Nonlinear Limiters for Discontinuous Galerkin Method

by

Bingyu Zhao

B.Sc., Nanjing University; Nanjing, China, 2012

M.Sc., Brown University; Providence, RI, 2013

A dissertation submitted in partial fulfillment of the
requirements for the degree of Doctor of Philosophy
in the Division of Applied Mathematics at Brown University

PROVIDENCE, RHODE ISLAND

May 2018

This dissertation by Bingyu Zhao is accepted in its present form
by the Division of Applied Mathematics as satisfying the
dissertation requirement for the degree of Doctor of Philosophy.

Date_____ _____

Chi-Wang Shu, Ph.D., Advisor

Recommended to the Graduate Council

Date_____ _____

Jun Zhu, Ph.D., Reader

Date_____ _____

Guosheng Fu, Ph.D., Reader

Approved by the Graduate Council

Date_____ _____

Andrew G. Campbell, Dean of the Graduate School

Vita

Education

Ph.D., Applied Mathematics (defensed in May 2017)
Division of Applied Mathematics, Brown University, Providence, RI
Adviser: Prof. Chi-Wang Shu

M.Sc., Applied Mathematics (May 2013)
Division of Applied Mathematics, Brown University, Providence, RI

Cross-disciplinary Scholars in Science and Technology Program (2011, 2012)
Department of Biomathematics, University of California Los Angeles, CA

B.Sc., Information and Computing Science (June 2012)
Department of Mathematics, Nanjing University, Nanjing, P.R.China

Research Experiences

Division of Applied Mathematics, Brown University, Providence, RI, USA, Sept 2013 - now
Dissertation: Nonlinear limiters for discontinuous Galerkin method

Institute for Mathematics and its Applications, University of Minnesota, Minneapolis, MN, USA, Aug 2014
WhAM Workshop on Numerical Partial Differential Equations and Scientific Computing
Research Project: Principal eigenvalue for a population dynamics model problem with both local and nonlocal dispersal

Computer Science and Mathematics Division, Oak Ridge National Laboratory, Oak Ridge, TN, USA, Jun - Aug 2014

Conducted research on maximum principle preserving limiters for implicit Discontinuous Galerkin schemes

Department of Biomathematics, University of California Los Angeles, Los Angeles, CA, USA, Feb - Jun 2012
Research Project: Combinatoric analysis of heterogeneous stochastic self-assembly

Teaching Experiences

Teaching Assistant Sept 2013 - Sept 2014
Division of Applied Mathematics, Brown University, Providence, RI

- *APMA 0350: Methods of Applied Mathematics III*, Teaching Assistant and Recitation Instructor, Spring 2014
- *APMA 2550: Numerical Solutions of Partial Differential Equations I*, Graduate Teaching Assistant, Fall 2013

Coach for Computational Mathematics Jul - Aug 2016
The 2016 Tsinghua Mathcamp, Tsinghua University, Beijing, China

Honors

- Graduate Student Fellowship, Brown University (2012)
- UCLA CSST Scholarship (2011, 2012)
- National Scholarship, the Ministry of Education of the People's Republic of China (2009)
- Outstanding Student Scholarship, Nanjing University (2010, 2011)

Publications

- M. D'Orsogna, B. Zhao and T. Chou, *Combinatoric analysis of heterogeneous stochastic self-assembly*, The Journal of Chemical Physics, v139 121918 (2013).

- M. Chugunova, B. Jadamba, C.-Y. Kao, C. Klymko, E. Thomas and B. Zhao, *Study of a Mixed Dispersal Population Dynamics Model*, Topics in Numerical Partial Differential Equations and Scientific Computing, Springer New York, 51-77, 2016.

Acknowledgments

First and foremost, my deepest gratitude is to Professor Chi-Wang Shu, my thesis advisor. This five years has been definitely the most unusual period of time in my life. When looking back, with each step, I have always been inspired by his profoundly deep insight in the scientific computing area, as well as by his noble academic personality. The high standards which he holds towards research have remarkably influenced me from my very first day at Brown, and will continue to do so in my future life ahead. Professor Shu worked tirelessly for my intellectual and professional growth, from fielding both my excitement and frustration along every step of my research, to carefully reviewing each word of my thesis as many times as necessary, I would like to thank Professor Shu for his infinite patience and excellent guidance throughout my graduate life.

I would also like to thank Professors Jun Zhu and Guosheng Fu for graciously agreeing to serve on my dissertation committee, generously spending time reading my thesis, and for providing their helpful advice.

Over the years, I have had the honer and pleasure to work on various interesting projects with many extraordinary scientists, Prof. Yulong Xing at University of California Riverside, Dr. Cory Hauck at Oak Ridge National Laboratory, Prof. Chiu-Yen Kao at Claremont McKenna College, Prof. Tom Chou at University of California Los Angeles. I am grateful for their help, insight and guidance in these invaluable experiences.

At Brown University, I am very lucky to have grown up academically within the graduate student community in the Division of Applied Mathematics. Surrounded by such exceptional peers, I find myself greatly benefited from the wonderful graduate environment. I would particularly like to thank Tong Qin, Cengke Shi, Zheng Sun and Jie Du, for their countless help, for always being willing to discuss, and for supporting me throughout the ups and downs.

I owe my loving thanks to my parents, it's an amazing experience throughout the journey to constantly realize how thankful I am for their positive impact on me, in many aspects of my life. They selflessly love and support me no matter what, and allow me to pursue my Ph.D. career overseas without having to worry about a thing in the world.

Finally to my dearest Mister SU, thank you for your magnificent love, for hand holding me during all those tough moments, for always being there full heartedly in my life.

Contents

List of Tables

List of Figures

Chapter One

Introduction

Discontinuous Galerkin (DG) methods are a class of finite-element methods using completely discontinuous basis functions, which are usually chosen as piecewise polynomials. In this dissertation, we consider using the DG methods to solve the following hyperbolic conservation law

$$u_t + f(u)_x = 0, \qquad u(x,0) = u_0(x) \tag{1.1}$$

and its two-dimensional version.

For the one dimensional hyperbolic conservation law, for simplicity we consider a uniform partition of the computational domain (a,b) in N cells, with cell boundaries $\{\, x_{j+1/2} \,\}_{j=0}^{N}$. We set $I_j = (x_{j-1/2}, x_{j+1/2})$ and the cell center $x_j = \frac{1}{2}(x_{j-1/2} + x_{j+1/2})$, for $j = 1, \ldots, N$, where

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

and denote the mesh size $\Delta x = (b-a)/N$.

With an abuse of notation, we seek a numerical approximation u such that for each time $t \in [0,T]$, $u(t)$ belongs to the finite dimensional space

$$V_h = V_h^k \equiv \{v : v|_{I_j} \in P^k(I_j),\ j = 1, \ldots, N\}, \tag{1.2}$$

where $P^k(I)$ denotes the space of polynomials in I of degree up to k.

The semi-discrete DG method for solving (1.1) is defined as follows: find the unique function $u = u(t) \in V_h$, such that, for $j = 1, 2, \ldots, N$,

$$\int_{I_j} u_t\, v\, dx - \int_{I_j} f(u)\, v_x\, dx + \hat{f}_{j+\frac{1}{2}}\, v(x_{j+\frac{1}{2}}^-) - \hat{f}_{j-\frac{1}{2}}\, v(x_{j-\frac{1}{2}}^+) = 0 \tag{1.3}$$

holds for all $v \in V_h$. Here $\hat{f}_{j+\frac{1}{2}}$ is a numerical flux, originally designed for finite difference and finite volume schemes for solving conservation laws [26].

The above defined DG method was first introduced in 1973 by Reed and Hill [38] solving the equations for neutron transport, which are time independent linear hyperbolic equations. A major development of the DG method is carried out by Cockburn et al. in a series of papers [9][11][12][13][14][15], where a framework has been established to easily solve nonlinear time-dependent hyperbolic equations, such as the Euler equations of compressible gas dynamics.

Since the basis functions of the DG methods can be completely discontinuous, DG has the flexibility which is not shared by typical finite-element methods, such as the allowance of arbitrary triangulation with hanging nodes, complete freedom in changing the polynomial degrees in each element independent of that in the neighbors (p adaptivity), and extremely local data structure and the resulting high parallel efficiency [10]. However, DG methods are oscillatory in the presence of strong discontinuities. One of the bottlenecks for the development of the DG methods is the design of good, robust limiters to control these oscillations. Efforts have been made in this dissertation to investigate two types of limiters for the DG methods.

An important property of the unique entropy solution to the initial value problem of a scalar conservation law (1.1) is that it satisfies a strict maximum principle, namely, if $u_M = \max_x u_0(x)$, $u_m = \min_x u_0(x)$, then $u(x,t) \in [u_m, u_M]$ for any x and t. It is a highly desirable property, not only because violating it may produce physically meaningless numerical solutions, but could also lead to numerical instability or ill-posedness. However, the existing high order schemes for solving (1.1) do not necessarily retain the maximum principle preserving property. Two kinds of high order discrete maximum principle preserving (MPP) limiters have been developed in

recent years. One is the polynomial re-scaling MPP limiter proposed by Zhang and Shu in [54][55][57][56][58][53]. The other is the high order parametrized MPP flux limiter proposed by Xu et al. in[50][28], which was later improved by Xiong et al. [47] by applying the limiter only at the final stage of the Runge-Kutta method. In this dissertation, we apply the parametrized MPP flux limiter on implicit discontinuous Galerkin methods solving one dimensional transport equation using the upwinding flux. We also explore another limiting approach by solving a global optimization problem with coupled inequalities ensuring the maximum principle.

One of the main difficulties in solving (1.1) is that the solution may contain discontinuities even if the initial condition is smooth. While DG methods have many advantages, when computing solutions with strong discontinuities, the scheme is not enough to control spurious numerical oscillations by itself, and sometimes may even result in nonlinear instabilities. Therefore nonlinear limiters prove to be an important component of to control these oscillations, especially for nonlinear problems containing strong shocks. There are many limiters for DG existing in the literature [19] [41] [4] [37] [60] [59], most of which act independently as an add-on procedure upon the original DG scheme. In this dissertation, we discuss a new limiting approach by changing the integral formulation of the DG scheme and incorporating the limiting procedure within the original DG framework, similar to finite element methods with artificial diffusion[31][21]. To do so we add a shock capturing term in the variational formulation of the DG scheme based on the WENO reconstructed polynomial in [59]. We expect the extra term to act as an artificial viscosity to control spurious numerical oscillations near discontinuities while maintaining the original order of accuracy, and we have the freedom of adjusting the added term by a parameter ε.

This dissertation is organized as follows. In Chapter 2, we investigate the

parametrized MPP flux limiters for the one dimensional transport equation. In Chapter 3, we present the artificial diffusion WENO limiter for the DG methods. Conclusion and remarks for future work will be given at the end of each chapter.

Chapter Two

Parametrized MPP Flux Limiters for the One Dimensional Transport Equation

2.1 Introduction

In this chapter, we investigate a parametrized maximum principle preserving (MPP) flux limiter, and a global flux limiter, for the high order implicit discontinuous Galerkin finite element method, solving the one dimensional transport equation

$$u_t + u_x = 0 \tag{2.1}$$

subject to the initial condition $u(x,0) = u_0(x)$, and the inflow boundary condition $u(0,t) = g(t)$.

For this equation, the solution satisfies a strict maximum principle, that is

$$u_m \leq u(x,t) \leq u_M, \tag{2.2}$$

if $u_m \leq u_0(x) \leq u_M$ and $u_m \leq g(t) \leq u_M$.

When $u_m = 0$, the problem is generally addressed as positivity preserving.

However, the existing high order schemes for solving hyperbolic conservation laws do not necessarily retain the maximum principle preserving property. Zhang and Shu [54] previously developed a class of maximum principle preserving limiters for arbitrarily high order finite volume WENO and discontinuous Galerkin methods, coupled with explicit strong stability preserving (SSP) time discretization [18] by limiting the reconstructed polynomials around cell averages. They demonstrated that the schemes were able to maintain the designed order of accuracy under a CFL constraint.

Recently, Xu et al. in [50][28][7][24][48][49][51] developed a new class of parametrized maximum principle preserving flux limiters for hyperbolic conservation laws. The key idea of this limiter is to limit the temporal integrated high order numerical flux with a first order monotone flux, which preserves the maximum principle. The main advantage of this new limiter is that the designed order of accuracy of the orginal scheme might be able to be maintained without excessively restricting the CFL number. Later they improved the parametrized MPP flux limiter by applying it only at the final Runge-Kutta (RK) stage to reduce the computational cost and to relax the CFL constraints.

In this chapter, we propose to apply the parametrized MPP flux limiter on implicit discontinuous Galerkin method to relax the strict CFL constraints of the explicit schemes. The feasibility of applying the MPP flux limiters to the DG solution lies on the fact that the cell averages for the DG scheme are updated in a conservative fashion by using flux differences. The parametrized MPP flux limiter is proposed ro preserve the MPP property for the cell averages and at the final Runge-Kutta stage only. Our approach can be viewed as a low-cost easy to implement post processing procedure that modifies the high order numerical fluxes towards a first order one, for the evolution of the cell averages, with all higher order moments unchanged. Based on the same framework in search for the limiting parameters $\theta_{j+1/2}$, we also explore another global flux limiting approach by solving a global optimization problem using the Simplex algorithm to ensure maximum principle on the cell averages. Experiments for the one dimensional transport equation with different initial conditions will show the performance of these two kinds of limiters for implicit DG schemes.

The outline of the chapter is as follows. In Section 2.2, the DG formulation with implicit Runge-Kutta time integration is described. The formulation of the parametrized MPP flux limiter is presented in Section 2.3, and also the global flux

limiter from the Simplex Algorithm in Section 2.4. Numerical results are provided in Section 2.5. Finally we give the concluding remarks in Section 2.6.

2.2 Implicit Discontinuous Galerkin Formulation

In this section, we give an overview of the algorithm formulation of the implicit discontinuous Galerkin method solving the transport equation (2.1).

The semi-discrete DG scheme has already been given in (1.3), we take the upwind numerical flux therein, that is

$$\hat{f} = f(u^-) = u^- \tag{2.3}$$

at all interfaces.

Here and below u^-, u^+ denote the left and right limits of the function u at the cell interface, respectively.

Now we consider the implementation of the DG scheme. We note that for every new time step, when we solve for the numerical solution for the current cell, we only need the information of the left adjacent cell (or the left boundary), and the numerical solution of the previous step. Therefore we can sweep the cells from left to right to get the numerical solution at the new time level in the computation.

After discretizing in space the problem by the DG method, we obtain a system of ODEs for the degrees of freedom that we can rewrite as follows:

$$\frac{d}{dt}\,u = L(u), \qquad \text{in } (0,T), \tag{2.4}$$

where $L(u)$ is the spatial discretization operator.

To discretize our ODE system in time, we use the implicit Runge-Kutta time discretization.

The general description of an m-stage Runge-Kutta method for solving the Cauchy problem

$$y' = f(t,y)$$
$$y(t_0) = y_0$$

is given by the formulas

$$\begin{aligned} y^{(i)} &= y_n + \Delta t \sum_{j=1}^{m} a_{ij} f(t_n + c_j \Delta t, y^{(j)}) \qquad (1 \le i \le m) \\ y_{n+1} &= y_n + \Delta t \sum_{i=1}^{m} b_i f(t_n + c_i \Delta t, y^{(i)}) \end{aligned}$$

which can be represented in a Butcher table as follows

$$\begin{array}{c|ccc} c_1 & a_{11} & \dots & a_{1m} \\ \vdots & \vdots & \dots & \vdots \\ c_m & a_{m1} & \dots & a_{mm} \\ \hline & b_1 & \dots & b_m \end{array}$$

The notation $\mathbf{A} = (a_{ij})$, $\mathbf{b} = (b_j)$ and $\mathbf{c} = (c_j)$ allows any Runge-Kutta method given in a Butcher table, in which the coefficients of the method satisfy the following condition

$$\mathbf{c} = \mathbf{A}\mathbf{e} \qquad \mathbf{b}^T\mathbf{e} = 1 \qquad \mathbf{e} = [1, \dots, 1]^T \tag{2.5}$$

Among the implicit Runge-Kutta methods, the diagonally implicit Runge-Kutta (DIRK) methods [33], for which A is lower triangular, can be implemented efficiently without transforming to the Jordan form of $\mathbf{A}$. For simplicity we use DIRK methods in this chapter for time integration. There are two popular types of such methods, namely, SDIRK (Singly DIRK) and ESDIRK (Explicit first stage Singly DIRK) methods. The SDIRK methods are both singly implicit and diagonally implicit, where $\mathbf{A}$ is lower triangular with all diagonal entries identical, whereas the ESDIRK methods have an explicit first stage. For details on efficient implementation of implicit Runge-Kutta methods, refer to [2][6][1][44][8][29].

We specifically choose two 3-stage third order L-stable methods in our numerical tests, SDIRK33 and ESDIRK33 (the first digit indicates the number of implicit stages, and the second digit indicates the order).

SDIRK33 method has the following Butcher table

α	α	0	0
τ_2	$\tau_2 - \alpha$	α	0
1	b_1	b_2	α
	b_1	b_2	α

where α is the root of $x^3 - 3x^2 + \frac{3}{2}x - \frac{1}{6} = 0$ lying in $(\frac{1}{6}, \frac{1}{2})$, $\tau_2 = \frac{1+\alpha}{2}$, $b_1 = -(6\alpha^2 - 16\alpha + 1)/4$ and $b_2 = (6\alpha^2 - 20\alpha + 5)/4$.

And the Butcher table for ESDIRK33 method reads

0	0			
2γ	γ	γ		
c_3	$c_3 - a_{32} - \gamma$	a_{32}	γ	
1	$1 - b_2 - b_3 - \gamma$	b_2	b_3	γ
1	$1 - b_2 - b_3 - \gamma$	b_2	b_3	γ

where γ is the root of $x^3 - 3x^2 + \frac{3}{2}x - \frac{1}{6} = 0$ lying in $(\frac{1}{6}, \frac{1}{2})$, $c_3 = \frac{1}{2} + \frac{\gamma}{4}$, $a_{32} = \frac{c_3(c_3 - 2\gamma)}{4\gamma}$, $b_2 = \frac{2 - 6\gamma - 3c_3 + 6\gamma c_3}{12\gamma(2\gamma - c_3)}$ and $b_3 = \frac{1 - 6\gamma + 6\gamma^2}{3c_3(c_3 - 2\gamma)}$.

Thus, for the final computing time T, if $\{t^n\}_{n=0}^N$ is a partition of $[0, T]$, and $\Delta t^n = t^{n+1} - t^n, n = 0, ..., N-1$, our time-marching algorithm reads as follows:

- Set u^0;
- For $n = 0, ..., N-1$ compute u^{n+1} from u^n as follows:
    1. for $i = 1, ..., m$ compute the intermediate solutions from:

$$u^{(i)} = u^n + \Delta t \sum_{j=1}^{i} a_{ij} L(u^{(j)});$$

2. set

$$u^{n+1} = u^n + \Delta t \sum_{i=1}^{m} b_i L(u^{(i)}).$$

Note since for both methods, the last row in $\mathbf{A}$ is identical to $\mathbf{b}^T$(the stiffly accurate methods), thus we actually have $u^{n+1} = u^{(m)}$ in the computation.

2.3 Parametrized Maximum Principle Preserving Flux Limiter Formulation

Upon the previously discussed implicit DG scheme, we apply the parametrized maximum principle preserving (MPP) flux limiter on the cell average of the numerical solutions at the final stage of every Runge-Kutta time level. In this section, we present the formulation of the parametrized MPP flux limiter. Take $v = 1$ in the semi-discrete DG scheme (1.3), we get the time evolution of the cell average

$$\frac{d}{dt}\bar{u} = -\frac{1}{\Delta x}(\hat{f}_{j+1/2} - \hat{f}_{j-1/2}) \tag{2.6}$$

Using DIRK in time, the updating of cell average in each Runge-Kutta step can be written as

$$\bar{u}_j^{n+1} = \bar{u}_j^n - \lambda(\hat{f}^{rk}_{j+1/2} - \hat{f}^{rk}_{j-1/2}) \tag{2.7}$$

where $\lambda = \frac{\Delta t}{\Delta x}$, and $\hat{f}^{rk}$ is the combined numerical flux for the m-stage implicit Runge-Kutta method at the current time level, which can be computed explicitly.

Recall the 3-stage third order DIRK algorithm as discussed in the previous session,

$$\begin{aligned} u^{n+1} &= u^n + \Delta t \sum_{i=1}^{3} b_i k_i \\ k_1 &= L(u^n + \Delta t a_{11} k_1) \\ k_2 &= L(u^n + \Delta t(a_{21} k_1 + a_{22} k_2)) \\ k_3 &= L(u^n + \Delta t(a_{31} k_1 + a_{32} k_2 + a_{33} k_3)) \end{aligned} \tag{2.8}$$

where k_i $(i = 1, 2, 3)$ can be obtained through each Runge-Kutta stage and the intermediate numerical solutions. They can be written in the form,

$$\bar{k}_i = -\frac{1}{\Delta x}(\hat{f}^{(i)}_{j+1/2} - \hat{f}^{(i)}_{j-1/2}) \qquad i = 1, 2, 3 \tag{2.9}$$

where $\bar{k}_i$ is the cell average of k_i, with $\hat{f}^{(i)}(i = 1, 2, 3)$ being the numerical fluxes based on the corresponding numerical solutions at the i-th Runge-Kutta stage.

Thus from (2.8) we can get the expression for the total high order numerical flux $\hat{f}^{rk}$ from the implicit Runge-Kutta time discretization

$$\hat{f}^{rk}_{j+1/2} = b_1 \hat{f}^{(1)}_{j+1/2} + b_2 \hat{f}^{(2)}_{j+1/2} + b_3 \hat{f}^{(3)}_{j+1/2} \tag{2.10}$$

which does not necessarily preserves the maximum principle. At every Runge-Kutta stage, we store $\hat{f}^{(i)}_{j+1/2}$ in order to get $\hat{f}^{rk}_{j+1/2}$ at the final stage.

On the other hand, we denote the numerical flux derived from the first order

implicit method as $\hat{h}_{j+1/2}$, which preserves the maximum principle.

Now we aim to replace the numerical flux $\hat{f}^{rk}_{j+1/2}$ by the modified one

$$\tilde{f}^{rk}_{j+1/2} = \theta_{j+1/2}(\hat{f}^{rk}_{j+1/2} - \hat{h}_{j+1/2}) + \hat{h}_{j+1/2} \tag{2.11}$$

with $\theta_{j+1/2} \in [0, 1]$ for the cell averages. That is to say, if we use Legendre polynomials, for instance, for the basis of the DG scheme, we modify the numerical flux only for the first degree of freedom of the numerical solution. Let

$$u_m = \min_{x,t}(u(x, 0), g(t)), \qquad u_M = \max_{x,t}(u(x, 0), g(t)),$$

to ensure the maximum principle on cell averages

$$u_m \leq \bar{u}^n_j \leq u_M, \qquad \forall j, n$$

we seek parameters $\{\theta_{j+1/2}\}$ at time t_n such that the newly constructed numerical fluxes can ensure the maximum principle on cell averages $(\bar{u})^{n+1}_j \in [u_m, u_M]$. i.e.,

$$u_m \leq \bar{u}^{n+1}_j = \bar{u}^n_j - \lambda(\tilde{f}^{rk}_{j+1/2} - \tilde{f}^{rk}_{j-1/2}) \leq u_M \tag{2.12}$$

We decouple the inequalities (2.12) and compute $\{\theta_{j+1/2}\}$ in the following fashion.

For each $\theta_{j+1/2}$ limiting the numerical flux $\hat{f}^{rk}_{j+1/2}$, we are looking for proper upper bounds $\Lambda_{+1/2,I_j}$ and $\Lambda_{-1/2,I_{j+1}}$ from the needs of keeping $\bar{u}^{n+1}_j$ and $\bar{u}^{n+1}_{j+1}$ within $[u_m, u_M]$, respectively. In cell I_j, in order to obtain an independent pair of $(\theta_{j-1/2}, \theta_{j+1/2})$ for the numerical flux to be locally defined, we ought to find upper bounds $\Lambda_{-1/2,I_j}$ and $\Lambda_{+1/2,I_j}$, such that any chosen pair $(\theta_{j-1/2}, \theta_{j+1/2}) \in [0, \Lambda_{-1/2,I_j}] \times [0, \Lambda_{+1/2,I_j}]$ should satisfy (2.12).

We separate (2.12) into the following two inequalities. For the maximum value part,

$$\lambda\theta_{j-1/2}(\hat{f}^{rk}_{j-1/2} - \hat{h}_{j-1/2}) - \lambda\theta_{j+1/2}(\hat{f}^{rk}_{j+1/2} - \hat{h}_{j+1/2}) - \Gamma^M_j \leq 0, \tag{2.13}$$

where

$$\Gamma^M_j = u_M - (\bar{u}_h)^n_j + \lambda(\hat{h}_{j+1/2} - \hat{h}_{j-1/2}) \geq 0.$$

For the minimum value part,

$$\lambda\theta_{j-1/2}(\hat{f}^{rk}_{j-1/2} - \hat{h}_{j-1/2}) - \lambda\theta_{j+1/2}(\hat{f}^{rk}_{j+1/2} - \hat{h}_{j+1/2}) - \Gamma^m_j \geq 0, \tag{2.14}$$

where

$$\Gamma^m_j = u_m - (\bar{u}_h)^n_j + \lambda(\hat{h}_{j+1/2} - \hat{h}_{j-1/2}) \leq 0.$$

We denote $F_{j\pm1/2} = \hat{f}^{rk}_{j\pm1/2} - \hat{h}_{j\pm1/2}$,

First we determine $\Lambda^M_{\pm1/2,I_j}$ based on the signs of $F_{j-1/2}$ and $F_{j+1/2}$, to ensure the upper bound part.

(a) If $F_{j-1/2} \leq 0$ and $F_{j+1/2} \geq 0$, then the inequality (2.13) holds automatically, thus

$$\Lambda^M_{\pm1/2,I_j} = 1$$

.

(b) If $F_{j-1/2} \leq 0$ and $F_{j+1/2} < 0$,

$$(\Lambda^M_{-1/2,I_j}, \Lambda^M_{+1/2,I_j}) = (1, \min(1, \frac{\Gamma^M_j}{-\lambda F_{j+1/2}})).$$

(c) If $F_{j-1/2} > 0$ and $F_{j+1/2} \geq 0$,

$$(\Lambda^M_{-1/2,I_j}, \Lambda^M_{+1/2,I_j}) = (\min(1, \frac{\Gamma^M_j}{\lambda F_{j-1/2}}), 1).$$

(d) If $F_{j-1/2} > 0$ and $F_{j+1/2} < 0$,

If (2.13) is satisfied with $(\theta_{j-1/2}, \theta_{j+1/2}) = (1, 1)$, then

$$(\Lambda^M_{-1/2,I_j}, \Lambda^M_{+1/2,I_j}) = (1, 1).$$

If $(\theta_{j-1/2}, \theta_{j+1/2}) = (1, 1)$ does not satisfy (2.13), then

$$(\Lambda^M_{-1/2,I_j}, \Lambda^M_{+1/2,I_j}) = (\frac{\Gamma^M_j}{\lambda F_{j-1/2} - \lambda F_{j+1/2}}, \frac{\Gamma^M_j}{\lambda F_{j-1/2} - \lambda F_{j+1/2}}).$$

Similarly, $\Lambda^m_{\pm 1/2,I_j}$ can be obtained to preserve the lower bound by (2.14).

(a) If $F_{j-1/2} \geq 0$ and $F_{j+1/2} \leq 0$,

$$\Lambda^m_{\pm 1/2,I_j} = 1$$

.

(b) If $F_{j-1/2} \geq 0$ and $F_{j+1/2} > 0$,

$$(\Lambda^m_{-1/2,I_j}, \Lambda^m_{+1/2,I_j}) = (1, \min(1, \frac{\Gamma^m_j}{-\lambda F_{j+1/2}})).$$

(c) If $F_{j-1/2} < 0$ and $F_{j+1/2} \leq 0$,

$$(\Lambda^m_{-1/2,I_j}, \Lambda^m_{+1/2,I_j}) = (\min(1, \frac{\Gamma^m_j}{\lambda F_{j-1/2}}), 1).$$

(d) If $F_{j-1/2} < 0$ and $F_{j+1/2} > 0$,

If (2.14) is satisfied with $(\theta_{j-1/2}, \theta_{j+1/2}) = (1, 1)$, then

$$(\Lambda^m_{-1/2,I_j}, \Lambda^m_{+1/2,I_j}) = (1, 1).$$

If $(\theta_{j-1/2}, \theta_{j+1/2}) = (1, 1)$ does not satisfy (2.14), then

$$(\Lambda^m_{-1/2,I_j}, \Lambda^m_{+1/2,I_j}) = (\frac{\Gamma^m_j}{\lambda F_{j-1/2} - \lambda F_{j+1/2}}, \frac{\Gamma^m_j}{\lambda F_{j-1/2} - \lambda F_{j+1/2}}).$$

Put the constraints for upper bound and lower bound together, we get

$$\Lambda_{+1/2,I_j} = \min(\Lambda^M_{+1/2,I_j}, \Lambda^m_{+1/2,I_j}) \tag{2.15}$$

$$\Lambda_{-1/2,I_j} = \min(\Lambda^M_{-1/2,I_j}, \Lambda^m_{-1/2,I_j}) \tag{2.16}$$

Since the range of $\theta_{j+1/2}$ should ensure the inequality (2.12) to hold for both the left and the right cells, I_j and I_{j+1}, it is taken as

$$\theta_{j+1/2} = \min(\Lambda_{+1/2,I_j}, \Lambda_{-1/2,I_{j+1}}) \tag{2.17}$$

With the parameters $\{\theta_{j+1/2}\}$ defined above, the modified numerical flux can be computed as in (2.11) to update the cell averages.

$$\bar{u}^{n+1}_j = \bar{u}^n_j - \lambda(\tilde{f}^{rk}_{j+1/2} - \tilde{f}^{rk}_{j-1/2}) \tag{2.18}$$

We keep all other degrees of freedom unchanged in our scheme. The design of this parametrized flux limiter automatically guarantees the maximum principle for the cell averages of the numerical solution.

2.4 Global Flux Limiter from the Simplex Algorithm

The design of the parameters $\{\theta_{j+1/2}\}$ in the previous section is a sufficient condition for the cell averages to preserve the maximum principle. It is pretty restrictive since we require for $\forall\theta_{j+1/2} \in [0, \min(\Lambda_{+1/2,I_j}, \Lambda_{-1/2,I_{j+1}})]$, the coupled inequalities (2.12) should hold for $j = 1, 2, \ldots, N$. For instance, when we consider the upper bound requirement for the cell I_j, the value for the parameter $\theta_{j-1/2}$ should work for all possible values of $\theta_{j+1/2} \in [0, \Lambda^M_{+1/2,I_j}]$, not just any pair $(\theta_{j-1/2}, \theta_{j+1/2})$ satisfying (2.13). To relax the restriction for $\theta_{j+1/2}$, instead of seeking its upper bounds and ensuring the coupled inequalities to be held for all positive $\theta_{j+1/2}$ smaller than that, we consider the system of inequalities

$$u_m \leq \bar{u}^n_j - \lambda(\tilde{f}^{rk}_{j+1/2} - \tilde{f}^{rk}_{j-1/2}) \leq u_M, \qquad j = 1, 2, \ldots, N$$

in a global manner and intend to find single values of $(\theta_{j+1/2})_{j=0}^N$ that satisfy the coupled inequalities, with parameters $\theta_{j+1/2} \in [0, 1]$ as large as possible.

We maximize the sum of all $\theta_{j+1/2}$ and propose the following optimization problem

$$\begin{aligned}
\max \quad & \sum_{j=0}^{N} \theta_{j+1/2}, \\
s.t., \quad & \lambda F_{j-1/2}\theta_{j-1/2} - \lambda F_{j+1/2}\theta_{j+1/2} \leq \Gamma^M_j, \qquad j = 1, 2, \ldots, N \\
& \lambda F_{j-1/2}\theta_{j-1/2} - \lambda F_{j+1/2}\theta_{j+1/2} \geq \Gamma^m_j, \qquad j = 1, 2, \ldots, N
\end{aligned}$$

$$0 \leq \theta_{j+1/2} \leq 1. \qquad\qquad j = 0, 1, \ldots, N$$

with $F_{j+1/2}$, Γ_j^M and Γ_j^m defined the same as in the previous section.

We can rewrite the optimization problem in the following form

$$\begin{aligned} \min \quad & \mathbf{f}^T\mathbf{x}, \\ s.t., \quad & \mathbf{A}\mathbf{x} \leq \mathbf{b} \\ & \mathbf{lb} \leq \mathbf{x} \leq \mathbf{ub} \end{aligned}$$

with

$$\mathbf{x} = [x_{1/2}, x_{3/2}, ..., x_{N+1/2}]^T$$

the variables of the problem,

$$\mathbf{f} = [\underbrace{1, 1, \ldots, 1}_{N+1}]^T$$

the coefficients of the objective function,

$$\mathbf{A} = \begin{bmatrix} -\lambda F_{1/2} & \lambda F_{3/2} & & & & \\ & -\lambda F_{3/2} & \lambda F_{5/2} & & & \\ & & \ddots & \ddots & & \\ & & & -\lambda F_{N-1/2} & \lambda F_{N+1/2} \\ \lambda F_{1/2} & -\lambda F_{3/2} & & & & \\ & \lambda F_{3/2} & -\lambda F_{5/2} & & & \\ & & \ddots & \ddots & & \\ & & & \lambda F_{N-1/2} & -\lambda F_{N+1/2} \end{bmatrix}$$

a $2N \times (N+1)$ matrix,

$$\mathbf{b} = [\Gamma_1^M, \ldots, \Gamma_N^M, -\Gamma_1^m, ..., -\Gamma_N^m]^T$$

with elements all positive,

$$\mathbf{lb} = [\underbrace{-1, -1, \ldots, -1}_{N+1}]^T$$

and

$$\mathbf{ub} = [\underbrace{0, 0, \ldots, 0}_{N+1}]^T$$

a set of lower and upper bounds on the design variables $\mathbf{x}$.

This form of linear programming problems can be easily and efficiently solved by the Simplex algorithm [32][16]. And then $\theta_{j+1/2}$ is taken to be $-x_{j+1/2}$ for $j = 0, 1, \ldots, N$.

Invented by Dantzig in 1947, the Simplex algorithm tests adjacent vertices of the feasible set in sequence so that at each new vertex the objective function improves or is unchanged, until the optimal basic feasible solution is identified. This method is very efficient in practice, generally converges in expected polynomial time.

If the original high order numerical solution satisfies the maximum principle itself, all the $\theta_{j+1/2}$ will be taken to be 1 in the Simplex algorithm, thus the limited numerical solution coincides with the original high order numerical solution.

At the final stage of each Runge-Kutta time step, we compute $\{F_{j+1/2}\}_{j=0}^N$, $\{\Gamma_j^M\}_{j=1}^N$ and $\{\Gamma_j^m\}_{j=1}^N$ and solve the global optimization problem defined above. After we obtain the parameters $\{\theta_{j+1/2}\}$, the numerical flux for the cell averages is

modified in the same way as in (2.11), and the rest of the scheme remains the same as in the previous section.

2.5 Numerical Examples

In this section, we apply the parametrized MPP flux limiter as well as the global flux limiter derived from the Simplex algorithm, described in Section 2.3 and Section 2.4, respectively, to the implicit DG method for solving the model problem with three different initial conditions as numerical tests. We present results for P^2 DG method coupled with the 3-stage third order SDIRK33 time discretization. Results are similar with the ESDIRK33 time discretization and are therefore not reported here.

Note that in the computation of both limiters, Γ_j^M should be positive and Γ_j^m should be negative, which comes from the first order method satisfying the maximum principe. But in actual computation, this may not be true due to round off errors, which may eventually affect the value of θ (θ may not be within $[0, 1]$). To address this issue, we monitor Γ_j^M (Γ_j^m) in our computation and set it to be 0 whenever it is negative (positive) with absolute value less than a tolerance 10^{-10}.

For all the tests, we use the uniform mesh with N cell, for $N = 20$ up to 2560, and compute up to $t = 1$. We test on different CFL numbers, especially for large ones. In particular, in the following, we take both CFL= 1 and CFL= 10. For all the three initial/boundary conditions, we will show the results for the implicit DG

scheme without any limiters, with the original parametrized flux limiter, and with the global flux limiter derived from the Simplex algorithm, respectively.

For all figures in this chapter, the solid lines are for the exact solution, and the symbols are for the numerical solution.

Example 2.5.0.1. Equation (2.1) with a Heaviside function as the initial condition

$$\mathbf{u}(x,0) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases}$$

and zero inflow boundary condition.

$u_M = 1$, $u_m = 0$. The results are shown in Figures 2.1 and 2.2.

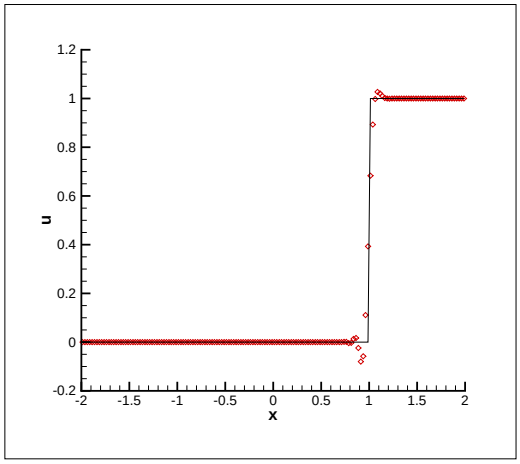
Example 2.5.0.2. Equation (2.1) with a hat function as the initial condition

$$\mathbf{u}(x,0) = \begin{cases} 0, & -2 < x < -1 \\ 1+x, & -1 \leq x \leq 0 \\ 1-x, & 0 \leq x < 1 \\ 0, & 1 < x < 2 \end{cases}$$

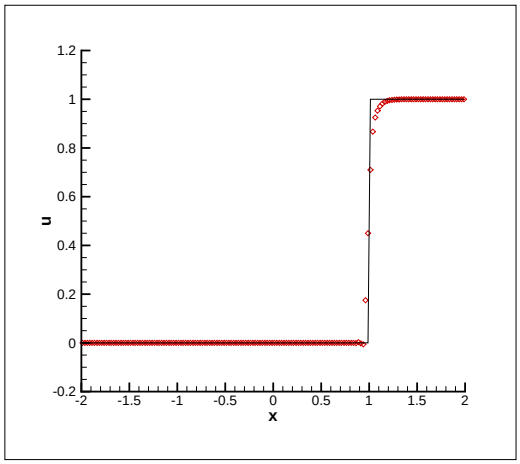
and zero inflow boundary condition at $x = -2$.

$u_M = 1$, $u_m = 0$. The results are shown in Figures 2.3 and 2.4.

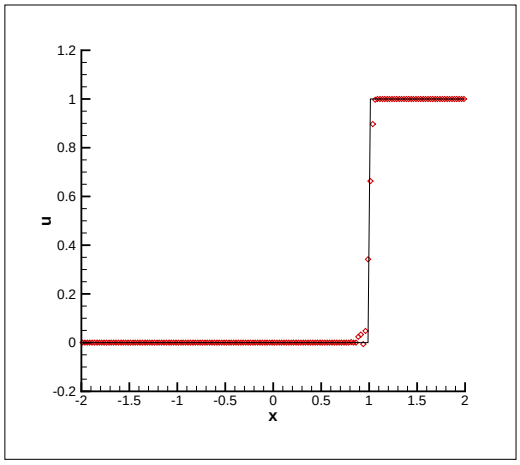
Example 2.5.0.3. Equation (2.1) with a smooth sine function as the initial condi-



(a)

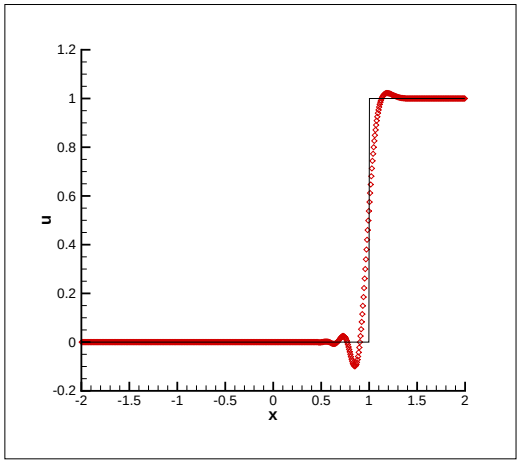


(b)

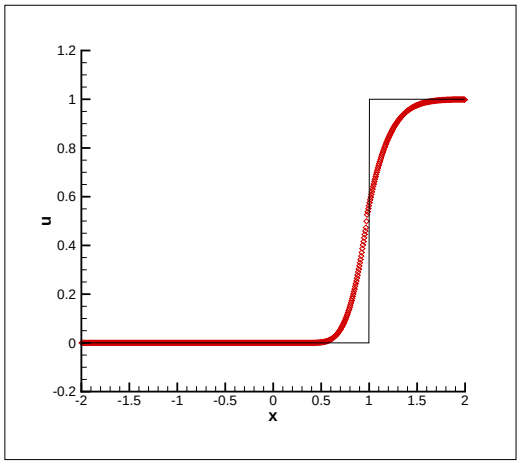


(c)

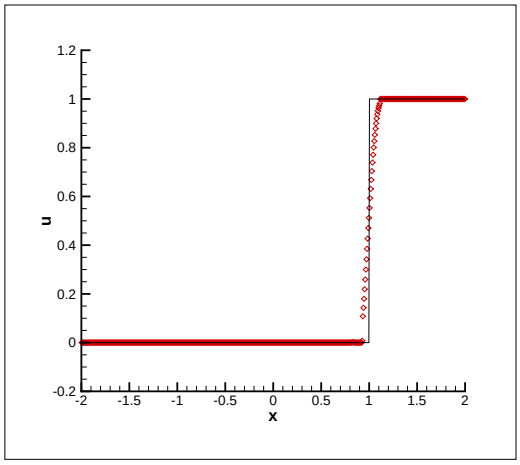
Figure 2.1: 1D transport equation with the Heaviside initial condition using the SDIRK33 P^2 DG sheme with (a) no limiter; (b) original parametrized MPP flux limiter; (c) global flux limiter from the Simplex algorithm. $N = 160$. $CFL = 1$.



(a)



(b)



(c)

Figure 2.2: 1D transport equation with the Heaviside initial condition using the SDIRK33 P^2 DG sheme with (a) no limiter; (b) original parametrized MPP flux limiter; (c) global flux limiter from the Simplex algorithm. $N = 640$. $CFL = 10$.

tion

$$\mathbf{u}(x,0) = \begin{cases} 0, & -2\pi < x < -\pi \\ \sin^4(x), & -\pi \leq x \leq \pi \\ 0, & \pi < x < 2\pi \end{cases}$$

and zero inflow boundary condition at $x = -2\pi$.

$u_M = 1$, $u_m = 0$. The results are shown in Figures 2.5 and 2.6.

For the first two examples with discontinuities, we show figures to demonstrate the effects of the two limiters. For the third one with smooth initial condition, we also record the errors in L_1, L_2 and L_∞ norms, as well as the maximum and minimum value of the cell averages, denoted as $aMax$ and $aMin$, respectively. In our error tables for the global flux limiter with the Simplex algorithm, we observe sometimes $aMin$ can be negative. This is due to the tolerance we set for the Simplex optimization solver.

As can be seen from the results of the first two examples, when the initial wave contains discontinuity and irregularity, the two limiters do take effect and keep the numerical solution staying in bounds. However, the original MPP flux limiter smear the solution a lot when acting on the implicit DG scheme, especially for large CFL numbers. On the other hand, the global flux limiter from the Simplex Algorithm performs a lot better.

For the third smooth initial condition, we observe order degeneration for the original parametrized MPP flux limiter with the magnitude of error significantly larger, while the global flux limiter can preserve third order of accuracy in all norms.

We remark that, since we only preserve the maximum principle for the cell av-

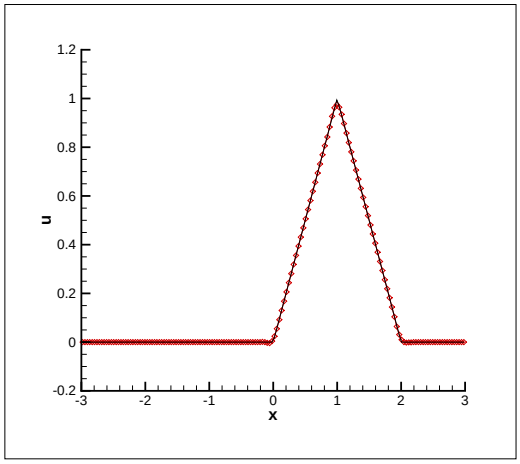
Table 2.1: Error for the DG scheme without limiters with N cells. $T = 1$. $CFL = 1$.

N	L_1 error	order	L_2 error	order	$L\infty$ error	order	aMax	aMin
20	2.31e-01		1.04e-01		8.69e-02		0.77	-1.30e-02
40	6.79e-02	1.77	3.05e-02	1.77	2.23e-02	1.96	0.94	-9.01e-03
80	1.12e-02	2.60	4.98e-03	2.62	3.54e-03	2.65	0.99	-1.85e-03
160	1.54e-03	2.86	6.88e-04	2.86	4.81e-04	2.88	1.00	-2.67e-04
320	2.00e-04	2.95	8.95e-05	2.94	6.20e-05	2.96	1.00	-3.65e-05
640	2.54e-05	2.97	1.14e-05	2.97	7.85e-06	2.98	1.00	-4.64e-06
1280	3.18e-06	3.00	1.43e-06	3.00	9.83e-07	3.00	1.00	-5.85e-07
2560	3.98e-07	3.00	1.79e-07	3.00	1.23e-07	3.00	1.00	-7.33e-08

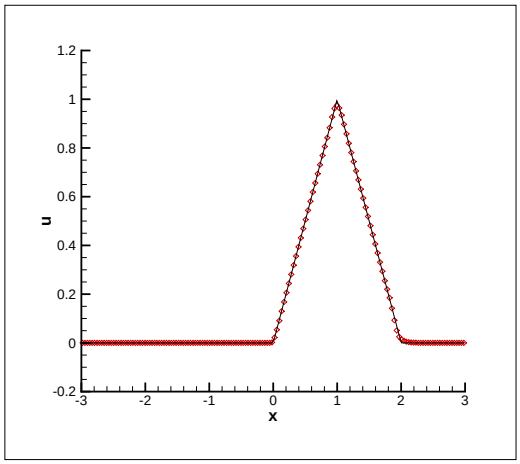
Table 2.2: Error for the DG scheme with the original parametrized flux limiter with N cells. $T = 1$. $CFL = 1$.

N	L_1 error	order	L_2 error	order	$L\infty$ error	order	aMax	aMin
20	3.72e-01		1.65e-01		1.55e-01		0.74	0.00e+00
40	2.91e-01	0.36	1.38e-01	0.26	1.32e-01	0.23	0.93	0.00e+00
80	1.01e-01	1.53	5.63e-02	1.29	5.91e-02	1.16	0.96	0.00e+00
160	1.76e-02	2.52	1.32e-02	2.09	2.03e-02	1.54	0.98	0.00e+00
320	2.28e-03	2.94	2.61e-03	2.34	5.52e-03	1.88	1.00	0.00e+00
640	3.08e-04	2.89	5.00e-04	2.38	1.42e-03	1.96	1.00	0.00e+00
1280	4.43e-05	2.80	9.60e-05	2.38	3.53e-04	2.01	1.00	0.00e+00
2560	6.47e-06	2.78	1.84e-05	2.38	8.74e-05	2.02	1.00	0.00e+00

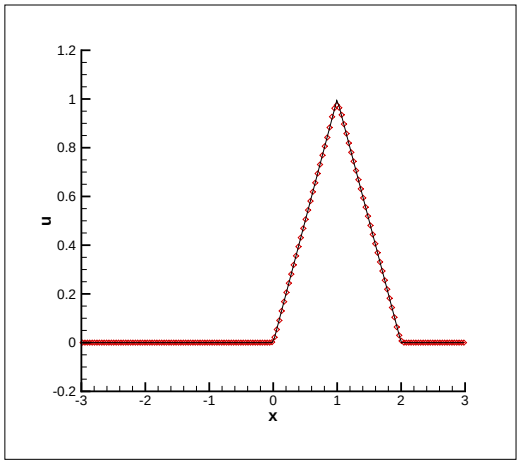
erages, the entire piecewise polynomial of the DG solution with the limiters applied might still be out of the bound $[u_m, u_M]$. One can apply the polynomial rescaling limiters [54] in the final time step only, to ensure the numerical solution to be within the desired bounds.



(a)

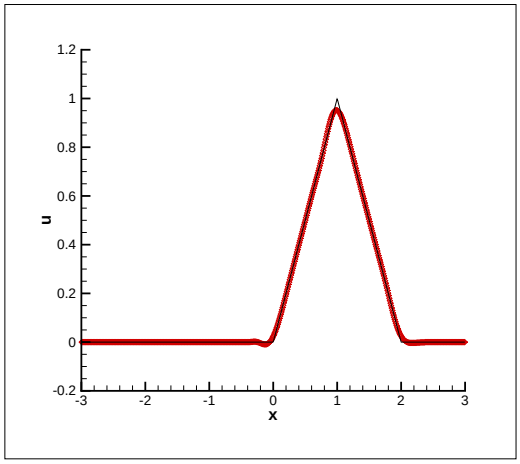


(b)

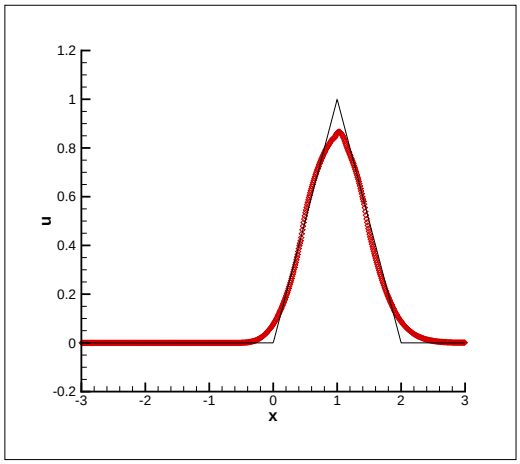


(c)

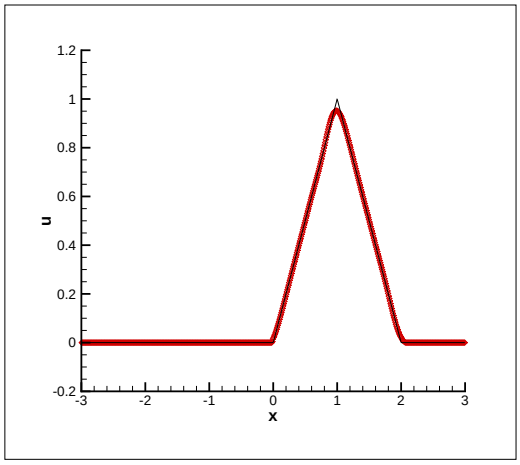
Figure 2.3: 1D transport equation with the Hat initial condition using the SDIRK33 P^2 DG sheme with (a) no limiter; (b) original parametrized MPP flux limiter; (c) global flux limiter from the Simplex algorithm. $N = 160$. $CFL = 1$.



(a)



(b)



(c)

Figure 2.4: 1D transport equation with the Hat initial condition using the SDIRK33 P^2 DG scheme with (a) no limiter; (b) original parametrized MPP flux limiter; (c) global flux limiter from the Simplex algorithm. $N = 640$. $CFL = 10$.

Table 2.3: Error for the DG scheme with the global simplex limiter with N cells. $T = 1$. $CFL = 1$.

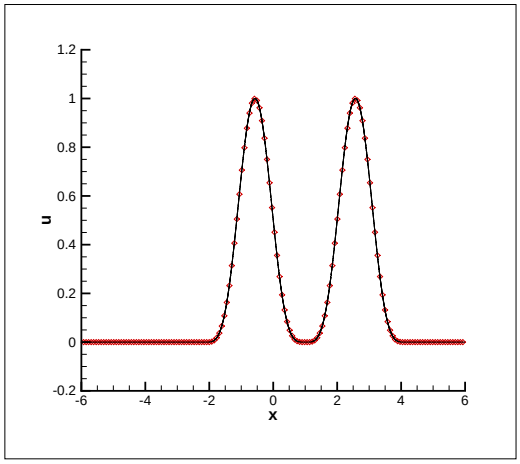
N	L_1 error	order	L_2 error	order	$L\infty$ error	order	aMax	aMin
20	2.19e-01		1.02e-01		8.67e-02		0.77	0
40	5.81e-02	1.91	2.84e-02	1.84	2.23e-02	1.96	0.94	-4.58e-20
80	1.10e-02	2.40	5.11e-03	2.48	3.73e-03	2.58	0.99	-8.59e-07
160	1.56e-03	2.82	7.19e-04	2.83	7.27e-04	2.36	1.00	-4.47e-07
320	2.00e-04	2.97	9.38e-05	2.94	1.23e-04	2.56	1.00	-7.63e-07
640	2.54e-05	2.98	1.17e-05	3.01	1.51e-05	3.03	1.00	-8.57e-07
1280	3.18e-06	3.00	1.43e-06	3.02	9.81e-07	3.94	1.00	-9.07e-07
2560	3.98e-07	3.00	1.79e-07	3.00	1.23e-07	3.00	1.00	-1.03e-07

Table 2.4: Error for the DG scheme without limiters with N cells. $T = 1$. $CFL = 10$.

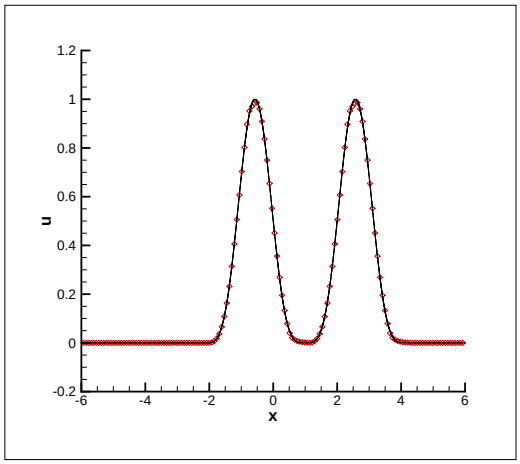
N	L_1 error	order	L_2 error	order	$L\infty$ error	order	aMax	aMin
20	5.85e-01		2.67e-01		2.06e-01		0.77	-4.00e-02
40	5.82e-01	0.01	2.67e-01	0.00	1.99e-01	0.05	0.86	-4.59e-02
80	5.82e-01	0.00	2.67e-01	0.00	2.00e-01	-0.01	0.88	-5.34e-02
160	3.42e-01	0.77	1.56e-01	0.78	1.18e-01	0.77	0.92	-3.20e-02
320	9.95e-02	1.78	4.46e-02	1.81	3.25e-02	1.86	0.97	-1.43e-02
640	2.13e-02	2.22	9.50e-03	2.23	6.69e-03	2.28	0.99	-3.66e-03
1280	3.01e-03	2.83	1.34e-03	2.82	9.29e-04	2.85	1.00	-5.43e-04
2560	3.88e-04	2.95	1.74e-04	2.95	1.19e-04	2.96	1.00	-7.11e-05

Table 2.5: Error for the DG scheme with the original parametrized flux limiter with N cells. $T = 1$. $CFL = 10$.

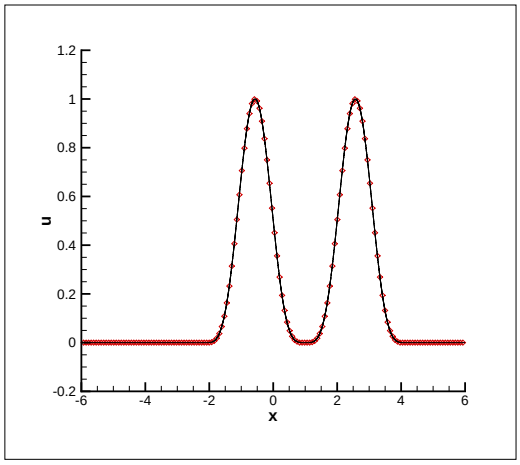
N	L_1 error	order	L_2 error	order	$L\infty$ error	order	aMax	aMin
20	6.82e-01		2.93e-01		2.49e-01		0.77	0.00e+00
40	1.10e+00	-0.69	4.80e-01	-0.71	3.99e-01	-0.68	0.79	0.00e+00
80	1.45e+00	-0.40	6.25e-01	-0.38	5.00e-01	-0.33	0.83	0.00e+00
160	1.23e+00	0.24	5.39e-01	0.21	5.08e-01	-0.02	0.79	0.00e+00
320	8.56e-01	0.52	3.71e-01	0.54	2.84e-01	0.84	0.82	0.00e+00
640	5.52e-01	0.63	2.44e-01	0.61	2.07e-01	0.46	0.86	0.00e+00
1280	2.35e-01	1.23	1.10e-01	1.15	9.75e-02	1.09	0.92	0.00e+00
2560	9.58e-02	1.29	4.94e-02	1.15	4.63e-02	1.07	0.96	0.00e+00



(a)

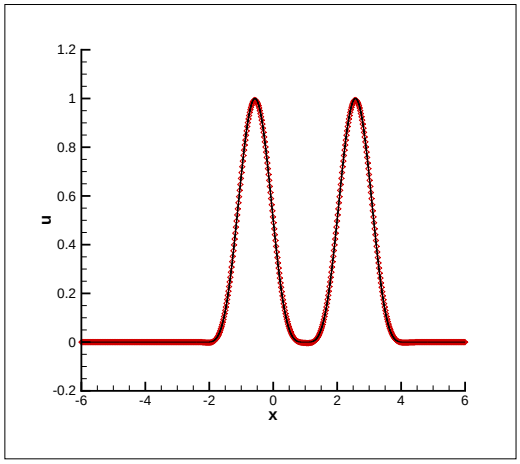


(b)

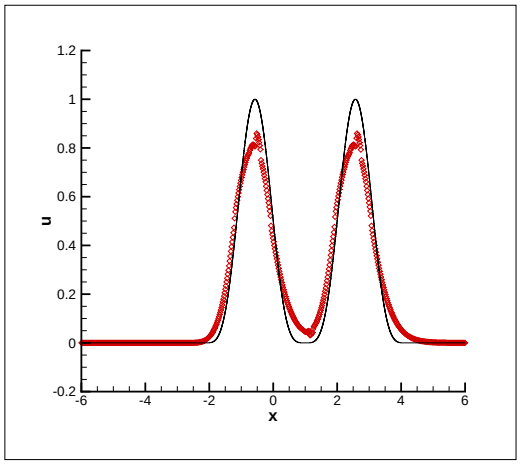


(c)

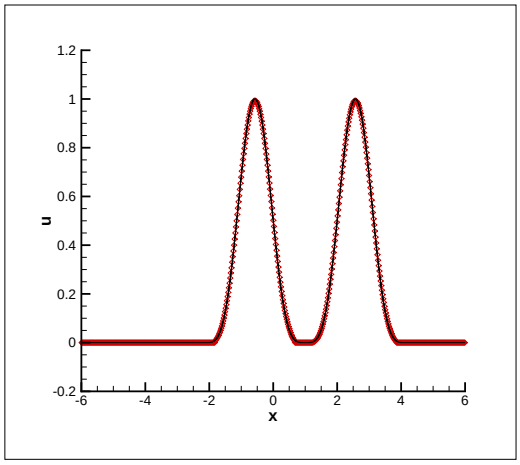
Figure 2.5: 1D transport equation with smooth initial condition using the SDIRK33 P^2 DG sheme with (a) no limiter; (b) original parametrized MPP flux limiter; (c) global flux limiter from the Simplex algorithm. $N = 160$. $CFL = 1$.



(a)



(b)



(c)

Figure 2.6: 1D transport equation with smooth initial condition using the SDIRK33 P^2 DG sheme with (a) no limiter; (b) original parametrized MPP flux limiter; (c) global flux limiter from the Simplex algorithm. $N = 640$. $CFL = 10$.

Table 2.6: Error for the DG scheme with the global simplex limiter with N cells. $T = 1$. $CFL = 10$.

N	L_1 error	order	L_2 error	order	$L\infty$ error	order	aMax	aMin
20	5.41e-01		2.57e-01		2.06e-01		0.77	0.00e+00
40	5.37e-01	0.01	2.60e-01	-0.01	1.99e-01	0.05	0.86	-2.21e-17
80	5.28e-01	0.02	2.60e-01	-0.00	2.00e-01	-0.01	0.88	-1.77e-16
160	3.03e-01	0.80	1.52e-01	0.78	1.18e-01	0.77	0.92	-2.13e-07
320	8.61e-02	1.82	4.29e-02	1.82	3.25e-02	1.86	0.97	-6.82e-07
640	2.01e-02	2.10	9.67e-03	2.15	8.65e-03	1.91	0.99	-9.73e-07
1280	3.05e-03	2.72	1.43e-03	2.76	1.83e-03	2.24	1.00	-9.89e-07
2560	3.87e-04	2.97	1.83e-04	2.96	2.56e-04	2.84	1.00	-9.10e-07

2.6 Conclusion and Future Work

In this chapter, we investigate two kinds of MPP flux limiters to the implicit DG method, for solving the one dimensional transport equations. First we apply the parametrized MPP flux limiter, proposed by Xu [50], to maintain the maximum principle for the cell averages. The limiter is based on the conservation of the DG scheme using the flux difference form in updating cell averages. At every final Runge-Kutta stage, we combine the high order numerical flux with a low order one, by a parameter $\theta_{j+\frac{1}{2}}$, and the formulation to determine $\theta_{j+\frac{1}{2}}$ can be easily implemented with both the high order and low order numerical fluxes computed beforehand.

Based on a similar idea, but by considering the coupled inequalities governing the maximum principle globally, we also investigate another global flux limiter derived from the solution of the proposed optimization problem using the Simplex algorithm. In our numerical tests, this global flux limiter can achieve the maximum principle successfully for the cell averages, with the original order of accuracy and the magnitude of error being maintained.

In future work, general implicit DG schemes for more complicated equations will be investigated with both the local parametrized MPP flux limiter and the global flux limiter. Also, for the global flux limiter, we will explore a more robust and efficient optimization solver.

Chapter Three

Artificial Diffusion WENO Limiter for Discontinuous Galerkin Methods

3.1 Introduction

In this chapter, we consider the following hyperbolic conservation law

$$u_t + f(u)_x = 0, \quad u(x,0) = u_0(x), \tag{3.1}$$

and its two-dimensional version, where u and $f(u)$ can be either scalars or vectors. We investigate a new limiting approach based on weighted essentially non-oscillatory (WENO) limiters for the discontinuous Galerkin (DG) methods to obtain a high order limiting procedure to control spurious numerical oscillations near discontinuities while maintaining uniform high order accuracy in smooth regions. The idea of this limiter is to add an artificial viscosity term containing the difference between the numerical solution and the reconstructed polynomial based on a WENO limiter [59], to the original variational formulation of the DG method.

One of the main difficulties in solving (3.1) is that the solution may contain discontinuities even if the initial condition is smooth. While DG methods have many advantages, when computing solutions with strong discontinuities, the scheme is not enough to control spurious numerical oscillations by itself, and sometimes may even result in nonlinear instabilities. Therefore nonlinear limiters prove to be an important component to control these oscillations, especially for nonlinear problems containing strong shocks.

There are many limiters for DG methods in the literature, most of which come from the methodologies of finite volume and finite difference high resolution schemes. For example, the total variation diminishing (TVD) limiters [19], the total variation bounded (TVB) limiters [41], and the moment-based limiters [4][5]. However sometimes they tend to degenerate accuracy when mistakenly used in smooth regions of

the solution. There is a class of limiters that first identify troubled cells, namely those cells where limiting might be needed, then replace the DG polynomial in those cells by new polynomials of the same degree, while maintaining the original cell averages for conservation. Among this class, the limiters based on the WENO methodology [23][17][20][22][27][30][34][40] designed in recent years are promising, which involve nonlinear reconstructions of the polynomials in troubled cells using the information of neighboring cells. The WENO reconstructed polynomials have the same high order of accuracy as the original polynomials when the solution is smooth, and they are (essentially) non-oscillatory near discontinuities. Qiu and Shu [37] and Zhu et al. [60] designed WENO limiters which abandon all moments in troubled cells except the cell averages and reconstruct those moments from the information of neighboring cells using a WENO methodology. Hermite type WENO limiters [35][36] tend to use more information than the cell averages to reduce the spread of reconstruction stencils. Recently, Zhong and Shu [59][61] developed a new WENO limiting procedure for RKDG methods to simplify the computation. The idea is to reconstruct the entire polynomial instead of point values and the reconstruction polynomial on the target cell is a convex combination of the polynomials on this cell and its immediate neighboring cells, with suitable adjustments for conservation, and with the nonlinear weights of the linear combination following the classical WENO procedure.

However, all previous WENO limiters can be considered as a post-processor of the computed DG solution. Because of this independently add-on procedure, the whole scheme obviously does not have any semi-discrete form, and sometimes it is hard to reach steady state convergence. Our motivation is to change the integral formulation of the DG scheme and incorporate the limiting procedure within the original DG framework, as is preferred by scholars in the finite element fields [31][21]. To do so we add a shock capturing term in the variational formulation

of the DG scheme based on the WENO reconstructed polynomial in [59]. Instead of the traditional artificial diffusion term $-\varepsilon \int u_x v_x$ being added (here v is the test function), we consider replacing u_x with $(u_x - (Pu)_x)$, where Pu is the reconstructed polynomial in the current cell I_j as computed in [59]. This is in spirit similar to spectral vanishing viscosity [31], except that Pu uses information both from the current cell and from neighboring cells, thus is more reasonable in predicting high frequency errors for hyperbolic problems. As the difference between u and Pu is large around the shocks, while small in the smooth region, we expect the extra term to act as an artificial viscosity to control spurious numerical oscillations near discontinuities while maintaining the original order of accuracy. With the extra term added, the new DG scheme is expected to perform better for steady state problems and achieve good analytical results for suitable choice of ε.

The rest of this chapter is organized as follows. We first present an overview of the RKDG method in Section 3.2. After we review the WENO limiter [59] in Section 3.3, we present our new artificial diffusion WENO limiter in the Section 3.4 and perform preliminary finite difference analysis in Section 3.5. Extensive numerical examples are tested in Section 3.6. And finally in Section 3.7 we give the concluding remarks.

3.2 Discontinuous Galerkin Method Formulation

In this section, we give an overview of the algorithm formulation of the discontinuous Galerkin method solving the hyperbolic conservation law (3.1) and its two dimensional version. The semi-discrete DG scheme solving (3.1) has already been given in (1.3), therein we use the Lax-Friedrichs flux for all of our numerical tests in this chapter

$$\hat{f}^{LF}(u^-, u^+) = \frac{1}{2}(f(u^-) + f(u^+) - \alpha(u^+ - u^-)),$$

where

$$\alpha = \max_u |f'(u)|.$$

For the two dimensional version

$$u_t + f(u)_x + g(u)_y = 0, \quad u(x, y, 0) = u_0(x, y), \tag{3.2}$$

where $f(u)$ and $g(u)$ can be vectors, the DG scheme with Euler forward time discretization on a rectangular mesh becomes

$$\begin{aligned}\int_{I_{i,j}} \frac{u^{n+1} - u}{\Delta t}\, v\, dxdy - \int_{I_{i,j}} f(u)\, v_x\, dxdy + \int_{J_j} \hat{f}_{i+\frac{1}{2}}(y)\, v(x^-_{i+\frac{1}{2}}, y)dy \\ - \int_{J_j} \hat{f}_{i-\frac{1}{2}}(y)\, v(x^+_{i-\frac{1}{2}}, y)dy - \int_{I_{i,j}} g(u)\, v_y\, dxdy + \int_{I_i} \hat{g}_{j+\frac{1}{2}}(x)\, v(x, y^-_{j+\frac{1}{2}})dx \\ - \int_{I_i} \hat{g}_{j-\frac{1}{2}}(x)\, v(x, y^+_{j-\frac{1}{2}})dx = 0\end{aligned} \tag{3.3}$$

where $u = u^n$, and $\hat{f}$ and $\hat{g}$ are numerical fluxes.

For the time discretization, we use the TVD Runge-Kutta time discretization

introduced in [43]. Denote $L(u,t)$ as the spatial discretization operator

$$u_t = L(u,t),$$

then the third-order Runge-Kutta method has the following form

$$\begin{aligned} u^{(1)} &= u^n + \Delta t L(u^n, t^n), && (3.4)\\ u^{(2)} &= \frac{3}{4}u^n + \frac{1}{4}u^{(1)} + \frac{1}{4}\Delta t L(u^{(1)}, t^n + \Delta t), && (3.5)\\ u^{n+1} &= \frac{1}{3}u^n + \frac{2}{3}u^{(2)} + \frac{2}{3}\Delta t L(u^{(2)}, t^n + \frac{1}{2}\Delta t). && (3.6) \end{aligned}$$

Other TVD, or strong stability preserving (SSP) time discretizations can also be used. For details, see the review paper [18].

The WENO limiters in [59] and our new limiting procedure we are about to introduce take place in each of the Runge-Kutta stage for every time level.

3.3 WENO Reconstructed Polynomial

In this section, we review the procedure of the WENO limiter in [59], which is also the way to determine Pu in our new method.

The WENO limiting procedure consists of two parts:

1. The first step is to identify the troubled cells, namely those cells which might need the limiting procedure. In our computation, we use the usual minmod type TVB limiters [41][11][9] as the troubled cell indicator. That is, whenever the minmod limiter changes the slope, the cell is declared to be a troubled cell.

However, we point out that the troubled cell indicator is not our emphasis here. We would like to see that, even if the indicator mistakenly identifies some cells to be troubled, it should not affect the performance of the method. So we omit the details of the identification of troubled cells here.

In our numerical tests, we observe that if we apply our new artificial diffusion WENO limiter on all cells of the domain, the scheme performs well, and there is not much difference in the numerical solution, whether to use the troubled cell indicator or not.

2. The second step is to replace the solution polynomials in the troubled cells by reconstructed polynomials which maintain the original cell averages (for conservation), have the same order of accuracy as before, but are less oscillatory.

Assume the current cell is a troubled cell, the idea of the previous WENO limiter is to reconstruct a new polynomial on it, which is a convex combination of the polynomials on this cell and its immediate neighboring cells, with necessary adjustments to keep the original cell average on the target cell. The nonlinear weights in the convex combination coefficients follow the classical WENO procedure. In this section, we mainly review this way to compute the WENO reconstructed polynomial.

1. One dimensional scalar case. For the current cell I_j, denote the DG solution polynomial of u on the cells I_{j-1}, I_j , I_{j+1} as $p_0(x)$, $p_1(x)$, $p_2(x)$, respectively.

We make the following modifications:

$$\tilde{p}_0(x) = p_0(x) - \bar{p}_0 + \bar{p}_1, \qquad \tilde{p}_2(x) = p_2(x) - \bar{p}_2 + \bar{p}_1, \tag{3.7}$$

where

$$\begin{aligned}
\bar{p}_0 &= \frac{1}{\Delta x_j} \int_{I_j} p_0(x)dx, \\
\bar{p}_1 &= \frac{1}{\Delta x_j} \int_{I_j} p_1(x)dx, \\
\bar{p}_2 &= \frac{1}{\Delta x_j} \int_{I_j} p_2(x)dx,
\end{aligned}$$

The final reconstructed polynomial is defined as follows

$$Pu(x) = P_1^{new}(x) = \omega_0\tilde{p}_0(x) + \omega_1 p_1(x) + \omega_2\tilde{p}_2(x), \tag{3.8}$$

which has the same cell average and order of accuracy as p_1 if the weights satisfy $\omega_0 + \omega_1 + \omega_2 = 1$.

As in [23][3], the nonlinear weights are defined as

$$\omega_l = \frac{\bar{\omega}_l}{\sum_s \bar{\omega}_s}, \tag{3.9}$$

where

$$\bar{\omega}_l = \frac{\gamma_l}{(\epsilon + \beta_l)^r}. \tag{3.10}$$

We use $\epsilon = 10^{-6}$ and $r = 2$ in our computation. γ_l are the linear weights and β_l are the smoothness indicators, defined as

$$\beta_l = \sum_{s=1}^{k} \int_{I_j} \Delta x_j^{2s-1} (\frac{\partial^s}{\partial x^s} p_l(x))^2 dx. \tag{3.11}$$

The linear weights can be chosen as any set of positive numbers adding up to one. In our tests we take

$$\gamma_0 = 0.001, \qquad \gamma_2 = 0.998, \qquad \gamma_2 = 0.001.$$

Thus we complete the procedure to compute $Pu(x)$ in the one dimensional scalar case.

2. For two dimensional scalar case, with rectangular meshes considered in this chapter, the WENO reconstructed polynomial is computed in a similar way as described above, only with four neighboring cells (cells $I_{i-1,j}$, $I_{i+1,j}$, $I_{i,j-1}$ and $I_{i,j+1}$ for cell $I_{i,j}$). For our numerical experiments, the linear weight is taken to be 0.996 for the center cell, and 0.001 for all four neighboring cells. The computation of the smoothness indicator and the nonlinear weights follows the standard WENO procedure

$$\beta_l = \sum_{1 \le |\alpha| \le k} \int_{I_{i,j}} (\Delta x_i \Delta y_j)^{|\alpha|-1} (D^\alpha p_l(x,y))^2 dx\, dy \tag{3.12}$$

where $\alpha = (\alpha_1, \alpha_2)$, $|\alpha| = \alpha_1 + \alpha_2$ and $D^\alpha = \frac{\partial^\alpha}{\partial x^{\alpha_1} \partial y^{\alpha_2}}$. For the two dimensional case and more details about this smoothness indicator, we refer to [3][23][42].

3. For one dimensional systems, i.e., when u and f are vectors with m components in Equation (3.1), in order to achieve better non-oscillatory qualities, the WENO reconstruction limiter is applied with a local characteristic field decomposition. For the current cell $I_{i,j}$, first compute

$$R(\bar{u}_j) = (r_j^{(1)}, r_j^{(2)}, \ldots, r_j^{(m)}), \tag{3.13}$$

and its inverse R^{-1}, where $r_j^{(p)}$, $p = 1, 2, \ldots, m$, are the normalized right eigen-

vectors of the Jacobian matrix $A_j = \frac{\partial f}{\partial u}$ evaluated at the cell average $\bar{u}_j$.

Denote the DG polynomial (m - component) u on the cells I_{j-1}, I_j , I_{j+1} as u_0, u_1, u_2, respectively. Project u_i ($i = 0, 1, 2$) into the characteristic fields $v_i = R^{-1}u_i$, for $i = 0, 1, 2$. Note that v_i is a m-component vector with each component a polynomial. Reconstruct each component of v_1 as in the one dimensional scalar case, to get the updated vector v_1^{new}. Then the reconstructed polynomial in the current cell is taken as $Pu(x) = u_1^{new}(x) = Rv_1^{new}$.

4. For two dimensional systems, we perform the characteristic-wise WENO reconstruction in the x-direction and y-direction separately to get two corresponding reconstructed polynomials in each direction.

   (a) In the x-direction, use the eigenspace of the Jacobian matrix $\frac{\partial f}{\partial u}$ and polynomials on the cells $I_{i-1,j}$, $I_{i,j}$ and $I_{i+1,j}$ to get the reconstructed polynomial in the x-direction, $Pu^{(x)}(x, y)$, as in the one dimensional case.

   (b) In the y-direction, use the eigenspace of the Jacobian matrix $\frac{\partial g}{\partial u}$ and polynomials on the cells $I_{i,j-1}$, $I_{i,j}$ and $I_{i,j+1}$ to get the reconstructed polynomial in the y-direction, $Pu^{(y)}(x, y)$, as in the one dimensional case.

3.4 Artificial Diffusion WENO Limiter

In this section, we describe our new artificial diffusion limiting approach, based on the WENO reconstructed polynomials computed in the previous section. For simplicity,

the limiter is applied on all cells here.

In one dimension, we consider the DG scheme (1.3) in the weak form

$$\int_{I_j} u_t\, v\, dx - \int_{I_j} f(u)\, v_x\, dx + \hat{f}_{j+\frac{1}{2}}\, v(x^-_{j+\frac{1}{2}}) - \hat{f}_{j-\frac{1}{2}}\, v(x^+_{j-\frac{1}{2}}) = 0 \tag{3.14}$$

At every time level t^n, instead of solving the original DG scheme, we add an extra term $-\varepsilon \int_{I_j} (u_x - (Pu)_x) v_x dx$ to the RHS of the DG scheme, where ε is a pre-determined parameter, and $Pu(x)$ is the new polynomial we reconstructed for every cell in the previous section. Then the scheme becomes

$$\int_{I_j} u_t\, v\, dx - \int_{I_j} f(u)\, v_x\, dx + \hat{f}_{j+\frac{1}{2}}\, v(x^-_{j+\frac{1}{2}}) - \hat{f}_{j-\frac{1}{2}}\, v(x^+_{j-\frac{1}{2}}) = -\varepsilon \int_{I_j} (u_x - (Pu)_x) v_x dx \tag{3.15}$$

In one dimension, we take $\varepsilon = C \cdot \Delta x$ so as not to degenerate the order of accuracy of the origianl DG scheme. The constant C is taken to be, for example, 0.1, 1, 10, in our numerical tests. For different problems, ε may be chosen suitably and differently, but we keep it uniform among all cells in our computation.

In two dimension, the weak formulation of the DG scheme solving (3.2) is

$$\begin{aligned}
&\int_{I_{i,j}} u_t\, v\, dxdy - \int_{I_{i,j}} f(u)\, v_x\, dxdy + \int_{J_j} \hat{f}_{i+\frac{1}{2}}(y)\, v(x^-_{i+\frac{1}{2}}, y) dy \\
&\quad - \int_{J_j} \hat{f}_{i-\frac{1}{2}}(y)\, v(x^+_{i-\frac{1}{2}}, y) dy - \int_{I_{i,j}} g(u)\, v_y\, dxdy + \int_{I_i} \hat{g}_{j+\frac{1}{2}}(x)\, v(x, y^-_{j+\frac{1}{2}}) dx
\end{aligned}$$

$$-\int_{I_i} \hat{g}_{j-\frac{1}{2}}(x)\, v(x, y^+_{j-\frac{1}{2}})dx = 0 \quad (3.16)$$

This time we add to the RHS, $-\varepsilon \int_{I_{i,j}} (u_x - (Pu^{(x)})_x)v_x + (u_y - (Pu^{(y)})_y)\, v_y dxdy$ as the extra artificial diffusion term to the RHS of the DG scheme, where ε is a constant parameter, $Pu^{(x)}$ is the newly reconstructed polynomial in the x-direction, and $Pu^{(y)}$ is the newly reconstructed polynomial in the y-direction. The scheme becomes,

$$\begin{aligned}
&\int_{I_{i,j}} u_t\, v\, dxdy - \int_{I_{i,j}} f(u)\, v_x\, dxdy + \int_{J_j} \hat{f}_{i+\frac{1}{2}}(y)\, v(x^-_{i+\frac{1}{2}}, y)dy \\
&\quad - \int_{J_j} \hat{f}_{i-\frac{1}{2}}(y)\, v(x^+_{i-\frac{1}{2}}, y)dy - \int_{I_{i,j}} g(u)\, v_y\, dxdy + \int_{I_i} \hat{g}_{j+\frac{1}{2}}(x)\, v(x, y^-_{j+\frac{1}{2}})dx \\
&\quad - \int_{I_i} \hat{g}_{j-\frac{1}{2}}(x)\, v(x, y^+_{j-\frac{1}{2}})dx = \\
&\quad - \varepsilon \int_{I_{i,j}} (u_x - (Pu^{(x)})_x)v_x + (u_y - (Pu^{(y)})_y)v_y\, dxdy \quad (3.17)
\end{aligned}$$

In two dimension, we take ε as a constant, for example, 0.01, 0.1, 1, in our numerical tests.

We solve the new DG schemes (3.15) and (3.17), instead of the original ones (3.14)-(3.16) to achieve the desired properties of the limiter. The extra term added is supposed to act as an artificial viscosity, whose magnitude is controlled by the difference between u and Pu. This is desirable for us, since the difference are presumably big around the shocks, while small in the smooth region. In the next section, we will give an analysis to show that the extra term added is indeed a numerical viscosity.

We summarize our new artificial diffusion WENO limiting procedure case by case

as follows.

At every time level t^n, denote the current numerical solution to be u^n.

1. One dimensional scalar case. For the current cell I_j, denote u^n on the cells I_{j-1}, I_j , I_{j+1} as $p_0(x)$, $p_1(x)$, $p_2(x)$, respectively.

   (a) Compute the RHS of the original DG scheme (3.14).

   (b) Rewrite p_0 and p_2 in cell I_j and compute the corresponding coefficients.

   (c) Choose the linear weights γ_i, compute the smoothness indicator β_i by (3.11), and then the nonlinear weights by ω_i (3.10)-(3.9), for $i = 1, 2, 3$.

   (d) Compute the convex combination (3.8) to get the reconstructed polynomial Pu.

   (e) Compute the derivative of $Pu(x)$ and then the numerical integral of $-\int_{I_j}(u_x - Pu_x)v_x dx$.

   (f) Multiply the integral by ε and add it to the RHS of the DG scheme. Solve for the degrees of freedom and integrate in time to get u^{n+1}.

2. Two dimension scalar case. For the current cell $I_{i,j}$, denote u^n on the cells $I_{i,j}$, $I_{i-1,j}$, $I_{i+1,j}$, $I_{i,j+1}$, $I_{i,j-1}$ as p_0, p_1, p_2, p_3, p_4, respectively.

   (a) Compute the RHS of the original DG scheme (3.16).

   (b) Rewrite p_l $(l = 1, 2, 3, 4)$ in cell $I_{i,j}$ and compute the corresponding coefficients.

   (c) Choose the linear weights γ_l, compute the smoothness indicator β_l by (3.12), and then the nonlinear weights ω_l by (3.10)-(3.9). $l = 0, 1, 2, 3, 4$.

   (d) Compute the convex combination of p_l $(l = 0, 1, 2, 3, 4)$ to get the reconstructed polynomial Pu.

(e) Compute the derivatives $(Pu(x,y))_x$ and $(Pu(x,y))_y$ and then the numerical integral of $-\int_{I_{i,j}}(u_x-(Pu)_x)v_x+(u_y-(Pu)_y)v_y\,dx\,dy$.

(f) Multiply the integral by ε and add it to the RHS of the DG scheme. Solve for the degrees of freedom and integrate in time to get u^{n+1}.

3. One dimensional system case. For the current cell I_j, denote u^n on the cells I_{j-1}, I_j, I_{j+1} as $u_0(x)$, $u_1(x)$, $u_2(x)$, respectively.

(a) Compute the RHS of the original DG scheme (3.14).

(b) Compute the cell average $\bar{u}_j$ and matrices $R(\bar{u}_j)$ and R^{-1}, as in (3.13).

(c) Project u_i $(i=1,2,3)$ to the characteristic fields to get $v_i=R^{-1}u_i$.

(d) For each component of v_i, repeat the steps (b)-(d) as in the one dimensional scalar case to get the modified v_1^{new}.

(e) Project back to get the reconstructed polynomial $Pu=Rv_1^{new}$.

(f) Compute the derivative of $Pu(x)$ and then the numerical integral of $-\int_{I_j}(u_x-(Pu)_x)v_x dx$.

(g) Multiply the integral by ε and add it to the RHS of the DG scheme. Solve for the degrees of freedom and integrate in time to get u^{n+1}.

4. Two dimensional system case.

(a) Compute the RHS of the original DG scheme (3.16).

(b) Compute the cell average $\bar{u}_{i,j}$ and matrices $R_{(x)}=\frac{\partial f}{\partial u}$ and $R_{(x)}^{-1}$ in the x-direction.

(c) Compute the matrices $R_{(y)}=\frac{\partial g}{\partial u}$ and $R_{(y)}^{-1}$ in the y-direction.

(d) Use $R_{(x)}$ and polynomials $u_{i-1,j}$, $u_{i,j}$, $u_{i+1,j}$ to repeat steps (c) to (e) as in the one dimensional system case to get the reconstructed polynomial in the x-direction, $Pu^{(x)}(x,y)$.

(e) Use $R_{(y)}$ and polynomials $u_{i,j-1}$, $u_{i,j}$, $u_{i,j+1}$ to repeat steps (c) to (e) as in the one dimensional system case to get the reconstructed polynomial in the y-direction, $Pu^{(y)}(x,y)$.

(f) Compute the derivatives $\frac{\partial Pu^{(x)}(x,y)}{\partial x}$ and $\frac{\partial Pu^{(y)}(x,y)}{\partial y}$, and then the numerical integral of $-\int_{I_{i,j}}(u_x-(Pu^{(x)})_x)v_x+(u_y-(Pu^{(y)})_y)v_y dxdy$.

(g) Multiply the integral by ε and add it to the RHS of the DG scheme. Solve for the degrees of freedom and integrate in time to get u^{n+1}.

3.5 Analysis

In this section, we perform Fourier analysis to the modified DG scheme in the variational form in finite difference fashion. We rewrite the discontinuous Galerkin method as finite difference schemes[52], and then use finite difference techniques to analyze the extra term added to the RHS of the scheme. This is to ensure that we have indeed added a numerical viscosity.

We consider the one dimensional wave equation,

$$u_t + u_x = 0, \qquad x \in [a,b],$$
$$u(x,0) = u_0(x)$$

The DG scheme to solve this equation has been addressed in (1.3).

We look at the implementation of the scheme (1.3). If a local basis of $P^k_{I_j}$ is chosen and denoted as $\phi^l_j(x)$ for $l = 1, 2, \ldots, k+1$, then the numerical solution can be represented as

$$u(x) = \sum_{i=1}^{k+1} u^l_j \phi^l_j(x), \qquad x \in I_j. \tag{3.18}$$

After substituting (3.18) into (1.3) and inverting a local $(k+1) \times (k+1)$ mass matrix, the DG scheme can be written in a compact way

$$\frac{d\mathbf{u}_j}{dt} = \frac{a}{\Delta x}(A\mathbf{u}_j + B\mathbf{u}_{j-1}), \tag{3.19}$$

where $\mathbf{u}_j = (u^1_j, \ldots, u^{k+1}_j)^T$, A and B are constant matrices.

As in [52], we choose the degrees of freedom for the k-th degree polynomial inside the cell I_j as the point values of the solution in a uniform mesh, denoted by

$$u_{j+\frac{2i-k}{2(k+1)}}, \qquad i = 0, \ldots, k,$$

at the $k+1$ equally spaced points

$$(j + \frac{2i-k}{2(k+1)})\Delta x, \qquad i = 0, \ldots, k.$$

The schemes written in terms of these degrees of freedom become finite difference schemes on a globally uniform mesh (with a mesh size $\Delta x/(k+1)$). Note that they may not be standard finite difference schemes since each points in the group of $k+1$ points belonging to the cell I_j obeys a different form of the finite difference scheme.

We discuss the P^1 and P^2 cases separately as follows.

3.5.1 k=1

For the piecewise linear $k = 1$ case, we choose the degrees of freedom as the point values at the $2N$ uniformly spaced points

$$u_{j-\frac{1}{4}},\ u_{j+\frac{1}{4}}, \qquad j = 1, \ldots, N.$$

The local basis $\phi_{j-\frac{1}{4}}(x)$ is taken to be the linear polynomial which equals 1 at the point $(x_j - \frac{1}{4}\Delta x)$ and equals 0 at the point $(x_j + \frac{1}{4}\Delta x)$. Similarly $\phi_{j-\frac{1}{4}}(x)$ is taken to be the linear polynomial which equals 1 at the point $(x_j + \frac{1}{4}\Delta x)$ and equals 0 at the point $(x_j - \frac{1}{4}\Delta x)$. Thus the solution inside the cell I_j is represented by

$$u(x) = u_{j-\frac{1}{4}}\phi_{j-\frac{1}{4}}(x) + u_{j+\frac{1}{4}}\phi_{j+\frac{1}{4}}(x).$$

Now we are ready to evaluate the added term $-\varepsilon \int_{I_j}(u_x - (Pu)_x)v_x dx$, where

$$u_x - (Pu)_x = (1 - \omega_1)(P_1)_x - \omega_0(P_0)_x - \omega_2(P_2)_x.$$

We write out the local basis on cell I_j

$$\phi_{j-\frac{1}{4}}(x) = -\tfrac{2}{\Delta x}(x - (x_j + \tfrac{1}{4}\Delta x))$$
$$\phi_{j+\frac{1}{4}}(x) = \tfrac{2}{\Delta x}(x - (x_j - \tfrac{1}{4}\Delta x))$$

Thus,

$$(P_1)_x(x) = u_{j-\frac{1}{4}}\phi_{j-\frac{1}{4}x} + u_{j+\frac{1}{4}}\phi_{j+\frac{1}{4}x} = \frac{2}{\Delta x}(u_{j+\frac{1}{4}} - u_{j-\frac{1}{4}}) \tag{3.20}$$

Similarly,

$$(P_0)_x(x) = \frac{2}{\Delta x}(u_{j-\frac{3}{4}} - u_{j-\frac{5}{4}}),$$

$$(P_2)_x(x) = \frac{2}{\Delta x}(u_{j+\frac{5}{4}} - u_{j+\frac{3}{4}}),$$

Therefore we have

$$u_x - (Pu)_x = \frac{2}{\Delta x}[(1-\omega_1)(u_{j+\frac{1}{4}} - u_{j-\frac{1}{4}}) - \omega_0(u_{j-\frac{3}{4}} - u_{j-\frac{5}{4}}) - \omega_2(u_{j+\frac{5}{4}} - u_{j+\frac{3}{4}})] \quad (3.21)$$

Taking the test function v to be $\phi_{j-\frac{1}{4}}(x)$ and $\phi_{j-\frac{1}{4}}(x)$, respectively, evaluating the integral $-\varepsilon \int_{I_j}(u_x - (Pu)_x)v_x dx$, and inverting the mass matrix, we obtain the expression for the extra term

$$-\frac{6\varepsilon}{\Delta x^2}\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}[(1-\omega_1)\mathbf{u}_j - \omega_0\mathbf{u}_{j-1} - \omega_2\mathbf{u}_{j+1}] \quad (3.22)$$

As the nonlinear weights ω_i involve complex computation from WENO methodology, we consider only the linear weights γ_i in (3.22), and assume

$$\gamma_0 = \gamma_2, \qquad \gamma_0 + \gamma_1 + \gamma_2 = 1,$$

to get

$$\frac{6\varepsilon\gamma_0}{\Delta x^2}\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}[\mathbf{u}_{j-1} - 2\mathbf{u}_j + \mathbf{u}_{j+1}]. \quad (3.23)$$

As the two eigenvalues of the matrix $\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$ are non-negative, and the term $[\mathbf{u}_{j-1} - 2\mathbf{u}_j + \mathbf{u}_{j+1}]$ is an approximation to the second derivative, we conclude

that the extra term added to the scheme is indeed a numerical viscosity.

3.5.2 k=2

For the $k = 2$ case, the degrees of freedom are the point values at the $3N$ uniformly spaced points

$$u_{j-\frac{1}{3}}, u_j, u_{j+\frac{1}{3}}, \qquad j = 1, \ldots, N.$$

The local basis functions $\{\phi_{j-\frac{1}{3}}(x), \phi_j(x), \phi_{j+\frac{1}{3}}(x)\}$ inside cell I_j are the Lagrangian polynomials based on the points $x_{j-\frac{1}{3}}, x_j, x_{j+\frac{1}{3}}$, i.e.,

$$\begin{aligned}
\phi_{j-\frac{1}{3}}(x) &= \frac{9}{2}\Delta x^{-2}(x - x_j)(x - x_{j+\frac{1}{3}}) \\
\phi_j(x) &= -9\Delta x^{-2}(x - x_{j-\frac{1}{3}}(x - x_{j+\frac{1}{3}}) \\
\phi_{j+\frac{1}{3}}(x) &= \frac{9}{2}\Delta x^{-2}(x - x_{j-\frac{1}{3}})(x - x_j)
\end{aligned}$$

The solution inside cell I_j is then represented by

$$u(x) = u_{j-\frac{1}{3}}\phi_{j-\frac{1}{3}}(x) + u_j\phi_j(x) + u_{j+\frac{1}{3}}\phi_{j+\frac{1}{3}}(x) \tag{3.24}$$

Taking the derivative, we get

$$(P_1)_x(x) = \frac{9}{2}\Delta x^{-2}[u_{j-\frac{1}{3}}(2x - x_j - x_{j+\frac{1}{3}}) + u_j(2x - x_{j-\frac{1}{3}} - x_{j+\frac{1}{3}}) + u_{j+\frac{1}{3}}(2x - x_j - x_{j-\frac{1}{3}})] \tag{3.25}$$

Similarly one can get the formula for $(P_0)_x(x)$ and $(P_2)_x(x)$. Therefore,

$$
\begin{aligned}
&u_x - (Pu)_x \\
&= -\omega_0 9\Delta x^{-2}\{u_{j-\frac{4}{3}}\frac{1}{2}(2x - x_{j-1} - x_{j-\frac{2}{3}}) - u_{j-1}(2x - x_{j-\frac{4}{3}} - x_{j-\frac{2}{3}}) \\
&+u_{j-\frac{2}{3}}(2x - x_j - x_{j-\frac{1}{3}})\} + (1-\omega_1)9\Delta x^{-2}\{u_{j-\frac{1}{3}}\frac{1}{2}(2x - x_j - x_{j+\frac{1}{3}}) \\
&-u_j(2x - x_{j-\frac{1}{3}} - x_{j+\frac{1}{3}}) + u_{j+\frac{1}{3}}\frac{1}{2}(2x - x_j - x_{j-\frac{1}{3}})\} \\
&-\omega_2 9\Delta x^{-2}\{u_{j+\frac{2}{3}}\frac{1}{2}(2x - x_{j+1} - x_{j+\frac{4}{3}}) - u_{j+1}(2x - x_{j+\frac{2}{3}} - x_{j+\frac{4}{3}}) \\
&+u_{j+\frac{4}{3}}(2x - x_{j+\frac{2}{3}} - x_{j+1})\}
\end{aligned}
$$

Take the test function v to be $\phi_{j-\frac{1}{3}}(x)$, $\phi_j(x)$ and $\phi_{j+\frac{1}{3}}(x)$, respectively, and assume $\omega_i = \gamma_i$ and $\gamma_0 = \gamma_2$. The integral $-\varepsilon\int_{I_j}(u_x - Pu_x)v_x dx$ is evaluated to be

$$
\begin{pmatrix} -0.5 & 1.5 & -1 \\ -1.5 & 3 & -1.5 \\ 2 & -4.5 & 2.5 \end{pmatrix}\mathbf{u}_{j-1} - 2\begin{pmatrix} 1 & -1.5 & 0.5 \\ -1.5 & 3 & -1.5 \\ 0.5 & -1.5 & 1 \end{pmatrix}\mathbf{u}_j
$$
$$
+\begin{pmatrix} 2.5 & -4.5 & 2 \\ -1.5 & 3 & -1.5 \\ -1 & 1.5 & -0.5 \end{pmatrix}\mathbf{u}_{j+1} \quad (3.26)
$$

with a coefficient $9\Delta x^{-2}\gamma_0\varepsilon$ in the front.

By performing Taylor expansion around the corresponding point for each component of this term (3.26), we obtain the leading terms

$$
\left(\frac{1}{6}\Delta x^3 u_{xxx}|_{x_{j-\frac{1}{3}}}, -\frac{1}{6}\Delta x^4 u_{xxxx}|_{x_j}, -\frac{1}{6}\Delta x^3 u_{xxx}|_{x_{j+\frac{1}{3}}}\right)^T,
$$

which is a fourth order derivative in the middle and two third order derivatives with

the opposite sign for the other two points. This is a kind of numerical viscosity, but may also introduce some numerical dispersion error along.

3.6 Numerical Examples

In this section, we provide numerical experiments to demonstrate the performance of our new limiting methodology for the RKDG methods described in Section 3.4. For simplicity, for both one and two dimensions, we use structured and uniform meshes in our simulation.

For all the computational results, we use the local Lax-Friedrichs flux. For the one-dimensional examples, the CFL number is taken as 0.2 for the P^1 case, 0.1 for the P^2 case and 0.05 for the P^3 case. For all numerical results shown below, the limiter is applied in all cells.

For the one dimensional figures, the solid lines are for the exact solutions, and the symbols are for the numerical solutions.

Table 3.1: Error for the 1D Burgers' equation at $t = 0.5$ using the DG scheme without limiter.

N	P^1				P^2			
	L_1 error	order	L_∞ error	order	L_1 error	order	L_∞ error	order
20	3.81E-03		3.83E-02		2.73E-04		5.04E-03	
40	9.45E-04	2.01	1.02E-02	1.90	4.22E-05	2.70	8.95E-04	2.49
80	2.35E-04	2.01	2.65E-03	1.95	6.17E-06	2.77	1.60E-04	2.48
160	5.89E-05	2.00	6.73E-04	1.98	8.86E-07	2.80	2.55E-05	2.65
320	1.47E-05	2.00	1.70E-04	1.98	1.25E-07	2.82	3.79E-06	2.75
640	3.69E-06	2.00	4.27E-05	1.99	1.74E-08	2.85	5.27E-07	2.85

3.6.1 Scalar conservation laws

Example 3.6.1.1. We solve the Burgers' equation

$$u_t + (\frac{u^2}{2})_x = 0,\ x \in [0, 2\pi] \tag{3.27}$$

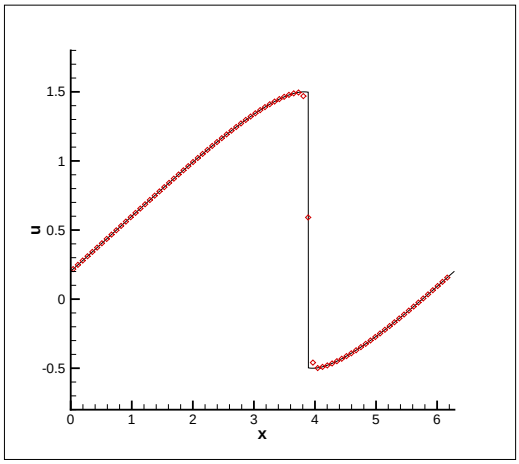
with periodic boundary conditions. The initial condition $u(x, 0) = 0.5 + \sin(x)$. The exact solution is smooth up to $t = 1$, then it develops a moving shock which interacts with a rarefaction wave. For smooth solutions at $t = 0.5$, we document the errors with the original DG scheme in Table 3.1, with the limiter for $\varepsilon = 0.3\Delta x$ in Table 3.2, and with the limiter for $\varepsilon = \Delta x$ in Table 3.3. We can clearly see from the latter two tables that the designed order of accuracy is achieved in the L_1 and L_∞ norm, and the magnitude of the errors of the original RKDG method is maintained. To make the solution non-oscillatory, we set $\varepsilon = 0.3\Delta x$ and $\varepsilon = \Delta x$, and plot the solutions for different k and ε in Figure 3.1. We can see that the shock is captured very well at $t = 1.5$ without oscillations. As also can be observed, when ε increases, the solution do become more dissipative.

Table 3.2: Error for the 1D Burgers' equation at $t = 0.5$ using the DG scheme with limiter. $\varepsilon = 0.3\Delta x$.

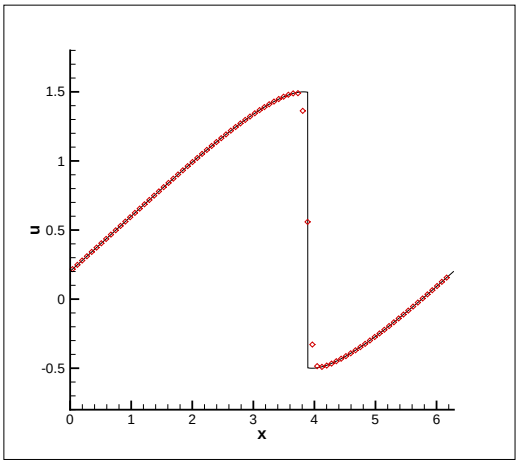
N	P^1				P^2			
	L_1 error	order	L_∞ error	order	L_1 error	order	L_∞ error	order
20	5.72E-03		5.35E-02		2.68E-04		5.06E-03	
40	1.57E-03	1.86	1.25E-02	2.10	4.14E-05	2.69	8.80E-04	2.52
80	3.10E-04	2.34	2.65E-03	2.24	6.03E-06	2.78	1.58E-04	2.48
160	5.96E-05	2.38	6.73E-04	1.98	8.63E-07	2.80	2.49E-05	2.66
320	1.47E-05	2.02	1.70E-04	1.98	1.22E-07	2.82	3.70E-06	2.75
640	3.69E-06	2.00	4.27E-05	1.99	1.70E-08	2.85	5.14E-07	2.85

Table 3.3: Error for the 1D Burgers' equation at $t = 0.5$ using the DG scheme with limiter. $\varepsilon = \Delta x$.

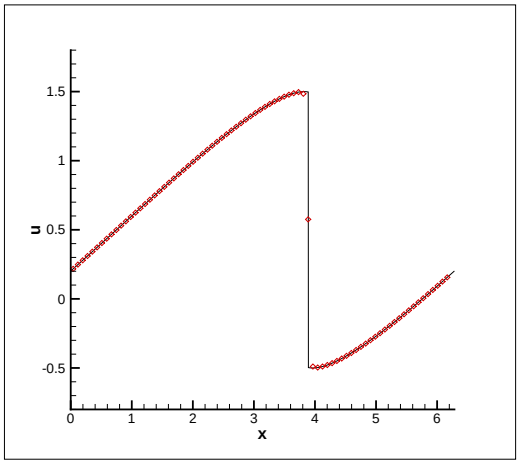
N	P^1				P^2			
	L_1 error	order	L_∞ error	order	L_1 error	order	L_∞ error	order
20	1.01E-02		6.31E-02		2.61E-04		5.01E-03	
40	2.19E-03	2.20	1.84E-02	1.78	3.99E-05	2.71	8.45E-04	2.57
80	4.17E-04	2.39	3.98E-03	2.21	5.75E-06	2.80	1.52E-04	2.47
160	6.37E-05	2.71	6.72E-04	2.57	8.15E-07	2.82	2.38E-05	2.68
320	1.48E-05	2.11	1.70E-04	1.98	1.15E-07	2.83	3.51E-06	2.76
640	3.69E-06	2.00	4.27E-05	1.99	1.60E-08	2.85	4.85E-07	2.85



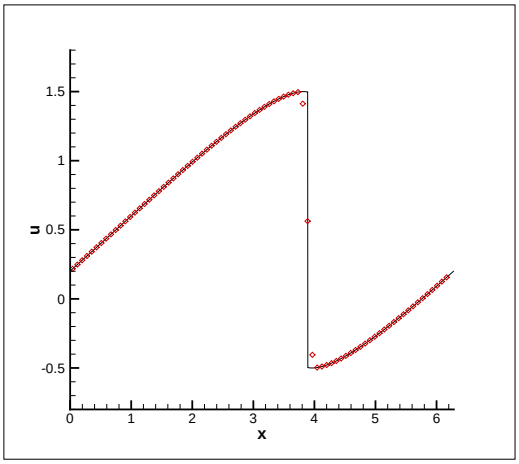
(a) $\varepsilon = 0.3 \cdot \Delta x$, $k = 1$



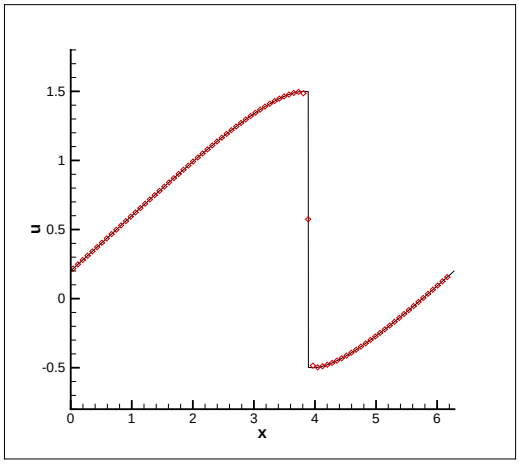
(b) $\varepsilon = 1 \cdot \Delta x$, $k = 1$



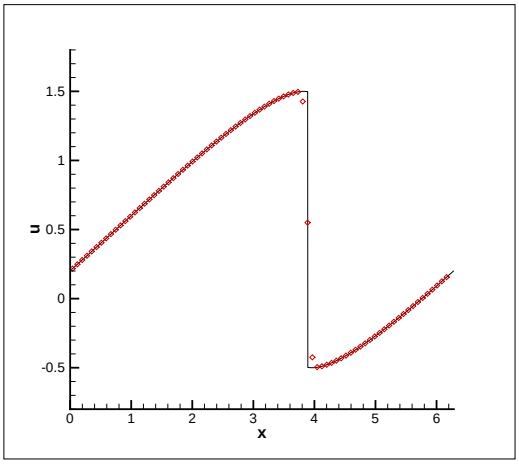
(c) $\varepsilon = 0.3 \cdot \Delta x$, $k = 2$



(d) $\varepsilon = 1 \cdot \Delta x$, $k = 2$



(e) $\varepsilon = 0.3 \cdot \Delta x$, $k = 3$



(f) $\varepsilon = 1 \cdot \Delta x$, $k = 3$

Figure 3.1: Numerical solution for the Burgers' equation at $t = 1.5$ using the P^k DG scheme with limiter for $\varepsilon = 0.3 \cdot \Delta x$ and $\varepsilon = 1 \cdot \Delta x$. $k = 1, 2, 3$. $N = 80$.

Example 3.6.1.2. We test the Buckley-Leverett problem

$$u_t + (\frac{4u^2}{4u^2 + (1-u)^2})_x = 0. \tag{3.28}$$

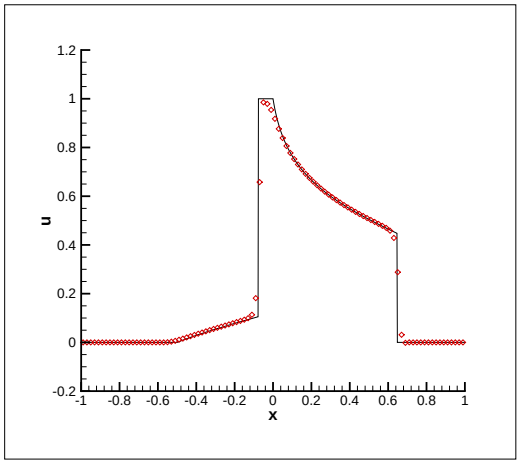
The initial condition is taken as $u = 1$ in $[-\frac{1}{2}, 0]$ and $u = 0$ elsewhere. We test on the Buckley-Leverett problem up to $t = 0.4$. The exact solution is a shock-rarefaction-contact discontinuity mixture. We use the Lax-Friedrichs flux for the DG scheme and apply the artificial diffusion WENO limiter on every cell, with $\varepsilon = 0.6\Delta x$. The computational result is displayed in Figure 3.2, which is quite satisfactory.

Example 3.6.1.3. We solve the two dimensional Burgers' equation

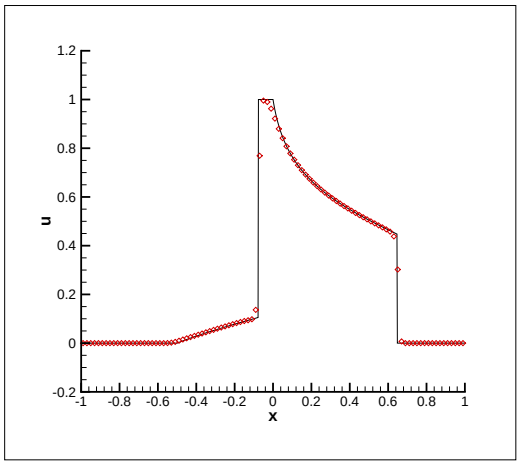
$$u_t + (\frac{u^2}{2})_x + (\frac{u^2}{2})_y = 0,\ x, y \in [0, 2\pi] \tag{3.29}$$

with periodic boundary condition. The initial condition is $u(x, y, 0) = 0.5+\sin(x+y)$.

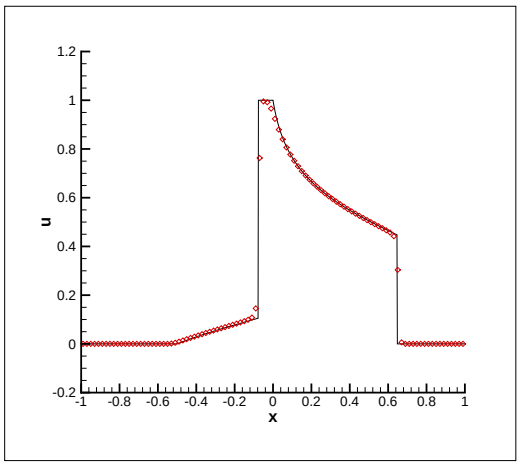
As in the one dimensional Burgers' example, we collect the L_1 and L_∞ errors at $t = 0.25$, when the solution is smooth, in Tables 3.4, 3.5 and 3.6, for $\varepsilon = 0$ (without limiter), 0.01 and 0.1, respectively. Again we can see that the errors with limiters maintain the same high order accuracy and very close in magnitude of the original DG method. A shock begins to form at $t = 0.5$. We compute the solutions of the RKDG methods using P^k polynomials with our new limiter with 80×80 meshes until $t = 0.75$ and plot the solution on the diagonal cells in Figures 3.3. We can see the sharp, non-oscillatory shock transitions for our methods for $k = 1$, 2 and 3.



(a) $k = 1$



(b) $k = 2$



(c) $k = 3$

Figure 3.2: Numerical solution for the Buckley Leverett problem at $t = 0.4$ using the DG scheme with limiter for $\varepsilon = 0.6\Delta x$. $k = 1, 2, 3$. $N = 100$.

Table 3.4: Error for the 2D Burgers' equation at $t = 0.25$ using the DG scheme without limiter.

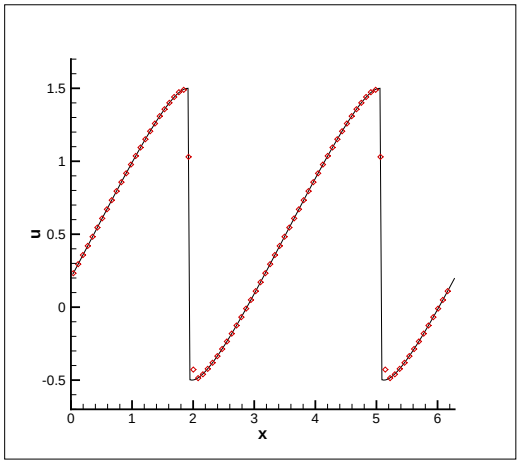
N	P^1				P^2			
	L_1 error	order	L_∞ error	order	L_1 error	order	L_∞ error	order
20×20	7.56E-03		1.23E-01		8.62E-04		4.37E-02	
40×40	1.92E-03	1.98	3.55E-02	1.79	1.16E-04	2.89	6.08E-03	2.84
80×80	4.79E-04	2.00	9.36E-03	1.92	1.50E-05	2.96	9.60E-04	2.66
160×160	1.20E-04	2.00	2.39E-03	1.97	1.90E-06	2.98	1.35E-04	2.83
320×320	2.99E-05	2.00	6.04E-04	1.98	2.43E-07	2.97	1.83E-05	2.88

Table 3.5: Error for the 2D Burgers' equation at $t = 0.25$ using the DG scheme with limiter. $\varepsilon = 0.01$.

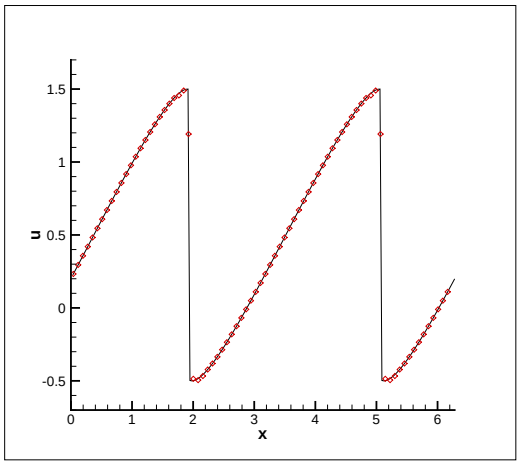
N	P^1				P^2			
	L_1 error	order	L_∞ error	order	L_1 error	order	L_∞ error	order
20×20	7.52e-3		1.23e-1		8.62E-04		4.36E-02	
40×40	1.96e-3	1.94	3.55e-2	1.79	1.16E-04	2.89	6.08E-03	2.84
80×80	5.18e-4	1.92	9.36e-3	1.92	1.50E-05	2.95	9.57E-04	2.67
160×160	1.22e-4	2.09	2.39e-3	1.97	1.89E-06	2.99	1.33E-04	2.85
320×320	3.00e-5	2.02	6.04e-4	1.98	2.38E-07	2.99	1.79E-05	2.89

Table 3.6: Error for the 2D Burgers' equation at $t = 0.25$ using the DG scheme with limiter. $\varepsilon = 0.1$.

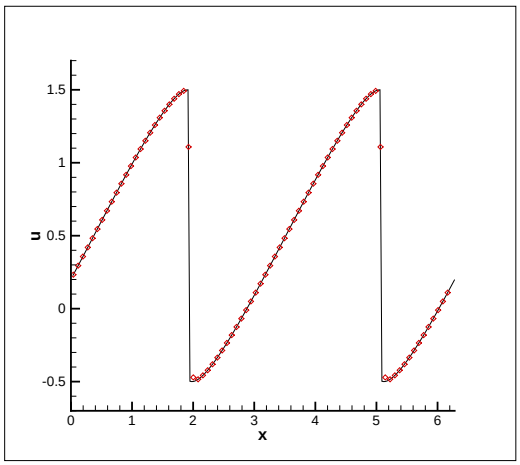
N	P^1				P^2			
	L_1 error	order	L_∞ error	order	L_1 error	order	L_∞ error	order
20×20	8.52E-03		1.35E-01		8.63E-04		4.35E-02	
40×40	3.07E-03	1.47	4.04E-02	1.74	1.17E-04	2.88	5.99E-03	2.86
80×80	8.69E-04	1.82	9.99E-03	2.02	1.50E-05	2.96	9.29E-04	2.69
160×160	1.58E-04	2.46	2.38E-03	2.07	1.90E-06	2.98	1.23E-04	2.92
320×320	3.28E-05	2.27	6.01E-04	1.99	2.51E-07	2.92	1.47E-05	3.06



(a) $k = 1$



(b) $k = 2$



(c) $k = 3$

Figure 3.3: Numerical solution for the two dimensional Burgers' equation at $t = 0.75$ using the DG scheme with limiter for $\varepsilon = 0.1$. $k = 1, 2, 3$. $N_x = N_y = 80$.

3.6.2 Euler system in one dimension

We test on the Euler system for our new artificial diffusion WENO limiter, the one dimensional version is given by

$$\mathbf{u}_t + \mathbf{f}(\mathbf{u})_x = 0 \tag{3.30}$$

where

$$\mathbf{u} = (\rho, \rho v, E)^T,$$

and

$$\mathbf{f}(\mathbf{u}) = (\rho v, \rho v^2, v(E+P))^T,$$

with

$$P = (\gamma - 1)(E - \frac{1}{2}\rho v^2),$$

where ρ is the density, v is the velocity, E is the total energy, P is the pressure, and $\gamma > 1$ is a constant ($\gamma = 1.4$ for the air). The speed of sound is given by $c = \sqrt{\gamma P/\rho}$ and the three eigenvalues of the Jacobian matrix are $v - c$, v and $v + c$. For details of the Jacobian, its eigenvalues, eigenvectors, etc., see [20][39].

Example 3.6.2.1. This example is to test the order of accuracy for the DG method with our artificial diffusion WENO limiter. The initial condition is set to be $\rho(x, 0) = 1 + 0.2 \sin x$, $v(x, 0) = 1$, $p(x, 0) = 1$ and the boundaries are periodic. The exactly solution is $\rho(x, t) = 1 + 0.2 \sin(x - t)$, $v(x, t) = 1$ and $p(x, t) = 1$. We collect the L_1 and L_∞ errors at $t = 2\pi$ in Tables 3.7, 3.8 and 3.9. Table 3.7 list the errors and orders of accuracy for the DG method without limiters, and Tables 3.8 and 3.9 list the DG method with our limiter for $\varepsilon = 0.1\Delta x$ and $0.5\Delta x$, respectively. We can see that out limiter maintains both the designed order of accuracy and the magnitude

Table 3.7: Error for the 1D Euler equation with the initial condition $\rho(x,0) = 1 + 0.2\sin x$, $v(x,0) = 1$, $p(x,0) = 1$ at $t = 2\pi$ using the DG scheme with N cells.

N	P^1				P^2			
	L_1 error	order	L_∞ error	order	L_1 error	order	L_∞ error	order
20	4.91e-04		2.98e-03		2.96e-05		1.56e-04	
40	1.29e-04	1.93	7.80e-04	1.93	3.83e-06	2.95	2.04e-05	2.94
80	3.33e-05	1.95	1.99e-04	1.97	4.83e-07	2.99	2.57e-06	2.99
160	8.52e-06	1.97	5.04e-05	1.98	6.05e-08	3.00	3.22e-07	3.00
320	2.15e-06	1.99	1.27e-05	1.99	7.57e-09	3.00	4.02e-08	3.00

Table 3.8: Error for the 1D Euler equation with the initial condition $\rho(x,0) = 1 + 0.2\sin x$, $v(x,0) = 1$, $p(x,0) = 1$ at $t = 2\pi$ using the DG scheme with the limiter. $\varepsilon = 0.1\Delta x$. N cells.

N	P^1				P^2			
	L_1 error	order	L_∞ error	order	L_1 error	order	L_∞ error	order
20	3.36e-03		1.02e-02		2.95e-05		1.56e-04	
40	3.35e-04	3.33	1.65e-03	2.63	3.83e-06	2.95	2.03e-05	2.94
80	3.52e-05	3.25	2.17e-04	2.93	4.83e-07	2.99	2.56e-06	2.99
160	8.53e-06	2.05	5.01e-05	2.11	6.05e-08	3.00	3.21e-07	3.00
320	2.15e-06	1.99	1.25e-05	2.00	7.57e-09	3.00	4.02e-08	3.00

of accuracy of the original DG method.

Example 3.6.2.2. Sod and Lax problems come from physical experiments where a gas tube is separated by a membrane into two sections. The gas in each section is uniform in the y and z direction, so the problem is modeled as in one dimension. We consider the Riemann problems of the Euler equation of gas dynamics for a polytropic gas

$$\mathbf{u}(x,0) = \begin{cases} \mathbf{u}_L, & x < 0 \\ \mathbf{u}_R, & x > 0 \end{cases} \tag{3.31}$$

Table 3.9: Error for the 1D Euler equation with the initial condition $\rho(x,0) = 1+0.2\sin x$, $v(x,0)=1$, $p(x,0)=1$ at $t=2\pi$ using the DG scheme with the limiter. $\varepsilon = 0.5\Delta x$. N cells.

N	P^1				P^2			
	L_1 error	order	L_∞ error	order	L_1 error	order	L_∞ error	order
20	8.49e-03		1.71e-02		3.15e-05		1.56e-04	
40	8.92e-04	3.25	3.31e-03	2.37	4.13e-06	2.93	2.04e-05	2.93
80	5.41e-05	4.04	3.13e-04	3.40	5.25e-07	2.98	2.56e-06	2.99
160	8.95e-06	2.60	5.27e-05	2.57	6.59e-08	2.99	3.24e-07	2.98
320	2.16e-06	2.05	1.23e-05	2.10	8.24e-09	3.00	4.07e-08	2.99

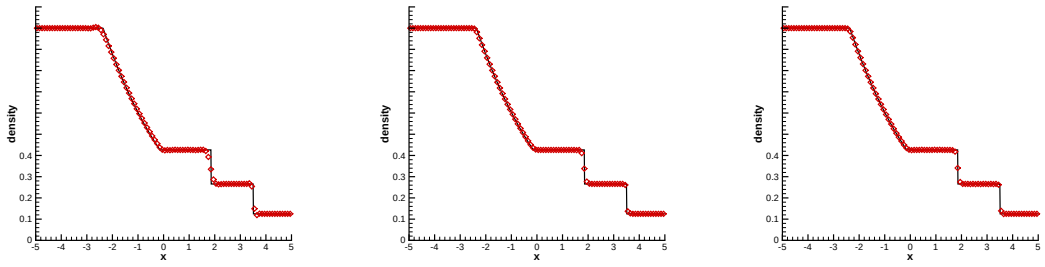
Two sets of initial conditions are considered. One is proposed by Sod [45]:

$$(\rho_L, v_L, P_L) = (1,0,1); \qquad (\rho_R, v_R, P_R) = (0.125, 0, 0.1).$$

The other is used by Lax [25]:

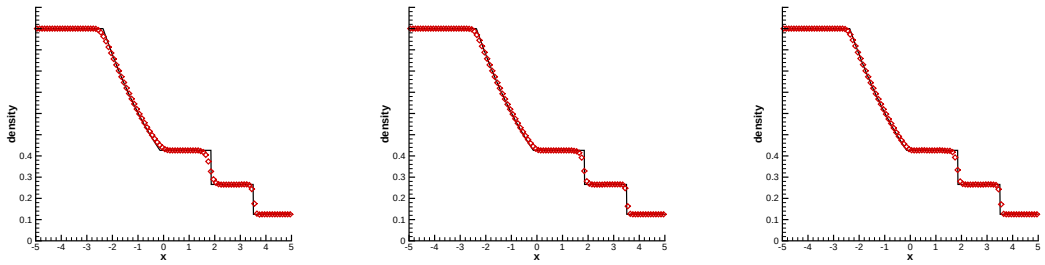
$$(\rho_L, v_L, P_L) = (0.445, 0.698, 3.528); \qquad (\rho_R, v_R, P_R) = (0.5, 0, 0.571).$$

The solution to these two problems contains three waves propagating from the discontinuity at the initial time. We compute up to time $t = 2$ for the Sod problem and $t = 1.3$ for the Lax problem, and results for density are plotted against the exact solution for different k and ε. Figure 3.4 shows the results of the Sod problem for $\varepsilon = 0.05\Delta x$, where we can see the oscillations near discontinuities are successfully controlled by the limiter, yet the method can still capture the shock transition without much smearing. As ε increases to $0.5\Delta x$ in Figure 3.5 and $10\Delta x$ in Figure 3.6, the solutions are more and more dissipative as we expect. Similarly, we take $\varepsilon = 0.05\Delta x$, $0.1\Delta x$, $0.5\Delta x$, $10\Delta x$ in Figures 3.7, 3.8, 3.9, 3.10, respectively. For the Lax problem, the solutions act more sensitively to the parameter ε, and for different k, the optimal parameter ε is different. For solutions containing shocks, we



(a) $k = 1$ (b) $k = 2$ (c) $k = 3$

Figure 3.4: Sod problem at $t = 2$ using the P^k DG scheme with different k. $\varepsilon = 0.05 \cdot \Delta x$. $N = 100$.



(a) $k = 1$ (b) $k = 2$ (c) $k = 3$

Figure 3.5: Sod problem at $t = 2$ using the P^k DG scheme with different k. $\varepsilon = 0.5 \cdot \Delta x$. $N = 100$.

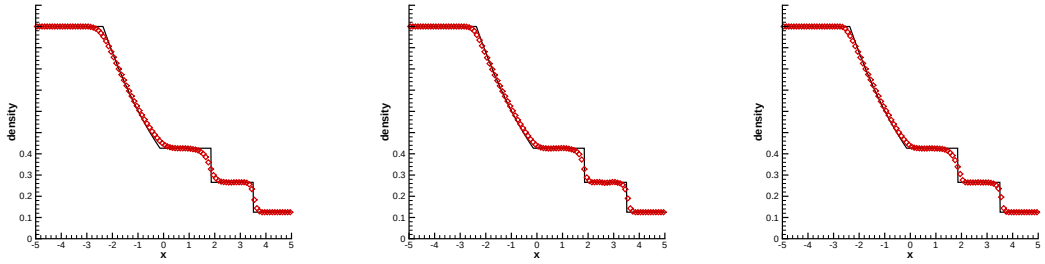
observe that the bigger k is, the less ε needs to be to make the numerical solution non-oscillatory near the shock.

Example 3.6.2.3. First proposed by Shu and Osher in [43], this test is a hydrodynamic shocktube where the left and the right states are given as follows

$$\mathbf{u}(x,0) = \begin{cases} \mathbf{u}_L, & x < -4 \\ \mathbf{u}_R, & x \geq -4 \end{cases} \tag{3.32}$$

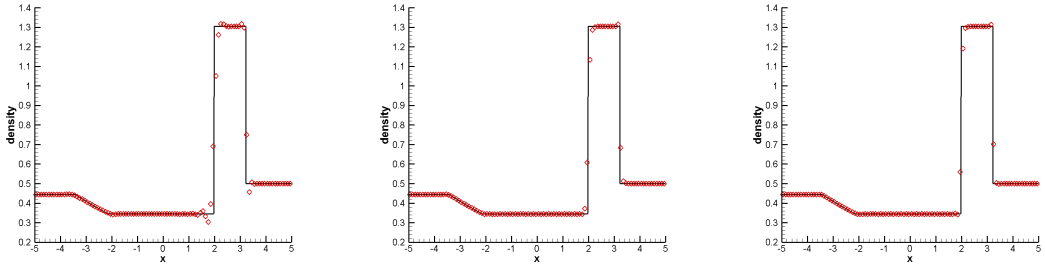
with the data

$$(\rho_L, v_L, P_L) = (3.857143, 2.629369, 10.33333),$$



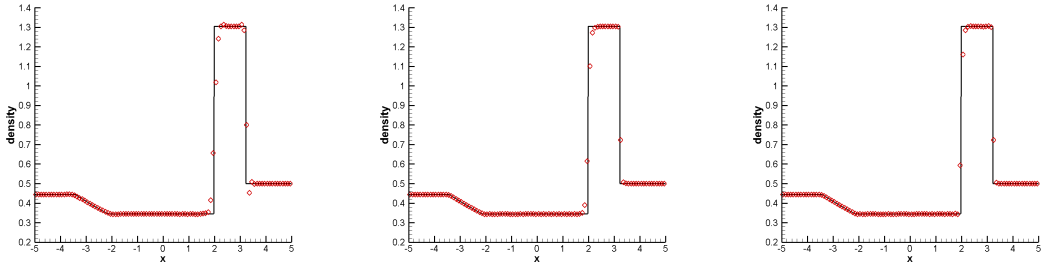
(a) $k = 1$ (b) $k = 2$ (c) $k = 3$

Figure 3.6: Sod problem at $t = 2$ using the P^k DG scheme with different k. $\varepsilon = 10 \cdot \Delta x$. $N = 100$.



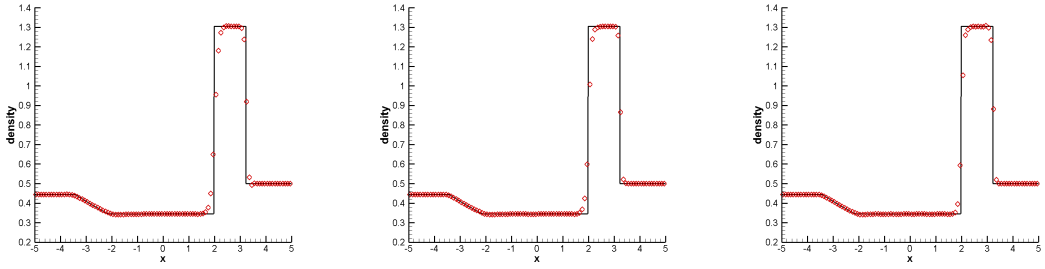
(a) $k = 1$ (b) $k = 2$ (c) $k = 3$

Figure 3.7: Lax problem at $t = 1.3$ using the P^k DG scheme with different k. $\varepsilon = 0.05 \cdot \Delta x$. $N = 100$.



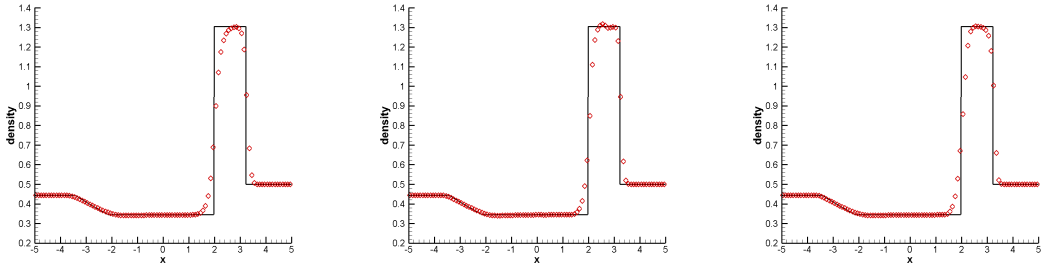
(a) $k = 1$ (b) $k = 2$ (c) $k = 3$

Figure 3.8: Lax problem at $t = 1.3$ using the P^k DG scheme with different k. $\varepsilon = 0.1 \cdot \Delta x$. $N = 100$.



(a) $k = 1$ (b) $k = 2$ (c) $k = 3$

Figure 3.9: Lax problem at $t = 1.3$ using the P^k DG scheme with different k. $\varepsilon = 0.5 \cdot \Delta x$. $N = 100$.



(a) $k = 1$ (b) $k = 2$ (c) $k = 3$

Figure 3.10: Lax problem at $t = 1.3$ using the P^k DG scheme with different k. $\varepsilon = 10 \cdot \Delta x$. $N = 100$.

$$(\rho_R, v_R, P_R) = (1 + 0.2\sin(5x), 0, 1),$$

which describe a moving Mach 3 shock interacting with sine waves in density. This example is to show the advantage in using higher order schemes when the solution contains both shocks and complex smooth region structures.

The solution is computed up to time $t = 1.8$ and compared against a converged solution computed by the fifth order finite difference WENO scheme with 2000 grid points. We use $N = 200$ in our DG scheme with limiters to obtain the numerical solution in Figures 3.11, 3.12 and 3.13 for $\varepsilon = 0.2\Delta x$, $0.5\Delta x$ and Δx, respectively. We observe that the fine structure in the density profile makes the higher order schemes ($k = 2, 3$) perform much better than the lower order methods, and they can capture the structure really well.

Example 3.6.2.4. We consider the interaction of blast waves of the Euler equation (3.30) with the initial condition

$$\mathbf{u}(x,0) = \begin{cases} \mathbf{u}_L, & 0 \le x < 0.1 \\ \mathbf{u}_M, & 0.1 \le x < 0.9 \\ \mathbf{u}_R, & 0.9 \le x < 1 \end{cases} \tag{3.33}$$

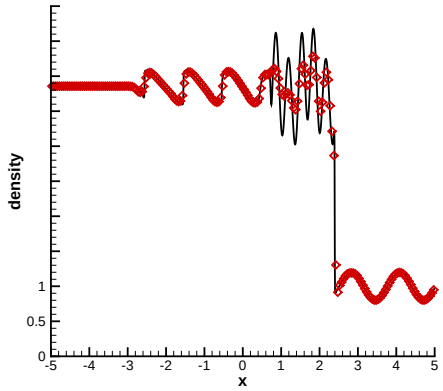
with the following data

$$(\rho_L, v_L, P_L) = (1, 0, 1000),$$

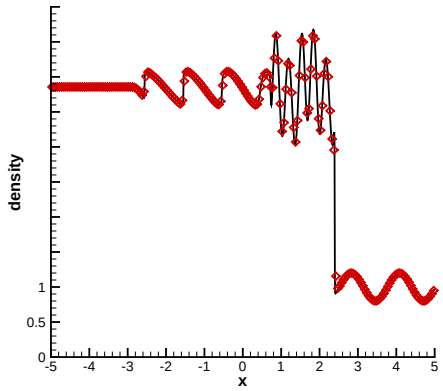
$$(\rho_M, v_M, P_M) = (1, 0, 0.01),$$

$$(\rho_R, v_R, P_R) = (1, 0, 100).$$

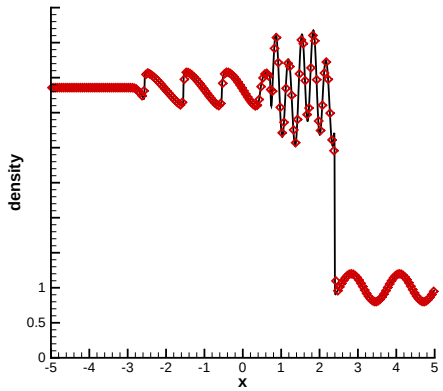
A reflecting boundary condition is applied to both ends. See [46][20].



(a) $k = 1$

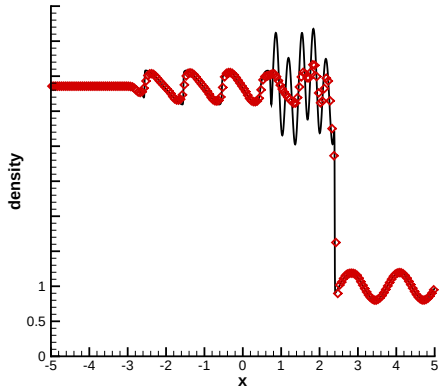


(b) $k = 2$

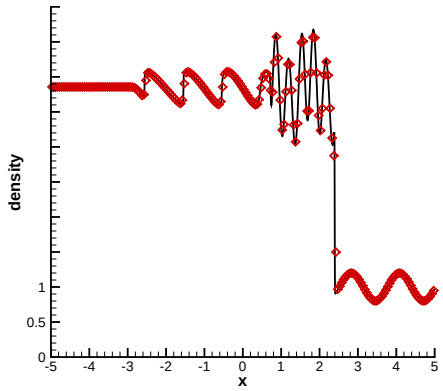


(c) $k = 3$

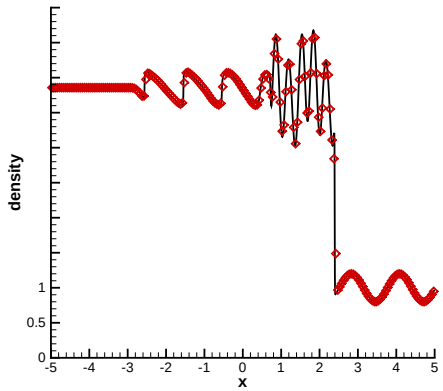
Figure 3.11: Shu-Osher shocktube problem at $t = 1.8$ using the P^k DG scheme with different k. $\varepsilon = 0.2 \cdot \Delta x$. $N = 200$.



(a) $k = 1$

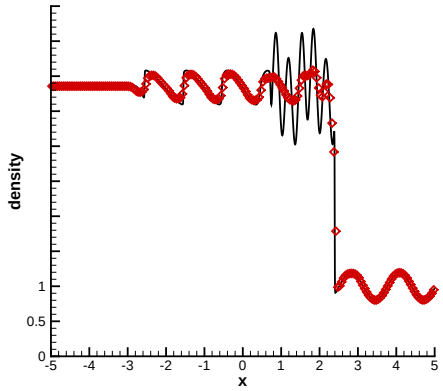


(b) $k = 2$

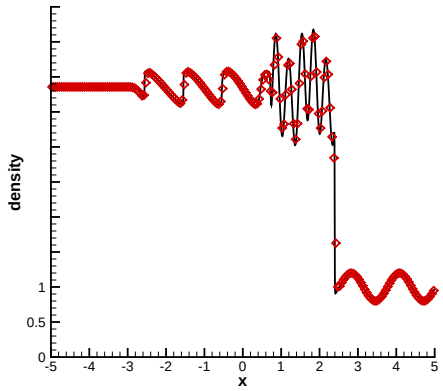


(c) $k = 3$

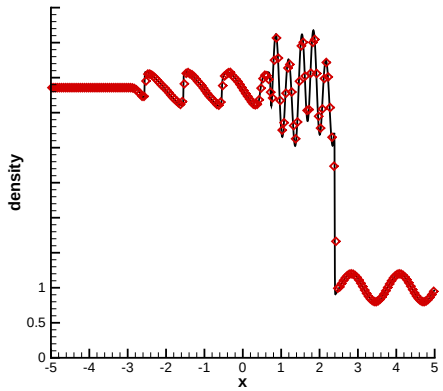
Figure 3.12: Shu-Osher shocktube problem at $t = 1.8$ using the P^k DG scheme with different k. $\varepsilon = 0.5 \cdot \Delta x$. $N = 200$.



(a) $k = 1$



(b) $k = 2$



(c) $k = 3$

Figure 3.13: Shu-Osher shocktube problem at $t = 1.8$ using the P^k DG scheme with different k. $\varepsilon = 1 \cdot \Delta x$. $N = 200$.

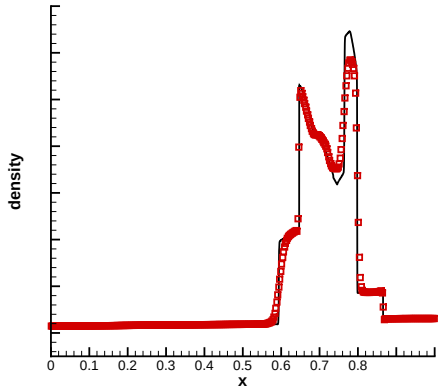
The results of density ρ at $t = 0.038$ are in Figures 3.14, 3.15 and 3.16 for $\varepsilon = 2.5\Delta x$, $5\Delta x$ and $10\Delta x$, respectively. The solid line is a converged solution computed by the fifth order finite difference WENO scheme with 16000 grid points, regarded as the "exact" solution. For this example, ε should be taken big enough (roughly $2.5\Delta x$ in our numerical tests) for the scheme to work, and we do not need positivity preserving limiters for all $k = 1$, 2 and 3. We see that the scheme again performs pretty well for this example, especially for higher order schemes. When ε is suitably chosen, our solution on the coarse mesh can closely match the reference solution without much smearing.

3.6.3 Euler system in two dimension

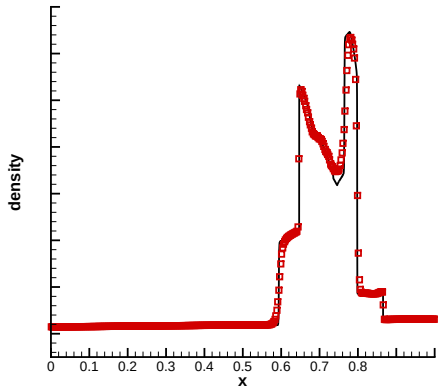
In this part we test our new artificial diffusion limiter on rectangular meshes solving two dimensional Euler equations

$$\mathbf{u}_t + \mathbf{f}(\mathbf{u})_x + \mathbf{g}(\mathbf{u})_y = 0 \tag{3.34}$$

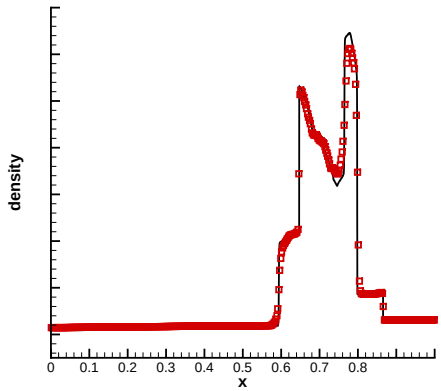
$$\mathbf{u} = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ E \end{pmatrix}, \qquad \mathbf{f}(\mathbf{u}) = \begin{pmatrix} \rho u \\ \rho u^2 + P \\ \rho uv \\ u(E+P) \end{pmatrix}, \qquad \mathbf{g}(\mathbf{u}) = \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + P \\ v(E+P) \end{pmatrix},$$



(a) $k = 1$

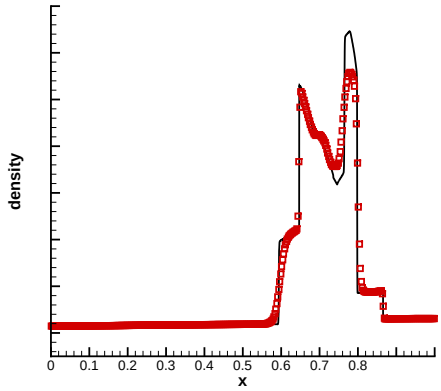


(b) $k = 2$

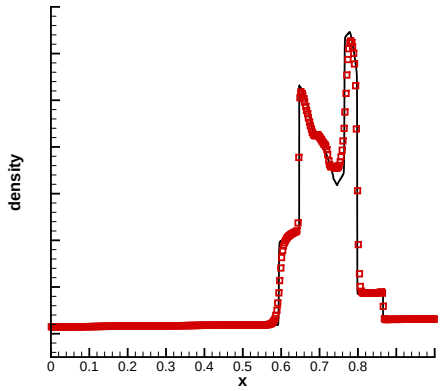


(c) $k = 3$

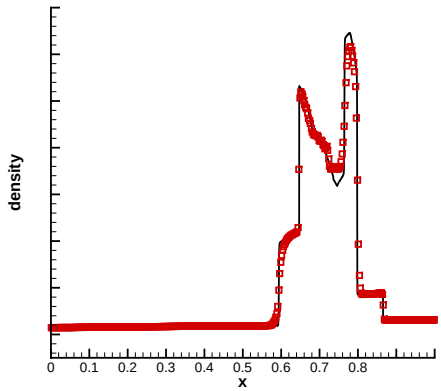
Figure 3.14: Blast wave problem at $t = 0.038$ using the P^k DG scheme with different k. $\varepsilon = 2.5 \cdot \Delta x$. $N = 400$.



(a) $k = 1$

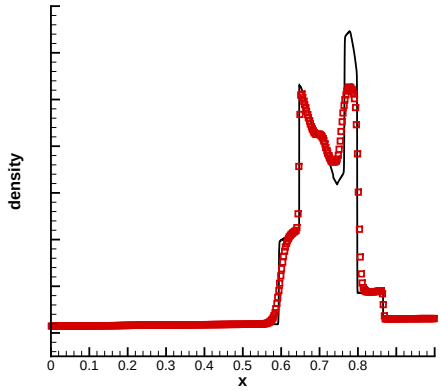


(b) $k = 2$

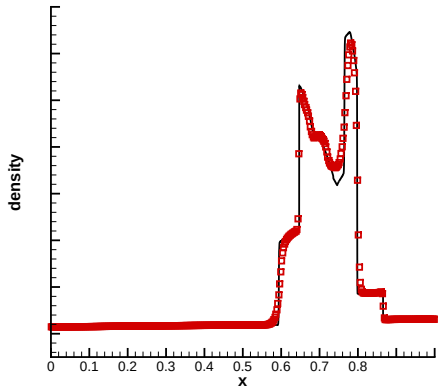


(c) $k = 3$

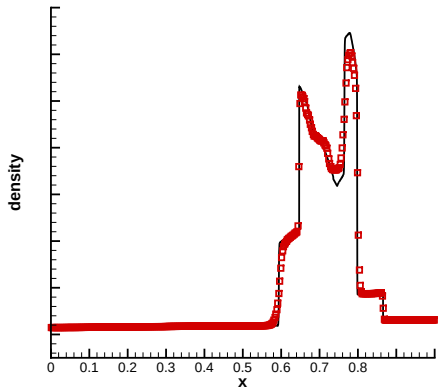
Figure 3.15: Blast wave problem at $t = 0.038$ using the P^k DG scheme with different k. $\varepsilon = 5 \cdot \Delta x$. $N = 400$.



(a) $k = 1$



(b) $k = 2$



(c) $k = 3$

Figure 3.16: Blast wave problem at $t = 0.038$ using the P^k DG scheme with different k. $\varepsilon = 10 \cdot \Delta x$. $N = 400$.

Table 3.10: Error for the 2D Euler equation with the initial condition $\rho(x,y,0) = 1 + 0.2\sin(x+y)$, $u(x,y,0) = 0.7$, $v(x,y,0) = 0.3$ and $P(x,y,0) = 1$ at $t = 1$ using the DG scheme with $N \times N$ cells.

N	P^1				P^2			
	L_1 error	order	L_∞ error	order	L_1 error	order	L_∞ error	order
20	2.04e-03		5.71e-03		9.00e-05		7.35e-04	
40	3.52e-04	2.54	1.74e-03	1.71	1.09e-05	3.05	1.05e-04	2.80
80	7.21e-05	2.29	4.74e-04	1.88	1.29e-06	3.08	1.39e-05	2.93
160	1.65e-05	2.14	1.22e-04	1.96	1.56e-07	3.05	1.76e-06	2.98

where ρ is the density, u is the velocity in the x-direction, v is the velocity in the y-direction, E is the total energy, p is the pressure

$$P = (\gamma - 1)(E - \frac{1}{2}\rho(u^2 + v^2)).$$

The speed of sound is given by $c = \sqrt{\gamma P/\rho}$. The eigenvalues of the Jacobian $\mathbf{f}'(\mathbf{u})$ are $u - c$, u and $u + c$, and the eigenvalues of the Jacobian $\mathbf{g}'(\mathbf{u})$ are $v - c$, v and $v + c$. $\gamma = 1.4$ is used in the computation.

Example 3.6.3.1. This example is to test the order of accuracy for the DG method with our artificial diffusion WENO limiter. The initial condition is set to be $\rho(x,y,0) = 1 + 0.2\sin(x+y)$, $u(x,y,0) = 0.7$, $v(x,y,0) = 0.3$, $P(x,y,0) = 1$ and the boundary conditions are periodic. The exactly solution is $\rho(x,y,t) = 1 + 0.2\sin(x+y-t)$, $u(x,y,t) = 0.7$, $v(x,y,t) = 0.3$ and $P(x,y,t) = 1$. We collect the L_1 and L_∞ errors at $t = 1$ in Tables 3.10 and 3.11. Again, we can see that our limiter maintains both the designed order of accuracy and the magnitude of the error of the original DG method.

Example 3.6.3.2. Consider the following idealized vortex problem [22][42] for the

Table 3.11: Error for the 2D Euler equation with the initial condition $\rho(x,y,0) = 1+0.2\sin(x+y)$, $u(x,y,0)=0.7$, $v(x,y,0)=0.3$ and $P(x,y,0)=1$ at $t=1$ using the DG scheme with the limiter. $\varepsilon = 0.1$. $N \times N$ cells.

N	P^1				P^2			
	L_1 error	order	L_∞ error	order	L_1 error	order	L_∞ error	order
20	2.33e-03		8.36e-03		8.98e-05		7.35e-04	
40	3.68e-04	2.66	2.01e-03	2.06	1.08e-05	3.06	1.05e-04	2.81
80	7.23e-05	2.35	4.72e-04	2.09	1.28e-06	3.08	1.38e-05	2.93
160	1.65e-05	2.13	1.24e-04	1.94	1.55e-07	3.05	1.75e-06	2.98

Euler equations in 2D: The mean flow is $\rho = 1$, $P = 1$ and $(u,v) = (1,1)$ (diagonal flow). We add to this mean flow, an isentropic vortex (perturbations in (u,v) and the temperature $T = \frac{P}{\rho}$, no perturbation in the entropy $S = \frac{P}{\rho^\gamma}$):

$$(\delta u, \delta v) = \frac{\epsilon}{2\pi} e^{0.5(1-r^2)}(-\bar{y}, \bar{x}), \tag{3.35}$$

$$\delta T = -\frac{(\gamma-1)\epsilon^2}{8\gamma\pi^2} e^{1-r^2}, \tag{3.36}$$

$$\delta S = 0, \tag{3.37}$$

where $(\bar{x}, \bar{y}) = (x-5, y-5)$, $r^2 = \bar{x}^2 + \bar{y}^2$, and the vortex strength $\epsilon = 5$. Since the mean flow is in the diagonal direction, the vortex movement is not aligned with the mesh direction.

The computational domain is taken as $[-5,15] \times [-5,15]$, extended periodically in both directions. It is clear that the exact solution of the Euler equation with the above initial and boundary conditions is just the passive convection of the vortex with the mean velocity. Our simulation is performed until $t = 2$. As the errors can be seen from Tables 3.12 and 3.13 for $\varepsilon = 0$ and 0.3, the order of accuracy and the magnitude of errors are maintained.

Example 3.6.3.3. We consider the double Mach reflection problem [46], which is

Table 3.12: Error for the 2D Euler system of the vortex evolution at $t = 2$ using the DG scheme with $N \times N$ cells.

N	P^1				P^2			
	L_1 error	order	L_∞ error	order	L_1 error	order	L_∞ error	order
20	2.11e-03		1.63e-01		5.44e-04		1.01e-01	
40	4.58e-04	2.21	4.05e-02	2.01	6.19e-05	3.14	9.99e-03	3.43
80	8.89e-05	2.37	1.28e-02	1.66	7.70e-06	3.01	1.28e-03	2.97
160	1.67e-05	2.41	2.91e-03	2.14	1.08e-06	2.84	1.78e-04	2.85

Table 3.13: Error for the 2D Euler system of the vortex evolution at $t = 2$ using the DG scheme with limiter. $\varepsilon = 0.3$. $N \times N$ cells.

N	P^1				P^2			
	L_1 error	order	L_∞ error	order	L_1 error	order	L_∞ error	order
20	3.20e-03		2.73e-01		8.01e-04		8.72e-02	
40	9.00e-04	1.83	8.03e-02	1.77	5.76e-05	3.80	9.34e-03	3.22
80	1.90e-04	2.24	2.29e-02	1.81	7.70e-06	2.90	1.21e-03	2.95
160	3.04e-05	2.65	5.02e-03	2.19	1.06e-06	2.86	1.78e-04	2.77

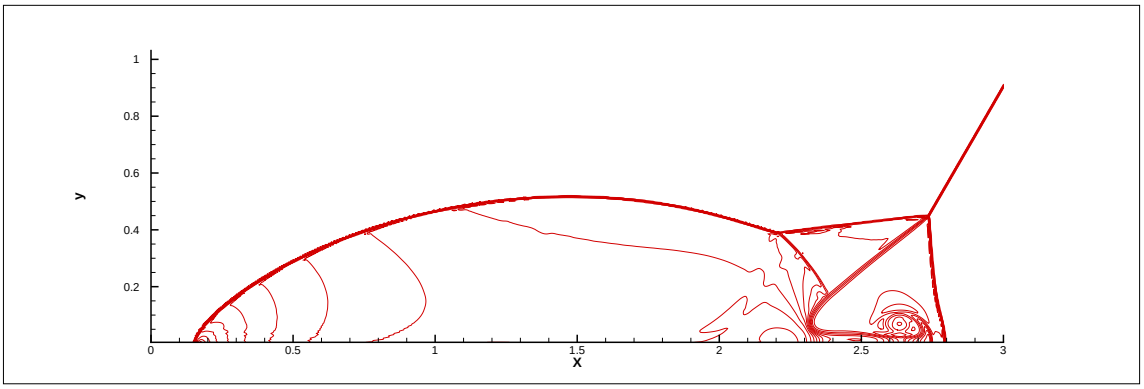
a standard test case for numerical schemes to capture strong shock wave and high resolution. A Mach 10 shock initially makes a 60 degree angle with a reflecting wall. The undisturbed air ahead of the shock has a density of 1.4 and a pressure of 1. We use rectangular elements for this problem. The computational domain for this problem is chosen to be $[0, 4] \times [0, 1]$. The reflecting wall lies at the bottom of the computational domain, starting from $x = 1/6$. Initially a right-moving Mach 10 shock is positioned at $x = 1/6$, $y = 0$ and makes a 60 degree angle with the x-axis. For the bottom boundary, the exact post-shock condition is imposed for the part from $x = 0$ to $x = 1/6$, and a reflective boundary condition is used for the rest. At the top boundary of our computational domain, the flow values are set to describe the exact motion of the Mach 10 shock. Inflow/outflow boundary conditions are used for the left and right boundaries.

This problem has fine details and complicated structure in the region. We com-

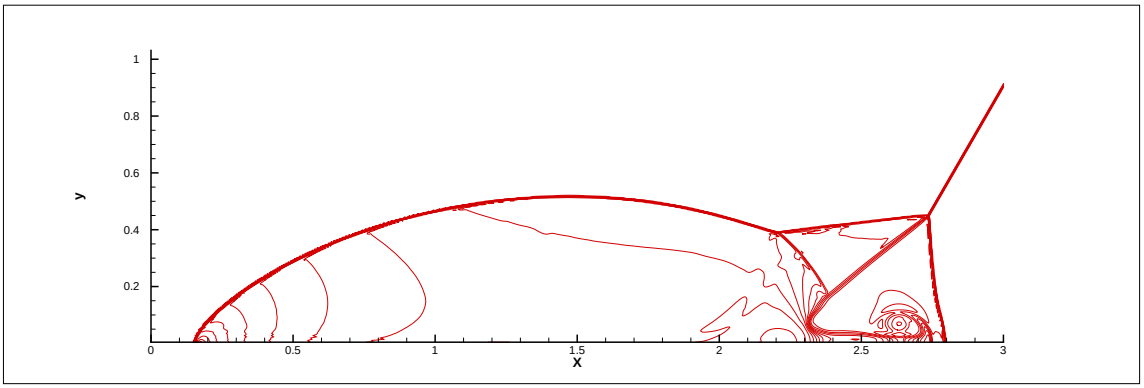
pute the solution up to $t = 0.2$ and only display part of the domain $[0, 3] \times [0, 1]$ in our figures. Three different uniform meshes, with 960×240, 1440×360 and 1920×480 cells, and several different values of ε are used in the numerical experiments. The density is plotted in Figure 3.17 for $k = 1$ and in Figures 3.18, 3.19, 3.20 for $k = 2$ with different mesh sizes. In each of them, schemes with different values of ε are used. We plot 29 contours equally distributed from $\rho = 1.3$ to 23 in all figures. One observes an increased resolution with an increasing k on the same mesh, and with finer mesh for the same k. Also, we can see clearly in this example that with the increase of ε, the solution becomes less oscillatory, especially seen from the several contours on the left part of the domain, and the small waving area in the right bottom triangular area. But we also observe in our numerical tests, when ε is taken to be very large, say 5, the solution around the right bottom triangular area becomes messy, which may be due to the numerical dispersion errors mentioned in the previous section.

3.7 Conclusion and Future Work

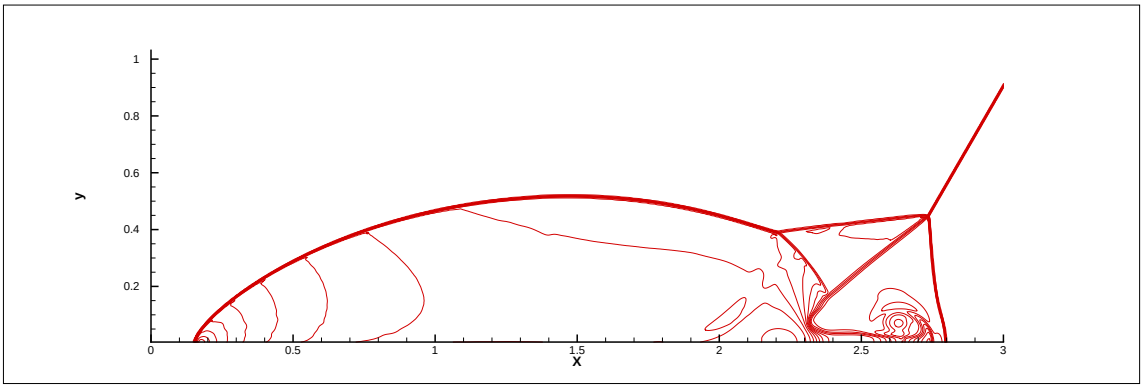
In this chapter, we have introduced a new limiting procedure for the DG method to solve hyperbolic conservation laws using the WENO methodology. The idea is to incorporate the limiter into the DG framework, by adding an extra term $-\varepsilon \int (u_x - (Pu)_x) v_x \, dx$ to the RHS of the DG scheme, where Pu is the reconstructed polynomial on the current cell based on the WENO methodology. This added term proves to be,



(a) $\varepsilon = 0.01$

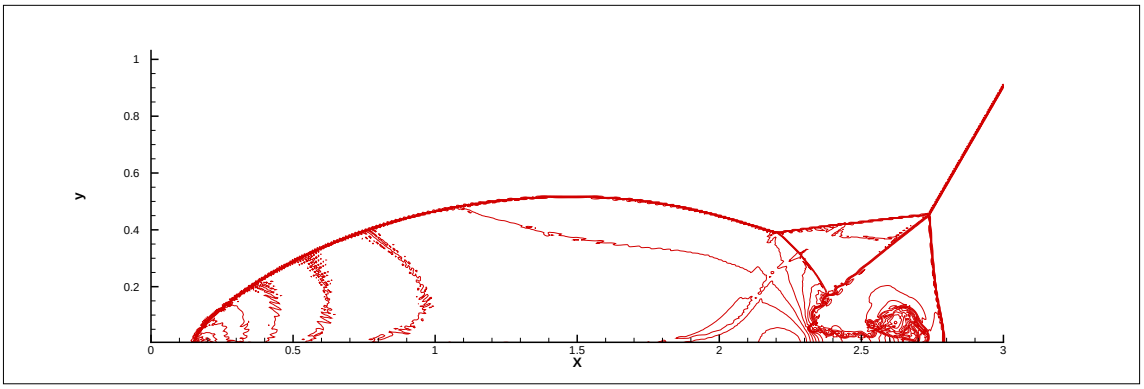


(b) $\varepsilon = 0.1$

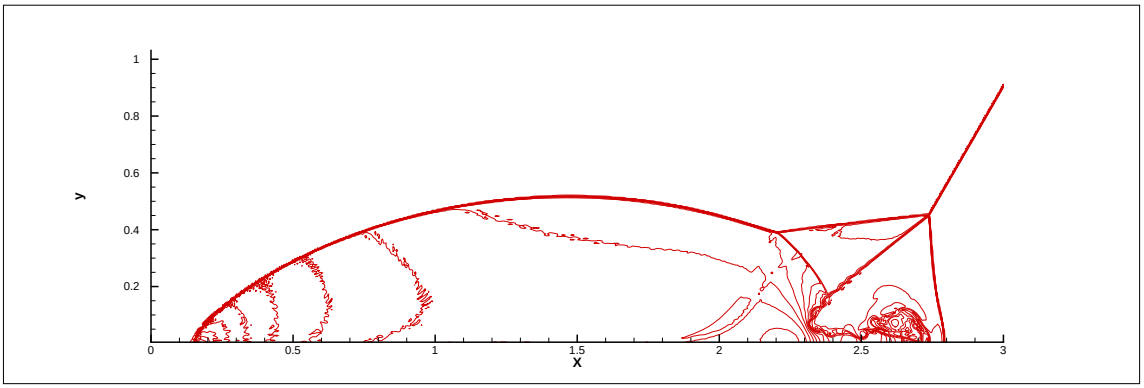


(c) $\varepsilon = 1$

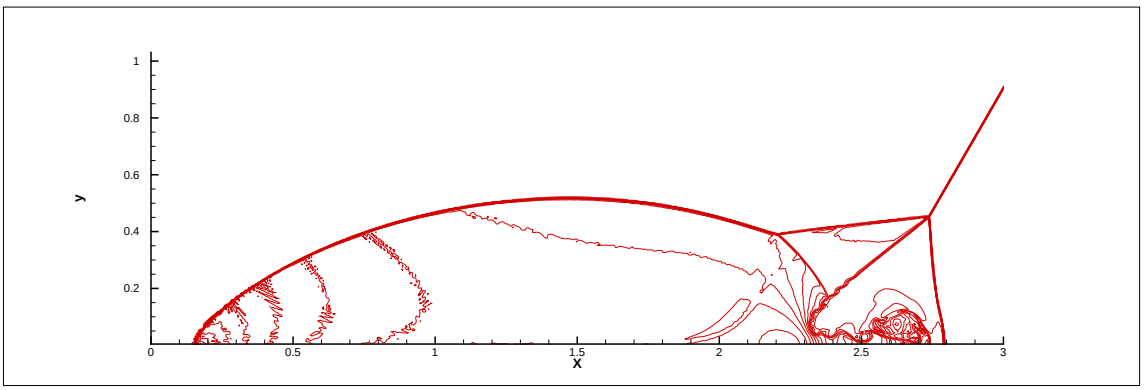
Figure 3.17: Double Mach reflection problem solution using the P^1 DG sheme with limiter applied for all cells. $\Delta x = \Delta y = 1/240$. Twenty-nine equally spaced density contours from 1.3 to 23.



(a) $\varepsilon = 0.1$

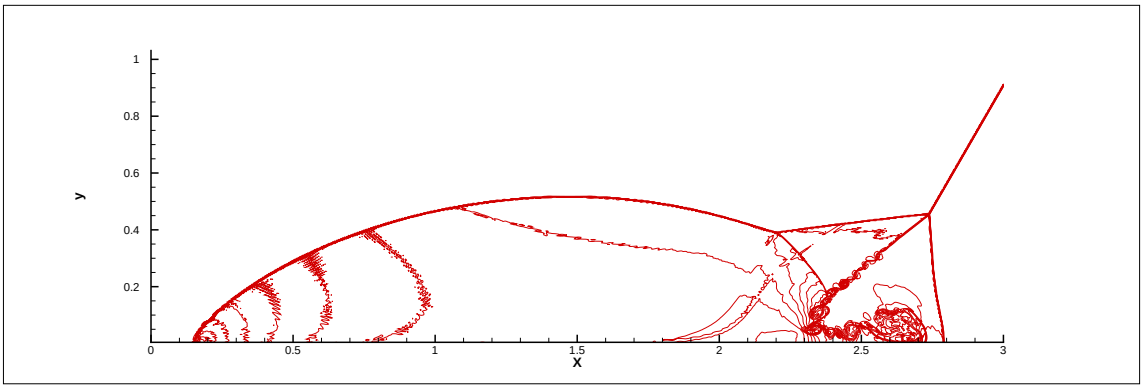


(b) $\varepsilon = 0.5$

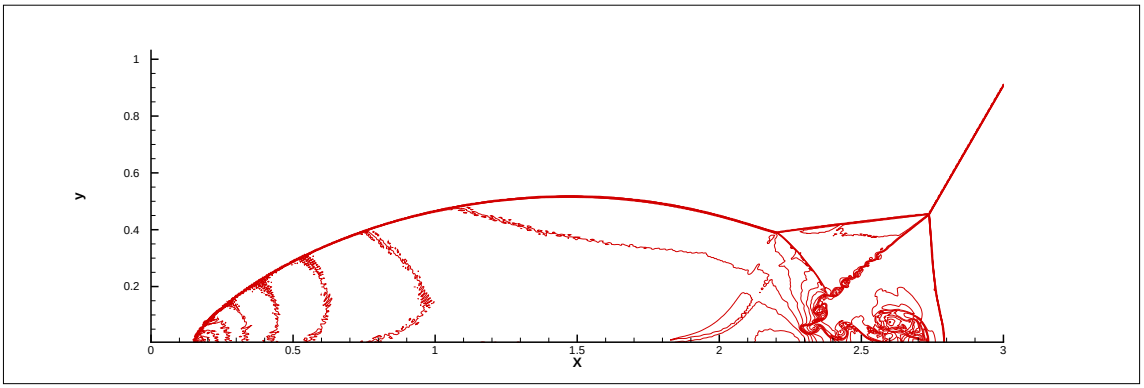


(c) $\varepsilon = 1$

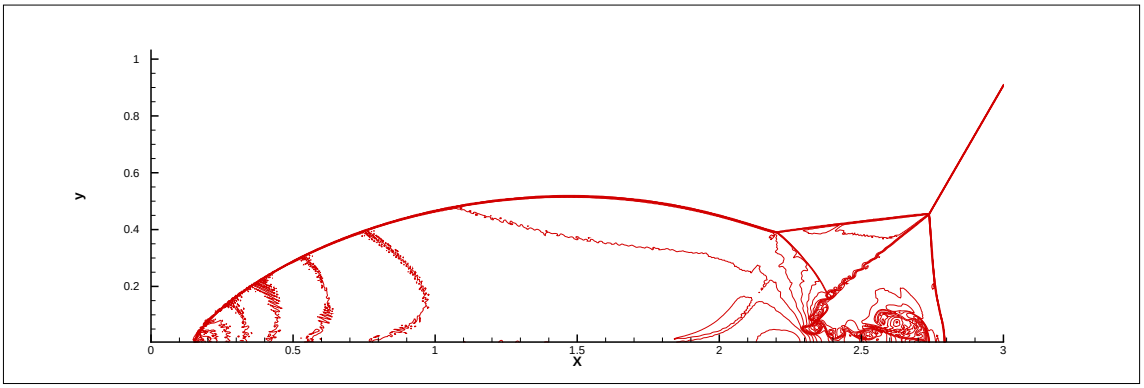
Figure 3.18: Double Mach reflection problem solution using the P^2 DG sheme with limiter applied for all cells. $\Delta x = \Delta y = 1/240$. Twenty-nine equally spaced density contours from 1.3 to 23.



(a) $\varepsilon = 0.1$

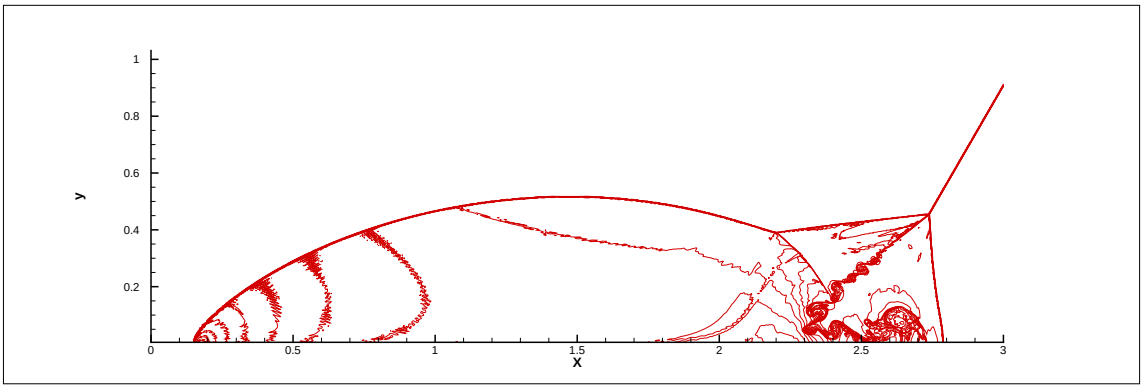


(b) $\varepsilon = 0.5$

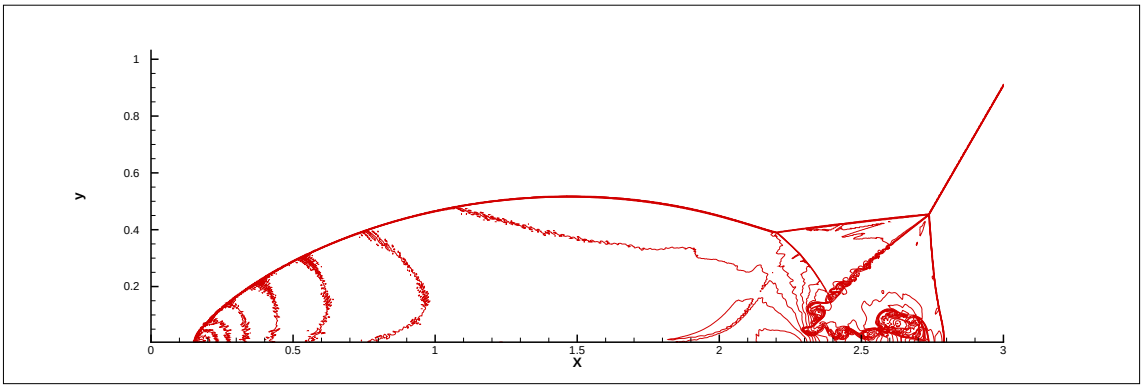


(c) $\varepsilon = 1$

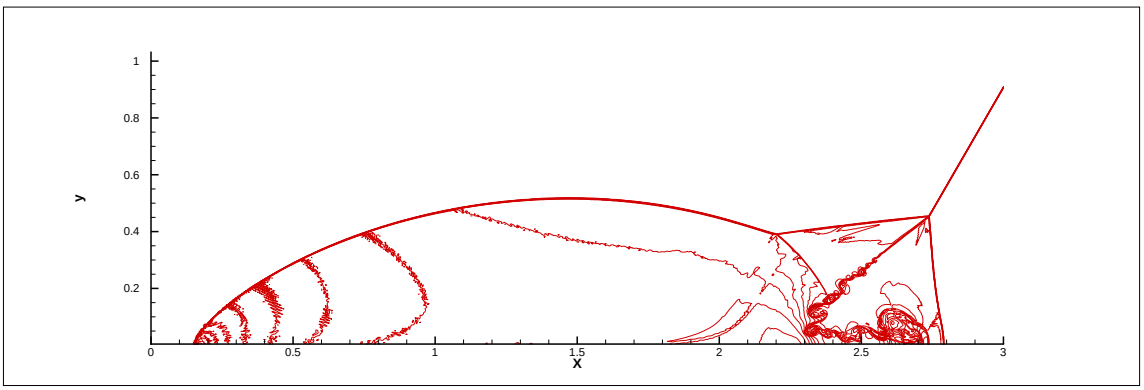
Figure 3.19: Double Mach reflection problem solution using the P^2 DG sheme with limiter applied for all cells. $\Delta x = \Delta y = 1/360$. Twenty-nine equally spaced density contours from 1.3 to 23.



(a) $\varepsilon = 0.1$



(b) $\varepsilon = 0.5$



(c) $\varepsilon = 1$

Figure 3.20: Double Mach reflection problem solution using the P^2 DG sheme with limiter applied for all cells. $\Delta x = \Delta y = 1/480$. Twenty-nine equally spaced density contours from 1.3 to 23.

or contain, numerical viscosity, when using linear weights for reconstruction in our finite difference analysis. The magnitude of the artificial diffusion depends on the difference between u and the reconstructed polynomial Pu, which is to our desire, large around shocks and small in smooth regions. Extensive numerical examples have shown that the new limiter can obtain both uniform high order accuracy and sharp, non-oscillatory shock transitions. Compared with previously WENO limiters, our method adds the numerical viscosity directly to the weak formulation, and has a semi-discrete form. It also enjoys the flexibility of being able to control the strength of the limiter by changing the parameter ε. When suitably chosen, the scheme can achieve its optimal performance. Also our method is not very sensitive to ε, which allows us to have a wider range for it to be properly chosen.

In the future work, we will test our limiters on steady state problems, and we expect the scheme to work well when converging to a steady state. Besides, the choice of ε in our approach can be subtle, we could investigate non-uniform parameters depending on the smoothness indicator of the cells to capture the shock.

Bibliography

[1] O. Y. Ababneh, R. Ahmad, and E. S. Ismail. Design of new diagonally implicit runge-kutta methods for stiff problems. *Applied Mathematical Sciences*, 3:2241–2253, 2009.

[2] R. Alexander. Diagonally implicit runge-kutta methods for stiff ordinary differential equations. *SIAM J Numer Anal*, 14:1006–1021, 1977.

[3] D.S. Balsara and C.-W. Shu. Monotonicity preserving weighted essentially non-oscillatory schemes with increasingly high order of accuracy. *Journal of Computational Physics*, 160:405–452, 2000.

[4] R. Biswas, K.D. Devine, and J. Flaherty. Parallel, adaptive finite element methods for conservation laws. *Applied Numerical Mathematics*, 14:255–283, 1994.

[5] A. Burbeau, P. Sagaut, and Ch.H. Bruneau. A problem-independent limiter for high-order runge-kutta discontinuous galerkin methods. *Journal of Computational Physics*, 169:111–150, 2001.

[6] K. Burrage, J. C. Butcher, and F. H. Chipman. An implementation of singly implicit runge-kutta methods. *BIT*, 20:326–340, 1980.

[7] A. Christlieb, Y. Liu, Q. Tang, and Z. Xu. High order parametrized maximum-principle-preserving and positivity-preserving WENO schemes on unstructured meshes. *Journal of Computational Physics*, 281:334–351, 2015.

[8] H. Claus. Singly-implicit runge-kutta methods for retarded and ordinary differential equations. *Computing*, 43:209–222, 1990.

[9] B. Cockburn, S. Hou, and C.-W. Shu. The runge-kutta local projection discontinuous galerkin finite element method for conservation laws IV: the multidimensional case. *Mathematics of Computation*, 54:545–581, 1990.

[10] B. Cockburn, G. E. Karniadakis, and C.-W. Shu. *The development of discontinuous Galerkin methods*. Springer, 2000.

[11] B. Cockburn, S.-Y. Lin, and C.-W. Shu. TVB runge-kutta local projection discontinuous galerkin finite element method for conservation laws III: one dimensional systems. *Journal of Computational Physics*, 84:90–113, 1989.

[12] B. Cockburn and C.-W. Shu. TVB runge-kutta local projection discontinuous galerkin finite element method for conservation laws II: general framework. *Mathematics of Computation*, 52:411–435, 1989.

[13] B. Cockburn and C.-W. Shu. The runge-kutta local projection p_1 discontinuous galerkin finite element method for scalar conservation laws. *Mathematical Modelling and Numerical Analysis*, 25:337–361, 1991.

[14] B. Cockburn and C.-W. Shu. The runge-kutta discontinuous galerkin method for conservation laws V: multidimensional systems. *Journal of Computational Physics*, 141:199–224, 1998.

[15] B. Cockburn and C.-W. Shu. Runge–Kutta discontinuous Galerkin methods for convection-dominated problems. *Journal of Scientific Computing*, 16(3):173–261, September 2001.

[16] G. B. Dantzig. Linear programming and extensions. *Princeton, NJ: Princeton University Press*, 1963.

[17] O. Friedrich. Weighted essentially non-oscillatory schemes for the interpolation of mean values on unstructured grids. 144:194–212, 1998.

[18] S. Gottlieb, D.I. Ketcheson, and C.-W. Shu. High order strong stability preserving time discretizations. *Journal of Scientific Computing*, 38:251–289, 2009.

[19] A. Harten. High resolution schemes for hyperbolic conservation laws. *Journal of Computational Physics*, 49:357–393, 1983.

[20] A. Harten, B. Engquist, S. Osher, and S.R. Chakravarthy. Uniformly high order accurate essentially non-oscillatory schemes, III. *Journal of Computational Physics*, 71:231–303, 1987.

[21] A. Hiltebrand and S. Mishra. Entropy stable shock capturing space-time discontinuous galerkin schemes for systems of conservation laws. *Numerische Mathematik*, 126:103–151, 2014.

[22] C. Hu and C.-W. Shu. Weighted essentially non-oscillatory schemes on triangular meshes. *Journal of Computational Physics*, 150:97–127, 1999.

[23] G.-S. Jiang and C.-W. Shu. Efficient Implementation of Weighted ENO Schemes. *Journal of Computational Physics*, 126(1):202–228, June 1996.

[24] Y. Jiang and Z. Xu. Parametrized maximum principle preserving limiter for finite difference WENO schemes solving convection-dominated diffusion equations. *SIAM Journal on Scientific Computing*, 35(6):A2524–A2553, 2013.

[25] P. D. Lax. Weak solutions of nonlinear hyperbolic equations and their numerical computation. *Communications on Pure and Applied Mathematics*, 7:159–193, 1954.

[26] Randall J. LeVeque. *Numerical Methods for Conservation Laws*. Lectures in Mathematics, ETH Zürich. Birkhäuser Basel, 1992.

[27] D. Levy, G. Puppo, and G. Russo. Central WENO schemes for hyperbolic systems of conservation laws. *Mathematical Modelling and Numerical Analysis*, 33:547–571, 2001.

[28] C. Liang and Z. Xu. Parametrized maximum principle preserving flux limiters for high order schemes solving multi-dimensional scalar hyperbolic conservation laws. *Journal of Scientific Computing*, 58:41–60, 2014.

[29] W. Liniger and R. A. Willoughby. Efficient integration methods for stiff systems of ordinary differential equations. *SIAM J Numer Anal*, 7:47–66, 1970.

[30] X. Liu, S. Osher, and T. Chen. Weighted essentially non-oscillatory schemes. *Journal of Computational Physics*, 115:200–212, 1994.

[31] Y. Maday and E. Tadmor. Analysis of the spectral vanishing viscosity method for periodic conservation laws. *SIAM Journal of Numerical Analysis*, 26:854–870, 1989.

[32] J. Nocedal and S. J. Wright. Numerical optimization. *New York: Springer-Verlag*, 1999.

[33] S. P. Norsset. Semi-explicit runge-kutta methods. *Report No. 6, Department of Maths, University of Troudneim, Norway*, 1974.

[34] J. Qiu and C.-W. Shu. On the construction, comparison, and local characteristic decomposition for high-order central WENO schemes. *Journal of Computational Physics*, 183:187–209, 2002.

[35] J. Qiu and C.-W. Shu. Hermite WENO schemes and their application as limiters for runge-kutta discontinuous galerkin method: one dimensional case. *Journal of Computational Physics*, 193:115–135, 2003.

[36] J. Qiu and C.-W. Shu. Hermite WENO schemes and their application as limiters for runge-kutta discontinuous galerkin method II: two dimensional case. *Computers and Fluids*, 34:642–663, 2005.

[37] J. Qiu and C.-W. Shu. Runge-kutta discontinuous galerkin method using WENO limiters. *SIAM Journal on Scientific Computing*, 26:907–929, 2005.

[38] W.H. Reed and T.R. Hill. Triangular mesh methods for the neutron transport equation. *Tech. Report*, (LA-UR-73-479), 1973.

[39] P.L. Roe. Approximate riemann solvers, parameter vectors, and difference schemes. *Journal of computational Physics*, 135(2):250–258, 1997.

[40] J. Shi, C. Hu, and C.-W. Shu. A technique of treating negative weights in WENO schemes. *Journal of Computational Physics*, 175:108–127, 2002.

[41] C.-W. Shu. TVB uniformly high-order schemes for conservation laws. *Mathematics of Computation*, 49(179):105–121, 1987.

[42] C.-W. Shu. Essentially non-oscillatory and weighted essentially non-oscillatory schemes for hyperbolic conservation laws. In *Advanced numerical approximation of nonlinear hyperbolic equations*, pages 325–432. Springer Berlin Heidelberg, 1998.

[43] C.-W. Shu and S. Osher. Efficient implementation of essentially non-oscillatory shock-capturing schemes. *Journal of Computational Physics*, 77:439–471, 1988.

[44] L. M. Skvortsov. Diagonally implicit runge-kutta methods for stiff problems. *Computational Mathematics and Mathematical Physics*, 46:2110–2123, 2006.

[45] G. Sod. A survey of several finite difference methods for systems of nonlinear hyperbolic conservation laws. *Journal of Computational Physics*, 27:1–31, 1978.

[46] P. Woodward and P. Colella. The numerical simulation of two-dimensional fluid flow with strong shocks. *Journal of Computational Physics*, 54:115–173, 1984.

[47] T. Xiong, J.-M. Qiu, and Z. Xu. parametrized maximum principle preserving flux limiter for finite difference RK-WENO schemes with applications in incompressible flows. *Journal of Computational Physics*, 252:310–331, 2013.

[48] T. Xiong, J.-M. Qiu, and Z. Xu. A parametrized maximum principle preserving flux limiter for finite difference RK-WENO schemes with applications in incompressible flows. *Journal of Computational Physics*, 252:310–331, 2013.

[49] T. Xiong, J.-M. Qiu, and Z. Xu. Parametrized positivity preserving flux limiters for the high order finite difference WENO scheme solving compressible euler equations. *Journal of Scientific Computing*, 67(3):1066–1088, 2016.

[50] Z. Xu. Parametrized maximum principle preserving flux limiters for high order scheme solving hyperbolic conservation laws: one-dimensional scalar problem. *Mathematics of Computation*, 83:2213–2238, 2014.

[51] P. Yang, T. Xiong, J.-M. Qiu, and Z. Xu. High order maximum principle preserving finite volume method for convection dominated problems. *Journal of Scientific Computing*, 67(2):795–820, 2016.

[52] M. Zhang and C.-W. Shu. An analysis of and a comparison between the discontinuous galerkin and the spectral finite volume methods. *Computers and Fluids*, 34:581–592, 2005.

[53] X. Zhang. On positivity-preserving high order discontinuous Galerkin schemes for compressible Navier–Stokes equations. *Journal of Computational Physics*, 328:301–343, January 2017.

[54] X. Zhang and C.-W. Shu. On maximum-principle-satisfying high order schemes for scalar conservation laws. *Journal of Computational Physics*, 229:3091–3120, 2010.

[55] X. Zhang and C.-W. Shu. On positivity-preserving high order discontinuous Galerkin schemes for compressible Euler equations on rectangular meshes. *Journal of Computational Physics*, 229(23):8918–8934, November 2010.

[56] X. Zhang and C.-W. Shu. Maximum-principle-satisfying and positivity-preserving high-order schemes for conservation laws: Survey and new developments. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 467(2134):2752–2776, October 2011.

[57] X. Zhang and C.-W. Shu. Positivity-preserving high order discontinuous Galerkin schemes for compressible Euler equations with source terms. *Journal of Computational Physics*, 230(4):1238–1248, February 2011.

[58] X. Zhang and C.-W. Shu. Positivity-preserving high order finite difference WENO schemes for compressible Euler equations. *Journal of Computational Physics*, 231(5):2245–2258, March 2012.

[59] X. Zhong and C.-W. Shu. A simple weighted essentially nonoscillatory limiter for runge-kutta discontinuous galerkin methods. *Journal of Computational Physics*, 232:397–415, 2012.

[60] J. Zhu, J. Qiu, C.-W. Shu, and M. Dumbser. Runge-kutta discontinuous galerkin method using WENO limiters II: unstructured meshes. *Journal of Computational Physics*, 227:4330–4353, 2008.

[61] J. Zhu, X. Zhong, C.-W. Shu, and J. Qiu. Runge-kutta discontinuous galerkin method using a new type of WENO limiters on unstructured meshes. *Journal of Computational Physics*, 248:200–220, 2013.