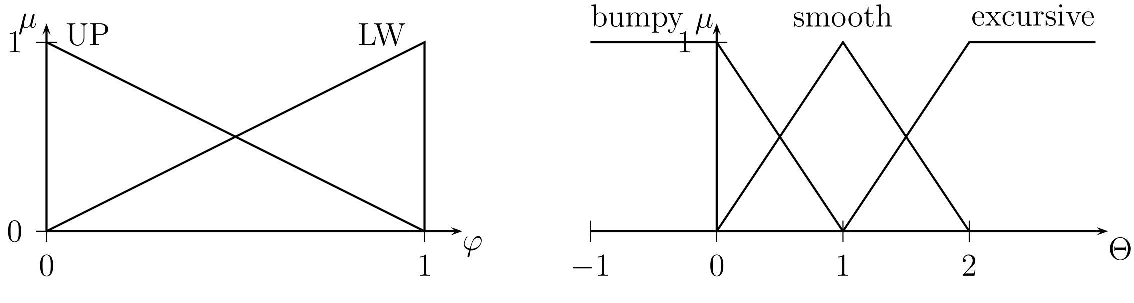


Figure 6: **Left.** Fuzzy sets for the parameter "flux limiter". **Right.** Linguistic terms for the parameter "smoothness".

We now turn our attention to the modeling of the input parameter, the smoothness Θ*. Recall that the smoothness is given by building the fraction of the two cell differences, see*

(9). If these values are identical, then Θ *equals one and we know that the solution is* smooth. *We may choose a triangle as shape for this fuzzy set.*
When departing from $\Theta = 1$ *there are now two cases. We know that if* Θ *is negative the solution has a local extremum, which we want to classify by means of the concept "bumpy". As this predicate is fulfilled for all negative* Θ*, we may choose as shape of the fuzzy set a trapezoid. If* Θ *is greater than 1, we assume the data to be steep. Consequently we design a linguistic term "excursive", classifying this situation. Again, we may choose maximal overlapping between the neighboring terms. A possible choice for the parameter "smoothness" is shown in Figure 6.*

(F4): Specification of the knowledge base The next and central step consists of a specification of the rule base. As we rely on a single input parameter "smoothness", our rules will have the following form:

$$\text{If smoothness is } LT_{i_1}^{\text{smoothness}} \text{ then flux limiter is } LT_{i_{n+1}}^{\text{flux limiter}} \tag{32}$$

where $LT^{smoothness}$ is either "smooth", "excursive", or "steep", and $LT_{i_{n+1}}^{\text{flux limiter}}$ is "LW" or "UP".
To obtain a complete rule base, i.e. a rule base for which for every input a nonzero output is computed, we have to specify one rule for each possible input situation. Consequently we have to define three rules.

In our application there is only one input parameter "smoothness" for the measured smoothness at a computational point. This is described by – at most – three linguistic terms "bumpy", "smooth", and "excursive", compare Example 1. Consequently we design three rules with an antecedent consisting of one of these descriptors. For the succedent we have to choose between the linguistic terms "LW" and "UP". The only situation where we want to choose the "LW" method is when the solution is smooth.
The resulting rule base is shown in Table 1. Let us stress that this is the complete set of rules we need within this paper. We may even need only two of these rules.

If **smoothness** is **bumpy** then **limiter** is **UP**
If **smoothness** is **smooth** then **limiter** is **LW**
If **smoothness** is **excursive** then **limiter** is **UP**

Table 1: Maximal rule base for our application.

5 Fuzzy Formulation of Classic Schemes

We are now ready to show that standard schemes can be given a novel interpretation using FL. Rebuilding existing schemes in our framework has several advantages: (i) It gives us the possibility to compare these schemes at the level of fuzzy logic; (ii) the intuition behind a numerical scheme becomes explicitly expressed in the rule base, and thus presented in a clear way; (iii) it allows for extensions and modifications of these schemes.

5.1 The Minmod Scheme

As shown in Figure 7 the control function of the Minmod scheme, cf. (10), can be classified into three parts: Two boundary parts in which the flux limiter function is constant, and an intermediate part in which the limiter function varies. In the left boundary part it is constantly zero, and in the right boundary part it is constantly one. The middle part is given by a smooth linear transition between the left and the right boundary part.

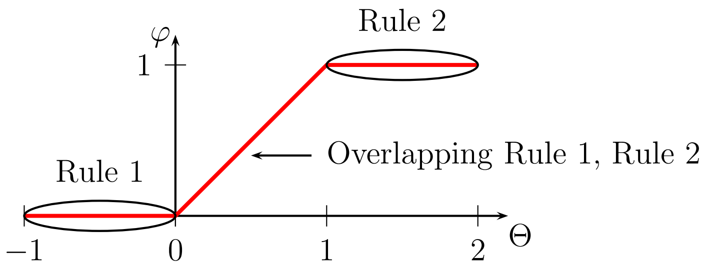


Figure 7: Control function of the Minmod scheme.

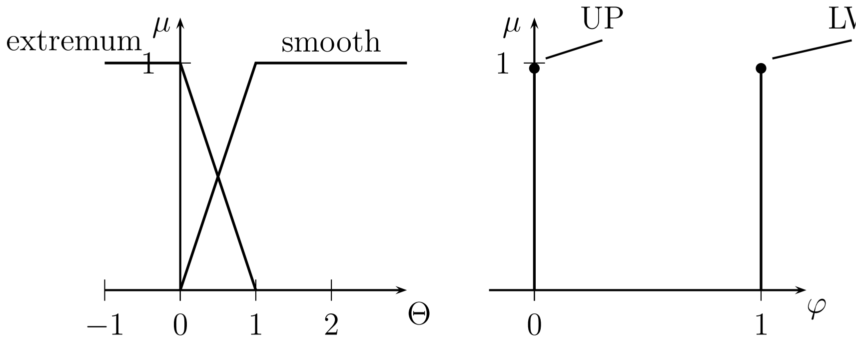
Explaining the Minmod scheme to a novice one could summarise the method via the following two *expert rules*: If the data are smooth use the Lax-Wendroff scheme, and the Upwind scheme otherwise. The first key situation is given by smooth initial data, i.e. in the region where the flux limiter function is greater or equal to one. The other key situation is identified by a smoothness measure of zero. Between these key situations we let fuzzy reasoning determine the value of the flux limiter.

We define two linguistic variables, one called “smoothness” and representing the input parameter, the other called “flux”, representing the output parameter of the fuzzy controller. Moreover, each linguistic variable is partitioned by means of two linguistic terms associated to it. For the input variable we design two classifier “extremum” and “smooth”, representing the mentioned key situations. “extremum” has the support set ranging from zero to the minimal admissible smoothness input value, which we fix to be -1. “smooth” has support set ranging from 1 to the maximal admissible smoothness value, which we fix to be 2. Values which are smaller or larger as the admissible input values are mapped to the minimal or maximal admissible input value, respectively.

For the output parameter we define two linguistic terms “UP”, representing the Upwind method, and “LW” for the Lax-Wendroff method. Both are modeled by means of singleton sets with support set zero and one, respectively. Choosing singleton sets for the output variables results in a very simple interpolation in the defuzzification process, namely a linear combination.

The last part of our modeling addresses the rule base consisting of two rules, the first assigning the linguistic output term “UP” to the input situation “extremum”, the other assigning the output situation “LW” to the input situation “smooth”. The fuzzy sets and the rule base are shown in Figure 8.

We conclude these developments via the following theorem.



(a) Input Variable (b) Output Variable

If **smoothness** is **extremum** then **limiter** is **UP**
If **smoothness** is **smooth** then **limiter** is **LW**

(c) Rule base

Figure 8: Fuzzy sets and rule base for the Minmod limiter.

Theorem 1 *The numerical fuzzy scheme given by the input linguistic input variable "smoothness" with linguistic terms* $\langle$ *"extremum",* $trapez(-1,-1,0,1)\rangle$ *and* $\langle$ *"smooth",* $trapez(0,1,2,2)\rangle$*, and the output variable "flux" with linguistic terms* $\langle$ *"UP",* $singleton(0)\rangle$ *and* $\langle$ *"LW",* $singleton(1)\rangle$*, the rule base consisting of the following two rules*

- R_1*: If smoothness is "extremum" then "flux" is "UP"*
- R_2*: If smoothness is "smooth" then "flux" is "LW"*

the inference engine $\langle Mamdani, \bigcup\rangle$ *and the center of gravity defuzzification method is equivalent to the Minmod scheme.*

Proof. We consider the three cases that either only R_1 or only R_2 is active, or that both R_1 and R_2 are active.

Case $x \leq 0$: If the input parameter $x \leq 0$, then only the rule R_1 is active, with $\mu_{\text{extremum}}(x) = 1$. As our inference engine $\langle Mamdani, \bigcup\rangle$ is well behaved, we get as output fuzzy set exactly the singleton set $singleton(0)$ associated to "UP", which has center of gravity 0.

Case $x \geq 1$: Similar to the first case there is only the rule R_2 active, with $\mu_{\text{smooth}}(x) = 1$, and we obtain as output fuzzy set the singleton set $singleton(1)$ associated to "LW", which has center of gravity 1.

Case $0 < x < 1$: We have $\mu_{\text{extremum}}(x) = 1 - x$ and $\mu_{\text{smooth}(x)} = x$. The application of the first rule (R_1) results in the individual output set

$$\mu_{Res,R_1}(y) = \min(\mu_{\text{extremum}}(x), \mu_{UP}(y)) \tag{33}$$

$$= \begin{cases} 1-x & \text{if } y = 0 \\ 0 & \text{else} \end{cases}, \tag{34}$$

and the application of the second rule (R_2) in

$$\mu_{Res,R_2}(y) = \min(\mu_{\text{smooth}}(x), \mu_{LW}(y)) \tag{35}$$

$$= \begin{cases} x & \text{if } y = 1 \\ 0 & \text{else} \end{cases} . \tag{36}$$

Aggregation of their results by means of $\bigcup$ yields

$$\mu_{Res,R_1,R_2}(y) = \begin{cases} x & \text{if } y = 0 \\ 1 - x & \text{if } y = 1 \\ 0 & \text{else} \end{cases} . \tag{37}$$

Defuzzification μ_{Res,R_1,R_2} by means of the center of gravity method for singletons gives the overall result

$$y_{res} = \frac{x \cdot 0 + (1-x) \cdot 1}{x + 1 - x} = 1 - x \tag{38}$$

In each situation the flux limiter thus agrees with the flux limiter of the Minmod scheme, hence both schemes are equivalent.

q.e.d.

Let us stress that we only need two rules in the rule base to construct the Minmod scheme.

5.2 The Superbee Scheme

The second scheme we consider is the Superbee scheme. Its control function is shown in Figure 9, compare (12). Compared to the Minmod scheme there are the following

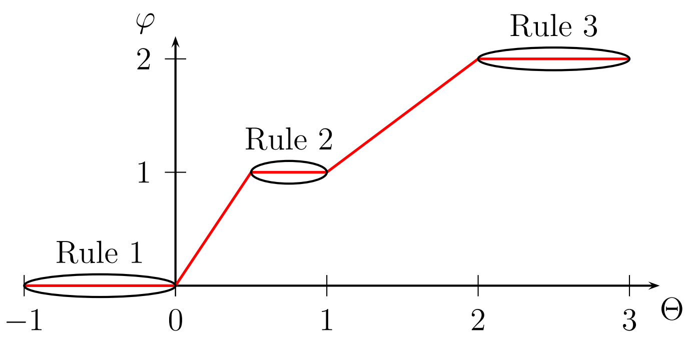
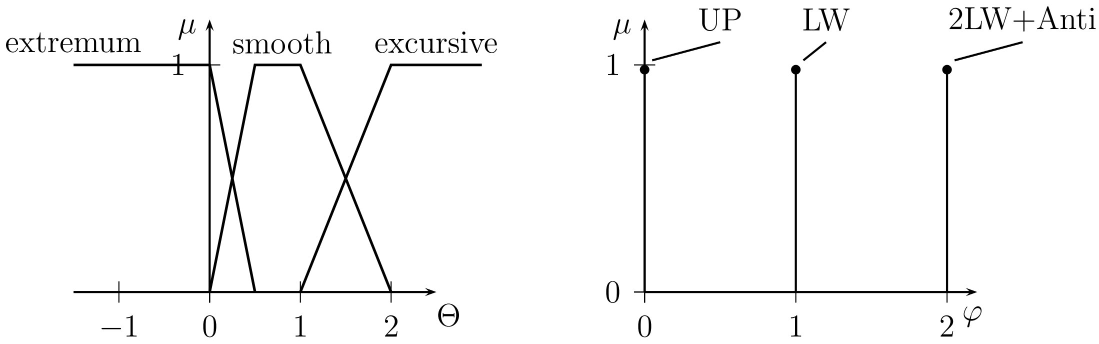


Figure 9: The Superbee limiter

differences: (i) The flux limiter function varies in the interval $[0, 2]$, we thus select the domain of the input variable to the interval $[-1, 3]$; (ii) the range of the flux limiter function

is $[0, 2]$, we thus extend the domain of our output parameter "flux" to the interval $[0, 2]$; (iii) there are three key situations in which the flux limiter function remains constant.
Similar to the Minmod scheme we define two linguistic terms "extremum" and "smooth" for the input parameter, and two linguistic terms "UP" and "LW" for the output parameter. Moreover, we see that the maximal output flux is two, which gives rise to an additional linguistic term for the output variable, which we call "2LW+Anti". We also define a new linguistic input parameter "excursive" as a concept identifying the situation in which "2LW+Anti" shall be selected. The names are motivated by the fact that a flux limiter value of two corresponds to the application of twice the Lax-Wendroff flux plus once the Upwind flux in backward direction (antiflux). We assign to "2LW+Anti" a singleton fuzzy set for $y = 2$.
Because of the new classification of our input data, we have to change the shape of the linguistic input term "smooth" and to define a fuzzy set for the linguistic term "excursive" when comparing to the setting used for the Minmod scheme. As tolerance of the fuzzy set attached to "smooth" we choose the interval $[0.5, 1]$, and as tolerance set of the fuzzy set attached to "excursive" we select the interval $[2, 3]$. As before we choose fully overlapping fuzzy sets as we want two rules to be active in the intermediate situation.
Finally, we take as rule base the rule base of the Minmod scheme and extend it by one rule which assigns the output value "2LW+Anti" to the input situation "excursive". The fuzzy sets for the Superbee scheme and the rule base is shown in Figure 10. The curious reader will already suspect that the resulting scheme can be made equivalent to the Superbee scheme:



(a) Input Parameter (b) Output Parameter

If **smoothness** is **extremum** then **limiter** is **UP**
If **smoothness** is **smooth** then **limiter** is **LW**
If **smoothness** is **excursive** then **limiter** is **2LW+Anti**

(c) Rule base

Figure 10: Fuzzy sets and rule base for the Superbee limiter.

Theorem 2 *The numerical fuzzy scheme given by the input linguistic input variable "smoothness" with linguistic terms* $\langle$ *"extremum"*, $trapez(-1, -1, 0, 0.5)\rangle$ *and* $\langle$ *"smooth"*, $trapez(0, 0.5, 1, 2)\rangle$, $\langle$*"excursive"*, $trapez(1, 2, 3, 3)\rangle$, *and the output variable "flux" with linguistic terms*

⟨ "UP", singleton(0)⟩, ⟨ "LW",singleton(1)⟩, and ⟨ "2LW+Anti",singleton(2)⟩, the rule base consisting of the following three rules

- *R_1: If smoothness is "extremum" then "flux" is "UP"*
- *R_2: If smoothness is "smooth" then "flux" is "LW"*
- *R_3: If smoothness is "excursive" then "flux" is "2LW+Anti"*

the inference engine $\langle Mamdani, \bigcup \rangle$ and the center of gravity defuzzification method is equivalent to the Superbee scheme.

Proof. We consider the cases that either only R_1 or only R_3 are active, that both R_1 and R_2 are active, and that both R_2 and R_3 are active, respectively.

Case $x \leq 0$, $x \geq 2$: As in the Minmod case.

Case $0 < x < 0.5$: We have $\mu_{\text{extremum}}(x) = 1 - 2x$ and $\mu_{\text{smooth}}(x) = 2x$. Consequently we get two output fuzzy sets

$$\mu_{Res,R_1}(y) = \min(\mu_{extremum}(x), \mu_{UP}(y)) \tag{39}$$

$$= \begin{cases} 1 - 2x & \text{if } y = 0 \\ 0 & \text{else} \end{cases} \tag{40}$$

$$\mu_{Res,R_2}(y) = \min(\mu_{smooth}(x), \mu_{LW}(y)) \tag{41}$$

$$= \begin{cases} 2x & \text{if } y = 1 \\ 0 & \text{else} \end{cases} \tag{42}$$

giving rise to the overall fuzzy set

$$\mu_{Res,R_1,R_2}(y) \begin{cases} 1 - 2x & \text{if } y = 0 \\ 2x & \text{if } y = 1 \\ 0 & \text{else} \end{cases} \tag{43}$$

Defuzzification by means of the center of gravity method provides the result

$$\mu_{res}(y) = \frac{(1 - 2x) \cdot 0 + 2x \cdot 1}{1 - 2x + 2x} = 2x \tag{44}$$

Case $1 < x < 2$: We have $\mu_{\text{smooth}}(x) = 1 - (x - 1)$ and $\mu_{\text{excursive}}(x) = x - 1$. Evaluation of the rules R_2 and R_3 and aggregation of their individual results yields the fuzzy set

$$\mu_{res}(y) = \begin{cases} 1 - (x - 1) & \text{if } y = 1 \\ x - 1 & \text{if } y = 2 \\ 0 & \text{else} \end{cases} \tag{45}$$

which is defuzzified to

$$\mu_{res}(y) = \frac{(1 - (x - 1)) \cdot 1 + (x - 1) \cdot 2}{1 - (x - 1) + x - 1} = 2 - x + 2x - 2 = x \tag{46}$$

In each situation the control function thus agrees with the flux limiter of the Superbee scheme, hence both schemes are equivalent. *q.e.d.*

5.3 The MC Scheme

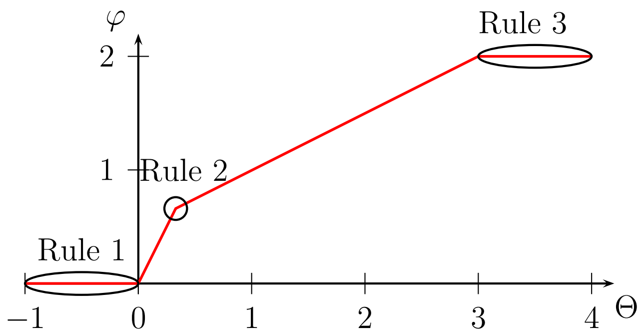


Figure 11: MC flux limiter

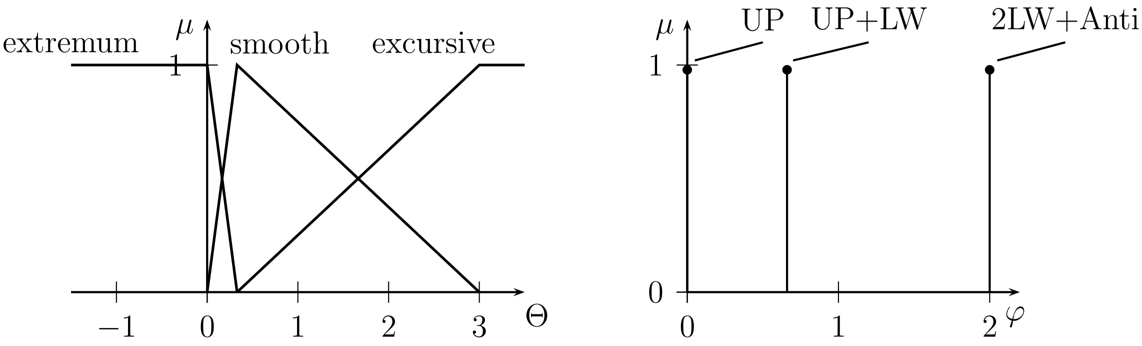
Let us now consider the MC scheme, whose flux limiter function is shown in Figure 11, where the limiter function is defined as in (13). To model the MC limiter within the framework of fuzzy logic, we follow the approach from the previous two sections which has shown to be successful. We identify as key situations those situations at which the gradient of the flux limiter function does not change. Note that in contrast to the previous two schemes, where the fuzzy sets for the output parameter where classified by singletons at integer values zero, one, and two, we identify this time the locations zero, $\frac{2}{3}$, and two. Thus from our basic scheme only the Upwind scheme is directly represented as linguistic term of the linguistic output variable, the other two are already combinations of Upwind and Lax-Wendroff method. The fuzzy sets and the rule base are shown in Figure 12. It is quite surprising that the natural encoding of our expert rules directly results in the MC scheme:

Theorem 3 *The numerical fuzzy scheme given by the input linguistic input variable "smoothness" with linguistic terms* $\langle$ *"extremum"*, $trapez(-1,-1,0,\frac{1}{3})\rangle$ *and* $\langle$ *"smooth"*, $triangle(0,\frac{1}{3},3)\rangle$, $\langle$*"excursive"*, $trapez(\frac{1}{3},3,5,5)\rangle$, *the output variable "flux" with linguistic terms* $\langle$ *"UP",singleton*$(0)\rangle$, $\langle$ *"UP+LW"*, $singleton(\frac{2}{3})\rangle$, *and* $\langle$*"2LW + Anti"*, $singleton(2)\rangle$ *the rule base consisting of the following three rules*

- R_1*: If smoothness is "extremum" then "flux" is "UP"*
- R_2*: If smoothness is "smooth" then "flux" is "UP+LW"*
- R_3*: If smoothness is "excursive" then "flux" is "2LW+Anti"*

the inference engine $\langle Mamdani, \bigcup\rangle$, *and the center of gravity defuzzification method is equivalent to the MC scheme.*

Proof. We consider the cases that either only R_1 or only R_3 are active, that both R_1 and R_2 are active, and that both R_2 and R_3 are active, respectively.



(a) Input Parameter

(b) Output Parameter

If **smoothness** is **extremum** then **limiter** is **UP**
If **smoothness** is **smooth** then **limiter** is **UP+LW**
If **smoothness** is **excursive** then **limiter** is **2LW+Anti**

(c) Rule base

Figure 12: Fuzzy sets and rule base for the MC limiter

Case $x \leq 0$, $x \geq 3$: Similar to the Minmod case.

Case $0 < x \leq \frac{1}{3}$: We have $\mu_{\text{extremum}}(x) = 1 - 3x$ and $\mu_{\text{smooth}}(x) = 3x$. Consequently we get two output fuzzy sets,

$$\mu_{res,R_1}(y) = \begin{cases} 1 - 3x & \text{if } y = 0 \\ 0 & \text{else} \end{cases} \tag{47}$$

$$\mu_{res,R_2}(y) = \begin{cases} 3x & \text{if } y = \frac{2}{3} \\ 0 & \text{else} \end{cases} \tag{48}$$

whose aggregation defines the overall fuzzy set

$$\mu_{res,R_1,R_2}(y) = \begin{cases} 1 - 3x & \text{if } y = 0 \\ 3x & \text{if } y = \frac{2}{3} \\ 0 & \text{else} \end{cases} \tag{49}$$

Thus defuzzification yields

$$\frac{(1 - 3x) \cdot 0 + 3x \cdot \frac{2}{3}}{1 - 3x + 3x} = 2x \tag{50}$$

Case $\frac{1}{3} \leq x < 3$: We have $\mu_{\text{smooth}}(x) = 1 - (\frac{3}{8}(x - \frac{1}{3}))$ and $\mu_{\text{smooth}}(x) = \frac{3}{8}(x - \frac{1}{3})$. Consequently we get two output fuzzy sets,

$$\mu_{res,R_1} = \begin{cases} 1 - (\frac{3}{8}(x - \frac{1}{3})) & \text{if } y = \frac{2}{3} \\ 0 & \text{else} \end{cases} \tag{51}$$

$$\mu_{res,R_3} = \begin{cases} \frac{3}{8}(x - \frac{1}{3}) & \text{if } y = 2 \\ 0 & \text{else} \end{cases} \tag{52}$$

whose aggregation defines the overall fuzzy set

$$\mu_{res,R_1,R_2}(y) = \begin{cases} 1 - (\frac{3}{8}(x - \frac{1}{3})) & \text{if } y = \frac{2}{3} \\ \frac{3}{8}(x - \frac{1}{3}) & \text{if } y = 2 \\ 0 & \text{else} \end{cases} \tag{53}$$

Thus defuzzification yields

$$\frac{(1 - \frac{3}{8}(x - \frac{1}{3})) \cdot \frac{2}{3} + \frac{3}{8}(x - \frac{1}{3}) \cdot 2}{1 - (\frac{3}{8}(x - \frac{1}{3})) + \frac{3}{8}(x - \frac{1}{3})} = \frac{2}{3} - \frac{1}{4}(x - \frac{1}{3}) + \frac{6}{8}(x - \frac{1}{3}) = \frac{1 + x}{2} \tag{54}$$

q.e.d.

Remarks. We have seen that one can easily model the Minmod scheme, the MC scheme, and the Superbee scheme in the framework of fuzzy logic. Only two respectively three linguistic terms and the same number of fuzzy rules were already sufficient for this. Even more, for all schemes the main rules in the rule base were identical, with one additional rule added for the MC and the Superbee scheme.
As a technical remark, the flux computation based on FL is relatively complex from a numerical point of view. Therefore the FL-based formulation can be a little bit slower with respect to runtime compared to the classic formulation. However, it is possible to compute the values of the control function beforehand and to store them in a lookup table. Then the output values for a specific combination of input values can directly be determined in $O(1)$ without further computation. As a consequence, the computation of the control function will be for free and the resulting schemes faster.

6 Improvement of Fuzzy Flux Limiter Schemes

We now show how our framework allows for the improvement of the numerical schemes considered so far. To this end, we aim to enhance the accuracy of a given numerical scheme by modifying the input parameters of the controller via the application of hedges. This is done for several characteristic situations encountered when solving HCLs. The presentation is focused on the improvement of the MC scheme: This is already a reasonable method and we want to show that also a good method can be improved significantly by our approach.

Analogous improvements of the Minmod and the Superbee scheme are also mentioned, yet not in as much detail as for the MC scheme.

Our method for the improvement of fuzzy flux limiter (FFL) schemes works by systematically considering combinations of applications of the standard operators *concentration*, *dilation*, and *contrast* to the linguistic input parameters. To obtain a feasible method and a finite search space we restrict the parameter n of the operators to values of $0, 2, 4, 6, 8, 10$. Each obtained controller is then evaluated for characteristic test data and the best numerical scheme is returned.

Let us note that more complex means of optimisation are possible, for instance the use of sophisticated learning techniques to adapt the control system automatically [4]. However, this goes much beyond the scope of this work.

The discrete setting. In our experiments we use a time step size $\Delta t = 0.0025$ and a spatial grid size $\Delta x = 0.01$. The error measurements are always given in terms of the discrete L_1-error. In the case of the sine wave experiments described in the following we take into account the staircaising effect by identifying the number of neighboring cells which come very close to the maximum of the sine wave. We measure the number of points adjoining the maximum and whose cell value differs from the maximum less than 0.01; these we denote as stairs.

6.1 Linear Advection

In our first experiment we deal with the specific linear advection equation

$$u(x,t)_t + u(x,t)_x \;=\; 0 \tag{55}$$

We consider two characteristics test cases, *the box function test* and *the sine wave test*. The first setting deals with the main difficulty arising in solutions of hyperbolic PDEs, namely discontinuous solution features. In the second test, a discretised sine wave signal is evolved under periodic boundary conditions, addressing the approximation of smooth solutions. The exact solutions in these problems are given just by translating the initial signals in time. Let us note, however, that in practical linear problems as e.g. in acoustics the solution can often not be determined analytically, and that the numerical resolution of linear problems can be more challenging than in the non-linear setting.

The box function test

The best improvement for modifying the MC scheme is obtained by applying the concentration operator with $n = 8$ to the linguistic term “extremum”, the concentration operator with $n = 6$ to the linguistic term “smooth”, and the dilation operator with $n = 8$ to the linguistic term “steep”. We see that the jump is approximated in a much sharper way, with an impressive improvement concerning the error, cf. Table 2. It is interesting but not surprising that adding numerical compression in this way leads to the modfied control function leaving the Sweby TVD region, cf. Figure 13(b).

Iterations	MC Error	MC mod. Error	Improvement
400	0.0323959	0.00880443	72.82%
800	0.0388851	0.0088456	77.25%
2000	0.0499126	0.00900397	81.96%
4000	0.0607585	0.00934881	84.61%

Table 2: **Box function test.** Original and modified MC scheme.

Iterations	Minmod Error	Minmod mod. Error	Improvement
400	0.0569887	0.0461998	18.93%
800	0.0725024	0.0581026	19.86%
2000	0.0993293	0.0787437	20.72%
4000	0.1257290	0.0992058	21.10%

Table 3: **Box function test.** Original and modified Minmod scheme.

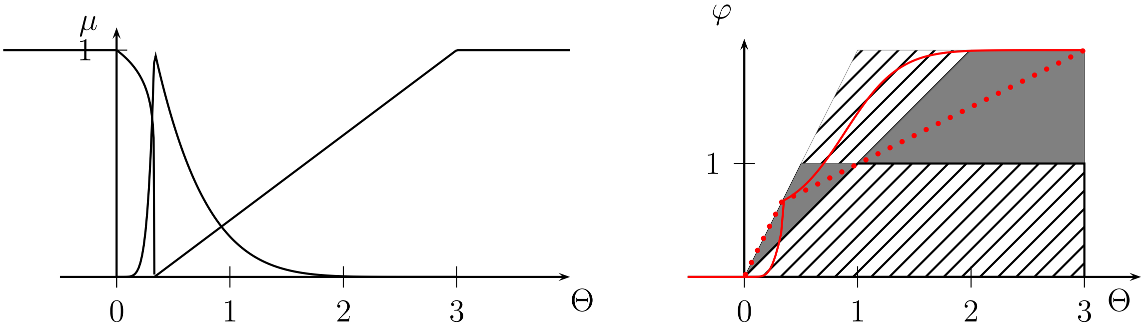
Iterations	Superbee Error	Superbee mod. Error	Improvement
400	0.0176138	0.0123212	30.05%
800	0.0181226	0.0124733	31.17%
2000	0.0182743	0.0127764	30.09%
4000	0.0182816	0.0131188	28.24%

Table 4: **Box function test.** Original and modified Superbee scheme.

The other schemes. The best improvement of the *Minmod scheme* ist obtained when applying the concentration operator with $n = 8$ to the linguistic term "extremum" and the dilation operator with $n = 2$ to the linguistic term "smooth". An analysis of the error is given in Table 3. Again, we see that the initial numerical method is drastically improved, around 20 percent for all iterations.
In the case of the *Superbee scheme*, the best improvement is obtained by the application of the dilation modifier with parameter value $n = 8$ to the linguistic term "smooth", and the dilation modifier with parameter value $n = 6$ to the linguistic term "steep". The improvement of about thirty percent is somewhat surprising as the basic Superbee scheme is expected to perform already very well for bumpy data.
It is also a somewhat surprising result of this experiment that the modified MC scheme performs slightly better than the Superbee scheme on discontinuous data, even after modification of the latter.

The sine wave test

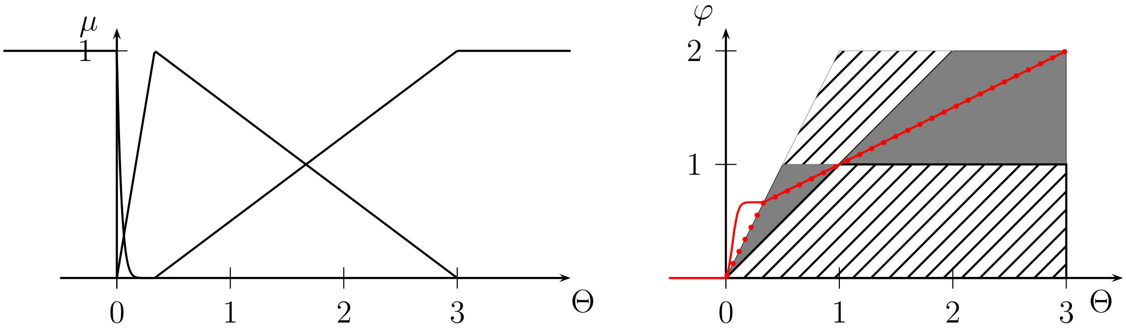
The most accurate FFL method on the basis of the *MC scheme* is obtained by applying the concentration operator with $n = 8$ to the linguistic term "extremum", and leaving the other linguistic terms unchanged. The fuzzy sets describing the input parameter and the resulting control function are shown in Figure 14(a) and Figure 14(b), respectively.



(a) Modified linguistic terms for the input variable

(b) Modified control function

Figure 13: **Box function test.** Modified linguistic terms and modified control function (red line) of the improved MC scheme.



(a) Modified linguistic terms for the input variable

(b) Modified control function

Figure 14: **Sine wave test.** Modified linguistic terms and modified control function of the improved MC scheme.

Let us briefly comment on the modified control function. We observe that it leaves to some part the TV stability region. However, this is not crucial for this test, as also the application of just the second-order Lax-Wendroff scheme we use as our higher-order component does not result in oscillations for smooth solutions, see e.g. [19].

The other schemes. The optimisation of the *Minmod scheme* yields the best improvement by applying the concentration operator with $n = 10$ to the linguistic term "extremum" and the dilation operator with the same parameter value to the linguistic term "smooth". An error analysis can be found in Table 6. We see that the basis scheme is drastically improved with respect to our test data, the improvement is about 40 per cent at $t = 1, 2$ and even about 50 per cent at $t = 5, 10$.
An undesirable property of the *Superbee scheme* is that it tends to introduce artificial stairs. We are especially interested to find a modification in which this effect is reduced. The modification we present here is given only by an application of the contrast operator with $n = 2$ to the linguistic term "steep", and letting the other linguistic terms untouched. An evaluation of the scheme can be found in Table 7. We see that in particular for the first iterations the modified scheme performs much better than the original scheme.

Iterations	MC Error	Stairs	MC mod. Error	Stairs	Improvement
400	0.00141052	5	0.00121663	5	13.75%
800	0.00246478	6	0.00225588	6	8.48%
2000	0.00532868	7	0.00499149	5	6.33%
4000	0.00948061	7	0.00916584	6	3.32%

Table 5: **Sine wave test.** Original and modified MC scheme.

Iterations	Minmod Error	Stairs	Minmod mod. Error	Stairs	Improvement
400	0.0067651	6	0.00418487	4	38.14%
800	0.0127694	7	0.00764521	4	40.13%
2000	0.031728	8	0.0165611	4	47.8%
4000	0.0561814	8	0.0268332	3	52.24%

Table 6: **Sine wave test.** Original and modified Minmod scheme.

Iterations	Superbee Error	Stairs	Superbee mod. Error	Stairs	Improvement
400	0.0048704	7	0.00350904	6	27.95%
800	0.00885299	8	0.00632758	7	28.53%
2000	0.0182049	11	0.0140664	7	22.73%
4000	0.0253891	12	0.0248731	7	2.03%

Table 7: **Sine wave test.** Original and modified Superbee scheme.

Discussion of first results

Let us consider the MC scheme and its improvements for the linear tests. The effects of the new control functions to our examples are shown in Figure 15.
We see that the sine wave – a smooth solution – is approximated almost perfectly. However, in this test case the improvement is not impressive as the MC scheme in its basic version already performs well for smooth data, see Table 5. For the box function test, an impressive improvement is gained.
Let us stress that this is qualitatively what can be expected also for the following tests. Smooth solutions can be made better depending on the accuracy obtained by the original MC scheme which is already in a reasonable range. For the Minmod scheme, for instance, such an improvement is potentially more significant since the original scheme is relatively dissipative. At discontinuous solution features, some accuracy can nearly always be gained. We summarise in the following these behaviours mainly by giving the L_1 errors as there is no surprise in plotted solutions; any improvements give qualitatively the same impression as indicated via Figure 15.

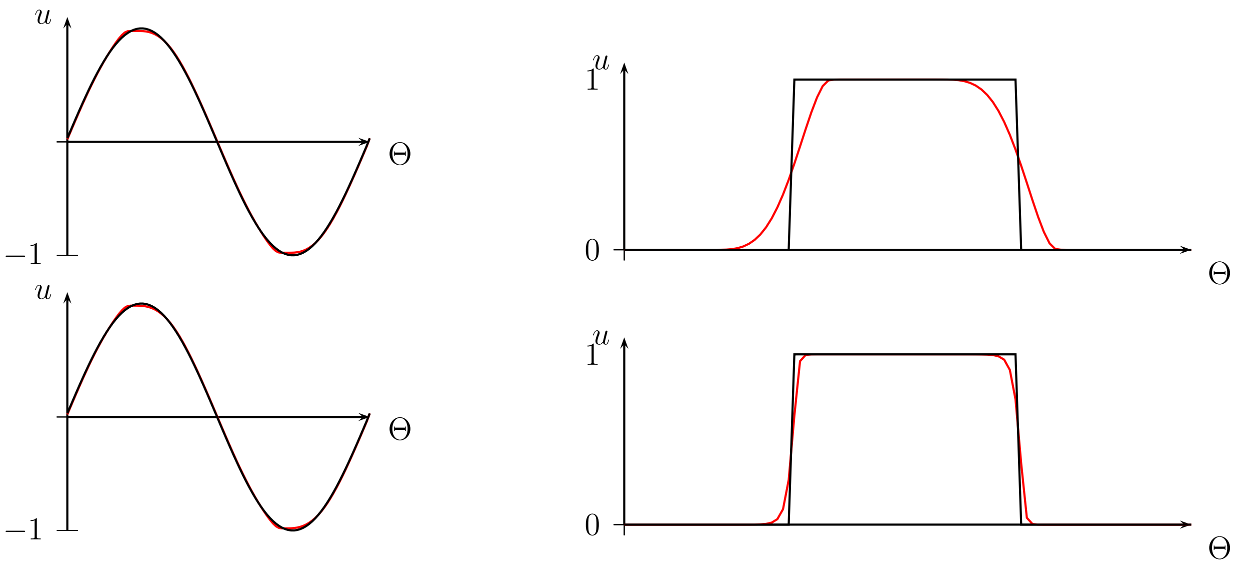


Figure 15: Original (top row) and modified MC scheme (bottom row) with $\Delta t = 0.0025$, $\Delta x = 0.01$, $t = 10$. **Left.** The sine wave test. **Right.** The box function test.

6.2 The Burgers equation

Burgers' equation is considered to be the most simple non-linear test case:

$$u_t + (\frac{1}{2}u^2)_x = 0 \tag{56}$$

The flux function is convex, and thus either (discontinuous) shocks or (continuous) rarefaction waves appear in the solution of Riemann problems, i.e. for a jump function with a single discontinuity as initial data.

Case 1: Shock Wave

As first initial data, we consider $u_l = 1$ and $u_r = 0$, i.e.,

$$u(x,0) = \begin{cases} 1 & x < 0 \\ 0 & x > 0 \end{cases} \tag{57}$$

The exact solution in this case is a a shock, i.e. the initial discontinuity is translated with velocity $s = 1/2$. We only consider the solution at $t = 1$ and $t = 2$, as this suffices to see what can be achieved by our improvement algorithm.

The most accurate improved *MC scheme* is obtained by applying the concentration operator with parameter value $n = 6$ to the linguistic term “extremum”, and the dilation modifier with parameter value $n = 8$ to the linguistic term “steep”, whereas the linguistic term “smooth” is not modified. The resulting control function is shown in Figure 16(a). The improved scheme is around 20 percent better than the original scheme, cf. Table 8.

The other schemes. The best improvement of the *Minmod scheme* is gained by modifying the linguistic terms for the input parameters as follows: The concentration modifier is applied to the linguistic term “extremum” with a parameter value of “n=2”, and the dilation modifier is applied to the linguistic term “smooth” with a parameter value of $n = 8$. Comparing the computed errors of the original with the improved scheme we see that the error is reduced by 36 percent, cf. Table 9.
Also the *Superbee scheme* can drastically be improved using our algorithm, namely around 20 percent for the test case, cf. Table 10. The improved scheme is obtained by the application of the concentration modifier with parameter value $n = 8$ to the linguistic term “extremum”, the application of the dilation operator with parameter value $n = 2$ to the linguistic term “smooth”, and the dilation operator with parameter value $n = 8$ to the linguistic term “steep”.

Case 2: Rarefaction Wave

As second test case, we consider the initial data

$$u(x,0) = \begin{cases} 0 & x < 0 \\ 1 & x > 0 \end{cases} \tag{58}$$

In this case, the correct solution is a rarefaction wave, where the density decreases continuously across the wave. It is given by the following equation:

$$u(x,t) = \begin{cases} u_l & x < u_l \cdot t \\ \dfrac{x}{t} & u_t \cdot t \leq x \leq u_r \cdot t \\ u_r & x > u_r \cdot t \end{cases} \tag{59}$$

The best improvement of the *MC scheme* is obtained when applying the concentration modifier with parameter value $n = 6$ to the linguistic term “smooth”, and applying the

Iterations	MC Error	MC mod. Error	Improvement
400	0.00313272	0.00248803	20.58%
800	0.00313222	0.00252466	19.40%

Table 8: **Shock wave test.** Original and modified MC scheme.

Iterations	Minmod Error	Minmod mod. Error	Improvement
400	0.00383789	0.00243818	36.47 %
800	0.00383739	0.00243768	36.48 %

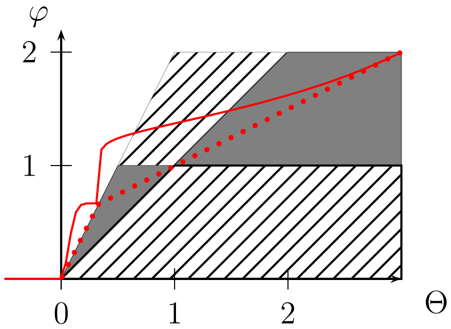
Table 9: **Shock wave test.** Original and modified Minmod scheme.

Iterations	Superbee Error	Superbee mod. Error	Improvement
400	0.00296601	0.00239133	19.38%
800	0.00296551	0.00239083	19.38%

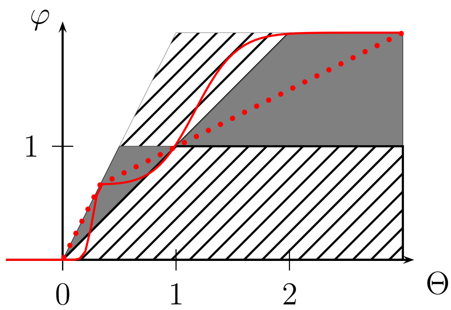
Table 10: **Shock wave test.** Original and modified Superbee scheme.

concentration operator with parameter value $n = 2$ to the linguistic term "steep". By doing so we can improve the accuracy of the numerical method by 35 percent, as shown in Table 11. The modified control functions for both tests concerned with Burgers' equation are shown in Figure 16. Note that even in the considered simple settings, the optimised control functions seem to be non-trivial, non-linear functions.

The other schemes. We first try to fine tune the *Minmod scheme*. However, for the first time our improvement algorithm fails, as the returned numerical scheme is less than a half percent better then the original numerical scheme. It is obtained by the application of the contrast operator with parameter value $n = 2$ to the linguistic term "extremum", and the application of the contrast operator with parameter value $n = 8$ to the linguistic term "smooth". An analysis of the data shows that this is because the smoothness measure Θ is higher than one in large parts of the rarefaction. The unsatisfactory behaviour of the tuning algorithm can be explained by the fact that the application of a modifier does not change the tolerance set of a fuzzy set. A closer look at the linguistic term "smooth" shows that a smoothness measure larger than 1 lies in the tolerance set of the fuzzy set. Thus the application of a modifier does not affect the treatment of the rarefaction wave.
The analysis of the input data in the previous case lets us hope that we can obtain an improved version of the *Superbee scheme*, as the rule base for the latter contains an extra rule covering a smoothness measure of larger than 1. Indeed, for this case we can improve the scheme around 50 percent, c.f. Table 12. The best accuracy is obtained by the application of the dilation operator with parameter value $n = 6$ to the linguistic term "extremum", the concentration operator with parameter value $n = 8$ to the linguistic term "smooth", and the concentration operator with parameter value $n = 2$ to the linguistic term "steep".



(a) Modified control function for shock wave test



(b) Modified control function for rarefaction wave test

Figure 16: Control functions of modified MC schemes for simulations of Burgers' equation.

6.3 The Buckley-Leverett Equation

This test is concerned with a non-linear HCL featuring the non-convex flux function

$$f(u) = \frac{u^2}{u^2 + a(1-u)^2} \tag{60}$$

where $a := 1/2$ is a constant parameter. Due to the non-convexity, the solution of the Riemann problem with initial data as in (6.2) is given by a mixed wave, i.e. a shock to which a rarefaction wave is attached, see e.g. [19].

The most accurate modified *MC scheme* is obtained by applying the dilation operator with parameter value $n = 6$ to the linguistic term "extremum", the contrast operator with parameter value $n = 2$ to the linguistic term "smooth", and the contrast operator with parameter value $n = 2$ to the linguistic term "steep". In this case, a significant gain in resolution quality is achieved when resolving the moving shock; this explains the considerably improvement observable in Table 13. The modified linguistic terms and the highly non-linear control function is displayed in Figure 17.

The other schemes. The fine tuning of the *Minmod scheme* results in a mediocre improvement using the dilation operator with parameter value $n = 2$ applied to the linguistic

Iterations	MC Error	MC mod. Error	Improvement
200	0.00106768	0.000685669	35.78%
400	0.00104649	0.000679524	35.07%

Table 11: **Rarefaction wave test.** Original and modified MC scheme.

Iterations	Superbee Error	Superbee mod. Error	Improvement
200	0.000553645	0.000282717	48.94%
400	0.000559025	0.000299164	46.48%

Table 12: **Rarefaction wave test.** Original and modified Superbee scheme.

term “extremum” and to the linguistic term “smooth”, cf. Table 14. However, a significant quality gain can not be achieved for this scheme here; compare our discussion of the rarefaction wave test in the previous paragraph.
The modification provided by our algorithm for the *Superbee scheme* is given by an application of the dilation operator with parameter value $n = 8$ to the linguistic term “extremum”, the application of concentration modifier with parameter value $n = 8$ to the linguistic term “smooth”, and the application of the dilation operator with parameter value $n = 8$ to the linguistic term “steep”. An evaluation of the scheme can be found in Table 15, revealing a considerable improvement of about fifty percent.

Iterations	MC Error	MC mod. Error	Improvement
200	0.00924061	0.00574125	37.87 %
400	0.00908056	0.00490976	45.93 %
600	0.00853300	0.00446720	47.65 %

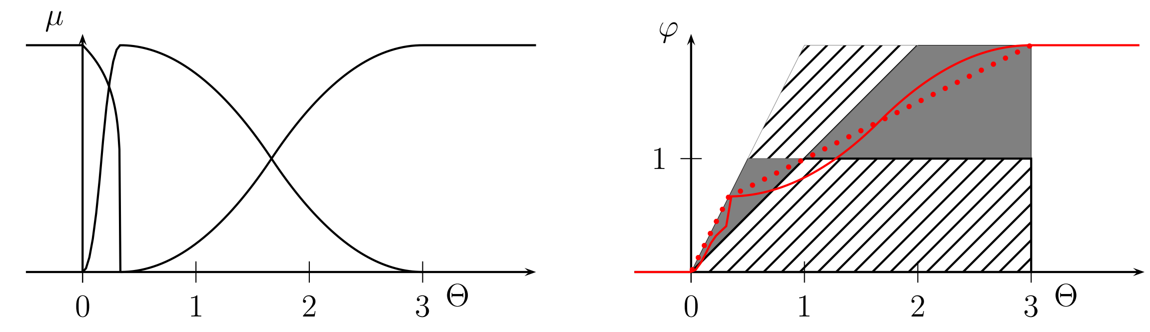
Table 13: **Buckley-Leverett test.** Original and modified MC scheme.

Iterations	Minmod Error	Minmod mod. Error	Improvement
200	0.00464404	0.00431400	7.11 %
400	0.00452810	0.00360098	20.47 %
600	0.00532273	0.00388623	26.99 %

Table 14: **Buckley-Leverett test.** Original and modified Minmod scheme.

Iterations	Superbee Error	Superbee mod. Error	Improvement
200	0.0139729	0.00822150	41.16%
400	0.0150731	0.00715809	52.51%
600	0.0155228	0.00654816	57.82%

Table 15: **Buckley-Leverett test.** Original and modified Superbee scheme.



(a) Modified linguistic terms for the input variable

(b) Modified control function

Figure 17: **Buckley-Leverett test.** Modified fuzzy sets and control function for the MC scheme.

7 Summary and Conclusion

In this paper we have proposed a novel idea for constructing flux limiter methods using fuzzy logic. The benefits of our framework are that flux limiters can be given an easy interpretation, and it is also easy to modify them. We have investigated how accurate novel FL-based schemes can be obtained by improving basic schemes using hedges.

Our work shows that the FL-based approach can be used with benefit for the construction of numerical schemes for PDEs. We are optimistic that this paper gives the foundation of a fruitful line of future works concerned with more intricate FL-based methods, using e.g. sophisticated optimisation procedures, other inference engines, or additional rules within the rule base.

References

[1] S. Benzoni-Gavage, D. Serre: Multi-dimensional hyperbolic partial differential equations – First-order systems and applications. Oxford Mathematical Monographs, (2006)

[2] Breuß, M., Dietrich, D.: Fuzzy Logic for Hyperbolic Numerics. In Mertsching, B.; Hund, M.; Aziz, Z. (Eds.): KI2009. Lecture Notes in Artificial Intelligence, Vol. 5803, Springer, Berlin, (2009)

[3] Chiang, C. K., Chung H. Y., Lin, J. J. A Self-Learning Fuzzy Logic Controller Using Genetic Algorithms with Reinforcements Transaction on Fuzzy Systems, Vol 5., (1997)

[4] Genetic algorithms for learning the rule base of fuzzy logic controller T. C. Chin and X. M. Qi Fuzzy Sets and Systems, 1–7, (1998)

[5] Clarke, F., Ekeland, I.: Nonlinear oscillations and boundary-value problems for Hamiltonian systems. Arch. Rat. Mech. Anal. 78, 315–333 (1982)

[6] Clarke, F., Ekeland, I.: Solutions périodiques, du période donnée, des équations hamiltoniennes. Note CRAS Paris 287, 1013–1015 (1978)

[7] M.G. Crandall, A. Majda: *Monotone difference approaximations for scalar conservation laws*, Math. Comp., **34**, 1-21, (1980)

[8] Evans, L. C.: Partial Differential Equations. Oxford University Press (1998)

[9] Giusti, E.: Minimal Surfaces and Functions of Bounded Variation. Monographs in Mathematics, vol. 80, Birkhäuser, (1984)

[10] E. Godlewski, P.-A. Raviart: Numerical Approximation of Hyperbolic Systems of Conservation Laws. Springer, New York, (1996)

[11] Goodridge, S. G., Kay M. G., Luo R. C.: Multi-layered fuzzy behavior fusion for reactive control of an autonomous mobile robot. Procs of the 6th IEEE Int Conf on Fuzzy Systems (Barcelona, SP) 579-584 (1996).

[12] Klir G. J., Yuan, B. Fuzzy Sets and Fuzzy Logic, Theory and Applications (1995)

[13] Harris, J.: Fuzzy Logic Applications in Engineering Science. Springer Verlag (2005)

[14] A. Harten: *High resolution schemes for hyperbolic conservation laws*, J. Comput. Phys., **49**, 357-393, (1983)

[15] A. Harten: *On a class of high resolution total variation stable finite difference schemes*, SIAM J. Numer. Anal., **21**, 1-23, (1984)

[16] Kosko, B.: Fuzzy Thinking: The New Science of Fuzzy Logic New York (1993)

[17] P.D. Lax, B. Wendroff: *Systems of conservation laws*, Comm. Pure Appl. Math., **13**, 217-237, (1960)

[18] Lee, C. C.: Fuzzy logic in control systems: Fuzzy logic controller, part I IEEE, Trans. Fuzzy Syst., vol 2, pp 4–15, (1994)

[19] R. J. LeVeque: Numerical Methods for Conservation Laws. Birkhäuser (1990)

[20] R. J. LeVeque: Finite Volume Methods for Hyperbolic Problems. Cambridge University Press (2002)

[21] Lim, M. H., Rahardja S., Gwee, B. H. A GA paradigm for learning fuzzy rules School of EEE, Nanyang Technological University (1996)

[22] A. Meister, J. Struckmeier (Eds.): Hyperbolic Partial Differential Equations: Theory, Numerics and Applications. Vieweg Verlag (2002)

[23] Michalek, R., Tarantello, G.: Subharmonic solutions with prescribed minimal period for nonautonomous Hamiltonian systems. J. Diff. Eq. 72, 28–55 (1988)

[24] Rabinowitz, P.: On subharmonic solutions of a Hamiltonian system. Comm. Pure Appl. Math. 33, 609–633 (1980)

[25] D. Serre: Systems of conservation laws I. Cambridge University Press (1999)

[26] D. Serre: Systems of conservation laws II. Cambridge University Press (2000)

[27] P. Sweby: High-resolution schemes using flux limiters for hyperbolic conservation laws. SIAM J. Numer. Anal. 21, 995–1011 (1984)

[28] Tarantello, G.: Subharmonic solutions for Hamiltonian systems via a ZZ_p pseudoindex theory. Annali di Matematica Pura (to appear)

[29] Terano, T., Asai, K., Sugeno, M.: Fuzzy Systems Theory and Its Applications New York, Academic (1992)

[30] E.F. Toro: Riemann Solvers and Numerical Methods for Fluid Dynamics. Springer-Verlag. Second Edition (1999)

[31] Y. Tsukamoto: An approach to fuzzy reasoning method. Advances in Fuzzy Set Theory and Applications North Holland, Amsterdam, 137–149 (1979)

[32] Mamdani, E.H. and S. Assilian An experiment in linguistic synthesis with a fuzzy logic controller International Journal of Man-Machine Studies, pp 1–13 (1975)

[33] Takagi, T. and Sugeno, M Fuzzy Identification of Systems and its Applications to Modeling and Control IEEE Transactions on Systems Man and Cybernetics, pp 116–132 (1985)

[34] Von Altrock, C.: Recent Successful Fuzzy Logic Applications in Industrial Automation IEEE, Fifth International Conference on Fuzzy Systems (1996)

[35] Warnecke, G. (Ed.): Analysis and Numerics for Conservation Laws. Springer Verlag (2005)

[36] Xu, D., Keller, J. M., Popescu, M. Applications of Fuzzy Logic in Bioinformatics: (Advances in Bioinformatics and Computational Biology) (2008)

[37] Zadeh, L. A.: Fuzzy Sets, Information and Control (1965)