2019

High-order hybrid Roe-WENO schemes for interface advective flux reconstruction

Hayden Arceneaux

HIGH-ORDER HYBRID ROE-WENO SCHEMES FOR INTERFACE ADVECTIVE FLUX RECONSTRUCTION

by

HAYDEN ARCENEAUX

A THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Engineering
in
The Department of Mechanical and Aerospace Engineering
to
The School of Graduate Studies
of
The University of Alabama in Huntsville

HUNTSVILLE, ALABAMA

2019

In presenting this thesis in partial fulfillment of the requirements for a master's degree from The University of Alabama in Huntsville, I agree that the Library of this University shall make it freely available for inspection. I further agree that permission for extensive copying for scholarly purposes may be granted by my advisor or, in his/her absence, by the Chair of the Department or the Dean of the School of Graduate Studies. It is also understood that due recognition shall be given to me and to The University of Alabama in Huntsville in any scholarly use which may be made of any material in this thesis.

Hayden Arceneaux — 6/26/19 (date)

THESIS APPROVAL FORM

Submitted by Hayden Arceneaux in partial fulfillment of the requirements for the degree of Master of Science in Engineering in Mechanical Engineering and accepted on behalf of the Faculty of the School of Graduate Studies by the thesis committee.

We, the undersigned members of the Graduate Faculty of The University of Alabama in Huntsville, certify that we have advised and/or supervised the candidate of the work described in this thesis. We further certify that we have reviewed the thesis manuscript and approve it in partial fulfillment of the requirements for the degree of Master of Science in Engineering in Mechanical Engineering.

Sarma Rani	6/26/2019	Committee Chair
Dr. Sarma Rani	(Date)	
	6/26/19	
Dr. Kader Frendi	(Date)	
	6/26/19	
Dr. Sivaguru Ravindran	(Date)	
	7/1/19	Department Chair
Dr. Keith Hollingsworth	(Date)	
	07/09/19	College Dean
Dr. Shankar Mahalingam	(Date)	
	7/10/19	Graduate Dean
Dr. David Berkowitz	(Date)	

ABSTRACT

School of Graduate Studies
The University of Alabama in Huntsville

Degree Masters of Science in Engineering College/Dept. Engineering/Mechanical and Aerospace Engineering

Name of Candidate Hayden Arceneaux

Title High-Order Hybrid Roe-WENO Schemes for Interface Advective Flux Reconstruction

Numerical solution of partial differential equations using the finite volume method (FVM) entails calculating the fluxes of conservative variables at the cell faces. The fluxes are computed through the "reconstruction" of cell-centered variables to the cell face(s). "Reconstruction" is performed using variables and their gradients stored at the cell center located to the left and to the right of an interface, leading to two distinct states of the conservative variables at a face. The resulting discontinuity in interface flux presents a well-known conundrum in the FVM, particularly when solving hyperbolic governing equations. For the non-linear, hyperbolic Euler equations, Godunov recognized that the discontinuity in the left and right reconstructed states is qualitatively similar to the Riemann's initial-value problem that involves the advection of a scalar discontinuity. Godunov developed an "exact" Riemann solver that enables the calculation of the interface fluxes exactly. The exact solver, however, is computationally expensive and inefficient leading to the development of a large variety of "approximate" Riemann solvers. The best-known "approximate" Riemann solvers are the Roe's scheme, which is spatially first-order accurate, and the Roe-MUSCL scheme, which is a second-order extension of the former. *Accordingly, the*

goal of this thesis is to develop a high-order flux calculation scheme that retains the basic structure of the Roe's interface flux.

Roe's interface flux consists of two terms – a central-differenced (CD) flux, and a diffusive flux that ensures the solution remains monotonic. In this thesis, the high-order scheme is achieved by augmenting the order of accuracy of both these terms. For the CD flux term, two approaches are considered. The first involves replacing the second-order CD flux with a fourth- or sixth-order CD flux obtained from the cell-centered quantities located on either side of the interface. The second involves retaining the original form, but increasing the order of accuracy of the left and right reconstructed states that are inputs to the flux. The latter approach is also adopted for the diffusive flux. Third- and fifth-order Weighted Essentially Non-Oscillatory (WENO) schemes are used to reconstruct the left and right states at a cell face, so that the overall flux scheme may be referred to as a hybrid Roe-WENO scheme. The developed higher-order scheme is applied to a number of canonical 1-D and 2-D scalar advection cases. It is seen that a higher-order reconstruction of the states at a cell face performs better than the fourth- or sixth-order central differencing.

Abstract Approval: Committee Chair

Dr. Sarma Rani

Department Chair

Dr. Keith Hollingsworth

Graduate Dean

Dr. David Berkowitz

ACKNOWLEDGMENTS

I would like to thank my advisor, Dr. Sarma Rani, for his guidance and wisdom. He was always available when I had a question and encouraged me when I was lost.

I would like to thank Les Hall and the McLaurin Aerospace family, without whom I would not have been given the opportunity to study shock capturing schemes. I am also grateful to Dr. Eppard and Dr. Applebaum for always helping when I had a question.

I am very grateful to Tim and Joy Suttles for their help reviewing my thesis. They have supported me since I began my research. I also appreciate them taking me out for a game of golf when I was in need for a break.

I am very grateful to my Mom Colleen, Dad Howard, and Sister Hayley who always believed in me even when I lost faith in myself. Without your love and support, I would not have been able to achieve my dreams.

Finally, I would like to thank Liz for supporting me. She always supported me even when it meant long nights stuck in my office or weekends where I could not spend time with her.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

TABLE PAGE

To my Father, Howard S. Arceneaux

CHAPTER 1

INTRODUCTION

Numerical schemes with increased accuracy are of high importance. Higher-order accurate schemes have better resolution of flow features than lower-order schemes. Higher-order schemes also have the advantage of having the same resolution as lower-order schemes on a coarser grid.

1.1 Introduction to Shock Capturing Schemes

Modern engineering approaches to fluid dynamics are relying less on experimentation and more on computational fluid dynamics (CFD). The accuracy of a CFD simulation is heavily influenced by the order of accuracy of the numerical scheme. Lower-order schemes have increased numerical errors compared to higher-order schemes. The simplest way to reduce numerical error in lower-order schemes is to increase the number of cells. However, this comes with additional computational cost. It is often more computationally efficient to use a higher-order scheme than to increase the number of cells with a lower-order scheme.

Currently, there are two standard methods of increasing the accuracy of a simulation: 1) higher-order shock capturing schemes or 2) shock-fitting algorithms.

Shock-capturing schemes involve the discretization of the governing partial differential equations. Shock-fitting schemes increase the mesh density in regions where a shock is detected. The content of this thesis focuses on shock-capturing schemes and not on shock-fitting schemes. More information on shock-fitting schemes can be in [1], [2], and [3].

Godunov was the first person to recognize that the discontinuous cell face is similar to the Riemann problem [4]. Godunov's method solves the Riemann problem at cell interfaces. Constant piecewise cell-centered data is extrapolated to the cell face, leading to a discontinuity similar to the one-dimensional shock tube. The analytical solution for the one-dimensional shock tube is then performed to solve for the flux. For this reason, Godunov's scheme is known as an "exact" Riemann solver [5]. Due to the piecewise constant data assumption, Godunov's scheme is only first-order accurate.

Godunov's Riemann solver relies on the analytical solution of the Riemann problem. However, the analytical solution of the Riemann problem requires a costly iteration to find the pressure [6]. In an effort to reduce computational cost, approximate Riemann solvers were developed. Approximate Riemann solvers approximate the analytical solution of the Riemann problem. These schemes are designed to avoid the costly pressure iteration improving run times.

The most popular approximate Riemann solver is Roe's Riemann solver. Roe's scheme was based on a linearized Jacobian solved using Roe-averaged states. The scheme is based on left- and right-moving waves. These waves are averaged by the left- and right-cell face to produce Roe's flux. Roe's scheme is a flux-difference scheme. Another popular approximate Riemann solver is the Harten Lax van Leer (HLL)

solver. The HLL scheme assumes two waves from the cell center and estimates the wave strength. The HLL scheme was modified to account for the contact surface giving the Harten Lax van Leer Contact (HLLC) scheme. The flux vector splitting technique [7] discretizes the equations based on the direction of information propagation.

Roe's original method has been modified throughout the years to improve the scheme. In Roe's original paper [8], he pointed out an issue with expansion waves causing a shock that satisfies the Rankine-Hugoniot condition. The expansion shock is non-physical and violates the entropy condition. Harten [9] fixed this by adding numerical dissipation in the region where expansion shocks would be present. In [10], van Leer replaces the piecewise-constant slope of the original Godunov scheme with a linear slope that increased the order of accuracy to second order.

The Monotonic Upwind Scheme for Conservation Laws (MUSCL) scheme is used in most commercial CFD packages. The MUSCL scheme is relatively simple to implement on both structured and unstructured grids. Instead of using the piecewise-constant left- and right-states of Godunov, van Leer used a piecewise-linear reconstruction. He also introduced flux limiters to keep the scheme Total Variation Diminishing (TVD) in regions of strong gradients. The implementation of the MUSCL scheme for unstructured grids is shown in [5] and [11]. The MUSCL scheme's order of accuracy can be increased to third order on unstructured grids as shown in [12].

Harten, Engquist, Osher, and Chakravarthy [13] extended this idea to have smooth stencils that were higher-order accurate by ignoring stencils that go through regions of high gradients. The Essentially Non-Oscillatory (ENO) scheme was further

developed by Liu, Osher, and Chan assigning weights to the stencils to avoid costly "if" statements [14]. A convex combination of these weights results in the Weighted Essentially Non-Oscillatory (WENO) scheme. WENO is more robust than the ENO scheme as it avoids a costly "if" statement. Extensions to n^{th}-order accuracy can be achieved easily in WENO schemes.

Improvements to the WENO scheme have been made using characteristic reconstruction [15]. A compact Characteristic Reconstruction WENO (CRWENO) scheme has also been developed to have spectral-like accuracy [16]. One disadvantage of the WENO scheme is that it has high, computational cost on unstructured grids. The WENO scheme was designed to be used with orthogonal grids. Extension to unstructured grids is not as simple as the MUSCL scheme. Some examples of unstructured WENO schemes are in [17] and [18]. Because of the complexity of extending the WENO scheme to unstructured grids, it is not commonly found in commercial CFD software.

1.2 Motivation for Higher-Order Shock-Capturing Schemes

The goal of this thesis is to develop a high-order flux calculation that retains the basic structure of Roe's interface flux. Traditionally, the Roe scheme is made higher order by using reconstruction of the cell-face values. The CD and diffusion terms in Roe's scheme are left unmodified. A high-order flux is evaluated by utilizing a hybrid Roe-WENO scheme.

For the CD-flux term, two approaches are used. It can be shown that the CD flux is identical to a second-order central differencing about the cell face. For

the first approach, a fourth- or sixth-order CD replaces the CD term. For the second approach, the reconstructed values about the cell face calculates the CD. The diffusion term uses reconstruction to obtain higher-order accuracy. The third- and fifth-order WENO scheme is chosen for the reconstruction procedure. Thus, a new hybrid Roe-WENO scheme is developed. The new scheme is applied to the 1- and 2-D linear advection equation.

1.3 Approach

Roe's flux consists of two terms - a CD flux and a diffusive flux that allows for the solution to remain monotonic. In this thesis, a high-order scheme is achieved by increasing the order of accuracy of both of these terms. The first method replaces the second-order CD flux with a fourth- and sixth-order CD flux obtained from the cell-centered quantities for each side of the cell face. The second method retains Roe's original form, but increases the accuracy of the left- and right-reconstructed states. The diffusive flux also increases the accuracy of the reconstructed states on the left- and right-cell face.

Figure 1.1 shows a diagram of the different options of creating a high-order scheme. Roe's flux is constructed by two terms: a central difference term, and a diffusive term. The CD term can be constructed by two methods: using the reconstructed values at the left- and right-cell face, or using a higher-order central differencing. The reconstruction procedures considered are the upwind, MUSCL, WENO3, and WENO5 schemes. The higher-order central differencing uses a second-, fourth-, and sixth-order CD scheme. Only one method is available to make the diffusive term

higher order and that is reconstruction. The same reconstruction schemes used for the CD term are used for the diffusion term. Combining the higher-order central differencing for the CD term with WENO reconstruction for the diffusion term creates a new hybrid Roe-WENO scheme.

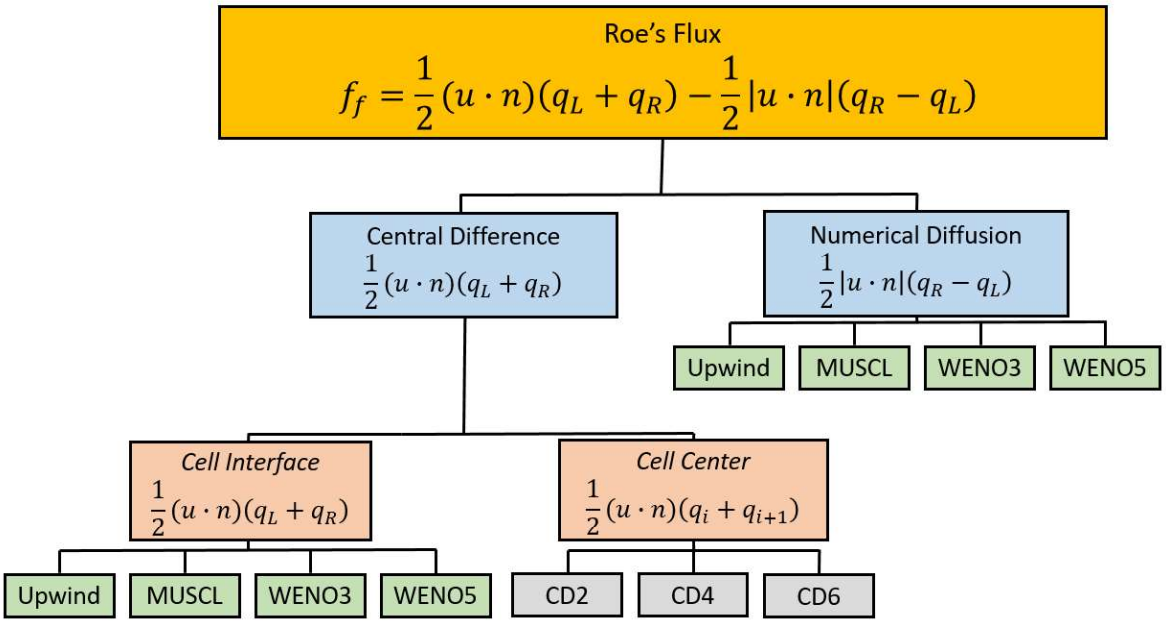

Figure 1.1: Roe Flux Outline

Six different test cases are performed to test the new hybrid Roe-WENO scheme:

- 1-D Single Discontinuity
- 1-D Half-Sine Wave
- 1-D Square Wave

- 2-D Single Discontinuity
- 2-D Half-Sine Wave
- 2-D Square Wave

Each test case will display how the different schemes handle various numerical phenomena. The two-dimensional cases are the one-dimensional cases extended to two dimensions. This allows for direct comparison of the scheme between one-dimension and multi-dimensional test cases.

1.4 Thesis Outline

Chapter 2 introduces the Roe scheme with a discussion of the linear advection equation. Then Godunov's Riemann solver discussed in detail, followed by Roe's scheme. The flux averaging is shown and a distinction is made between the CD term and the diffusive term.

Chapter 3 discusses the MUSCL scheme. Cell face extrapolation is introduced as a way to obtain higher-order accuracy. Next, the MUSCL scheme is discussed. The concept of flux limiters and why they are needed is also introduced. Finally, the coupling of the Roe scheme with MUSCL is discussed.

Chapter 4 proceeds in the same manner as Chapter 3 but for the WENO scheme. The history of the WENO scheme is detailed and how the scheme evolved over time. Next, the third- and fifth-order WENO schemes are presented. Finally, the hybrid Roe-WENO scheme is discussed.

Chapter 5 derives the fourth- and sixth-order CD fluxes. It also shows how these new fluxes are incorporated into a hybrid Roe-WENO scheme.

Chapter 6 deals with applying the various CD fluxes to test cases. The one-dimensional discontinuity, sine wave, and square wave are first discussed. The two-dimensional versions of these test cases are then shown.

Chapter 7 summarizes the findings. First, the one-dimensional results are presented followed by the two-dimensional results. The higher-order CD flux results are then discussed. It concludes with a discussion of future work involving increasing the order of averaging of the entire Roe flux.

CHAPTER 2

ROE SCHEME

Roe's scheme is the most popular approximate Riemann solver used in modern CFD codes. Chapter 2 explains Roe's scheme in detail. Godunov's scheme is also introduced as the precursor to Roe's scheme.

2.1 Linear Advection Equation

A conservation law is an equation of conserved quantities. An example of conserved quantities includes conservation of mass, momentum, and energy for the Euler equations. Equation 2.1 shows a one-dimensional hyperbolic-conservation law. A time derivative is taken of the conserved quantities. The spatial derivative is taken of a flux constructed from the conserved quantities. A conservative formulation allows for accurate representation of shocks. However, a non-conservative formulation does not guarantee shocks can be resolved correctly. As a result, a non-conservative formulation will fail when shocks are present [6].

$$U_t + f_x = 0 \tag{2.1}$$

The study of numerical methods applied to hyperbolic-conservation laws can be difficult as additional conserved variables increases the complexity of the solution method. It is therefore advantageous to study a simple conservation law. Deviations from the analytical solution are a result of the numerical scheme and not due to the nature of the equations. Thus, allowing for a direct scheme-by-scheme comparison. The linear advection equation, Equation 2.2, is the simplest conservation law available. In the linear advection equation, there is only one conserved variable q. The flux, f, is given as $f = aq$, where a is the velocity and q is the advected quantity.

$$\frac{\partial q}{\partial t} + a\frac{\partial q}{\partial x} = 0 \tag{2.2}$$

The linear advection equation has a simple solution, and is found using the method of characteristics [6]. Equation 2.3 gives the method of characteristics solution, where q_0 is the initial condition. The initial value, q_0, of Equation 2.3 remains constant. Any deviation from the initial condition is thus a result of the numerical scheme selected. As a result, the linear advection equation is most often the first equation used to test a new numerical scheme.

$$q(x,t) = q_0(x - at) \tag{2.3}$$

The linear advection equation has a numerical solution in the form of Equation 2.4 [6]. The time step and grid spacing are Δt and Δx respectively. Equation 2.4 assumes that the standard Euler time integration scheme. Other time integration schemes are available such as Runge-Kutta or implicit schemes, such as the Beam-

Warming scheme [19]. However, implicit schemes are more difficult to implement. The choice of how to evaluate the flux, f, has a large effect on the accuracy and convergence of the scheme selected.

$$q_i^{n+1} = q_i^n + \frac{\Delta t}{\Delta x}[f_{i-\frac{1}{2}} - f_{i+\frac{1}{2}}] \tag{2.4}$$

2.2 The Riemann Solver of Godunov

There are many numerical schemes available to discretize a hyperbolic-conservation law. The most famous of these schemes is the Godunov Riemann solver [4]. The Riemann problem contains a left and right state of a discontinuous quantity, shown in Figure 2.1. Godunov was the first to see that the left and right states at a cell face resembles the Riemann problem.

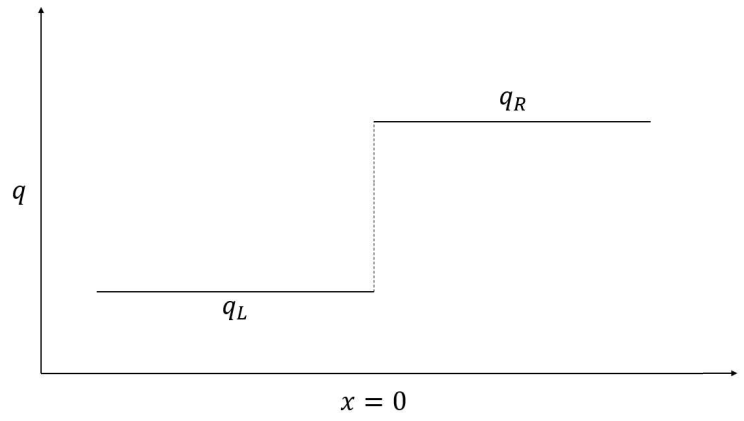

Figure 2.1: 1-D Riemann Problem for the Linear Advection Equation

The Riemann problem for the linear advection equation results in a single wave emanating from the discontinuity. Using the method of characteristics as discussed in the previous section, the derivation of an analytical solution emerges. Godunov was the first to recognize that the discontinuity left and right of a cell face is conceptually similar to the Riemann problem. He proposed using the analytical solution of the Riemann problem to determine the flux at a cell face. Godunov assumed constant piecewise data so that the cell-face quantities are equal to the cell-center quantities. Figure 2.2 shows a representation of the piecewise-constant data. It shows, that at the cell face, $i + \frac{1}{2}$, there is a local Riemann problem.

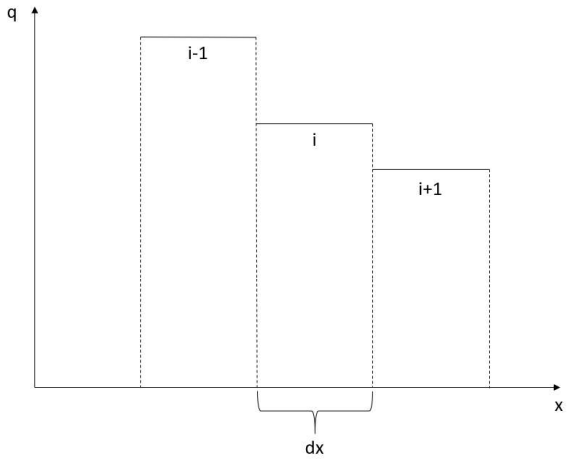

Figure 2.2: Configuration for the One-Dimensional Godunov Solver

As a result of Godunov's scheme treating the cell-face data as a local Riemann problem, it becomes an "exact" Riemann solver [5]. Because the cell-face values are discontinuous, Godunov's method captures shocks very accurately. However, due to the piecewise-data assumption, Godunov's scheme is only first-order accurate.

Additionally, the analytical solution of the Riemann problem for the Euler equations requires an iterative procedure to evaluate the pressure making Godunov's scheme computationally expensive.

While Godunov's method was designed to solve the Euler equations, the scheme can be used to solve any conservation law. For example, the Godunov scheme can be applied to the linear advection equation. Solving the local Riemann problems, $RP(q_{i-1}, q_i)$ and $RP(q_i, q_{i+1})$, gives values for the fluxes shown in Equation 2.5 and Equation 2.6.

$$f_{i+\frac{1}{2}} = aq_i \tag{2.5}$$

$$f_{i-\frac{1}{2}} = aq_{i-1} \tag{2.6}$$

The solution of the linear advection equation using Godunov's method, shown in Equation 2.7, recovers the upwind scheme. The upwind scheme is very diffusive resulting in smearing of shocks and other flow features. The order of accuracy increases when changing the piecewise-constant data assumption to a function. The computational expense reduces when using an approximate Riemann solver, e.g. the Roe Riemann solver.

$$q_i^{n+1} = q_i^n + \frac{a\Delta t}{\Delta x}[q_{i-1}^n - q_i^n] \tag{2.7}$$

2.3 Roe Riemann Solver

In order to avoid the costly pressure iteration in Godunov's method, approximate Riemann solvers were developed. Approximate Riemann solvers approximate Godunov's exact method allowing for reduced computation times. The most widely used approximate Riemann solver is the Roe Riemann solver. Roe's scheme contains both accuracy and computational efficiency, which is the reason it is widely used.

The Roe Riemann solver approximates the Jacobian flux. Equation 2.8 shows the conservation law written with the Jacobian flux.

$$\frac{\partial q}{\partial t} + A\frac{\partial q}{\partial x} = 0 \tag{2.8}$$

Where A is the Jacobian flux matrix. A self-similar solution will exist if the eigenvalues of A are all real, and the eigenvectors are complete. Roe proposed to approximate the Jacobian flux matrix with a constant coefficient-linear system. The approximate Jacobian flux is constructed of Roe-averaged states. The linearized Jacobian replaces A with $\tilde{A}$. These Roe-averaged states are taken from the left- and right-cell faces. The numerical flux can then be obtained and is given by Equation 2.9 and Equation 2.10.

$$F_{i+\frac{1}{2}} = F_L + \sum_{\tilde{\lambda}_i \leq 0} \tilde{\alpha}_i \tilde{\lambda}_i \tilde{\mathbf{K}}^{(i)} \tag{2.9}$$

$$F_{i+\frac{1}{2}} = F_R - \sum_{\tilde{\lambda}_i \geq 0} \tilde{\alpha}_i \tilde{\lambda}_i \tilde{\mathbf{K}}^{(i)} \tag{2.10}$$

Where $F_{L,R}$ is the left or right flux, $\tilde{\alpha}_i$ is the wave strength, $\tilde{\lambda}_i$ are the eigenvalues of the linearized Jacobian, and $\tilde{\mathbf{K}}^{(i)}$ is the right eigenvector of the conserved equations. Left-moving waves make up Equation 2.9 while right-moving waves make up Equation 2.10. A simple averaging of the two fluxes gives Equation 2.11.

$$F_{i+\frac{1}{2}} = \frac{1}{2}(F_L + F_R) - \frac{1}{2}\sum_{i=1}^{m} \tilde{\alpha}_i |\tilde{\lambda}_i| \tilde{\mathbf{K}}^{(i)} \tag{2.11}$$

The first term in Roe's flux is a central differencing about the cell face, $i + \frac{1}{2}$. The second term is a diffusive term that adds numerical viscosity to the flux. Roe initially developed this method to solve the Euler equations, however, they are applicable to any conservation law. For the linear advection equation, the complexity of Roe's flux reduces to give Equation 2.12.

$$f_{i+\frac{1}{2}} = \frac{1}{2}(f_L + f_R) - \frac{1}{2} \mid \alpha \mid (q_R - q_L) \tag{2.12}$$

Where $|\alpha|$ is the linearized Jacobian and $q_{R,L}$ is the advected quantity on the right and left states. For the linear advection equation the Jacobian is evaluated to be a as shown in Equation 2.13.

$$\mid \alpha \mid = \frac{\partial f}{\partial q} = a \tag{2.13}$$

For more complex systems of equations, such as Euler's equations, the Jacobian has more complexity. Substituting for f_L and f_R in the linear advection equation leads to Equation 2.14.

$$f_{i+\frac{1}{2}} = \frac{a}{2}(q_L + q_R) - \frac{a}{2}(q_R - q_L) \tag{2.14}$$

If constant piecewise cell-centered data is used, the Roe flux reduces to Equation 2.15.

$$f_{i+\frac{1}{2}} = aq_i \tag{2.15}$$

The results of Equation 2.15 show that for the linear advection equation, the Roe scheme is identical to the Godunov scheme. Since the Godunov scheme is also identical to the upwind scheme, this means that the Roe scheme is identical to the upwind scheme for the linear advection equation with constant piecewise-cell data. Note this is only true for the linear advection equation. For more complex equations (such as the Euler equations), the linearized Jacobian will be different.

Roe's scheme has first-order accuracy, the same as Godunov's method. Traditionally, to increase the order of accuracy of Roe's scheme, the underlining assumption of piecewise constant-cell states must be reevaluated.

2.4 Temporal Scheme

A third-order Total Variation Diminishing (TVD) Runge-Kutta is used for the time integration as shown in Equation 2.16 through Equation 2.18. The TVD Runge-Kutta scheme has the advantage of avoiding spurious oscillations in the temporal domain. The higher-order accuracy of the scheme reduces numerical error compared to the first-order Euler method. Because of these advantageous properties, Jiang and

Shu [20] used this numerical scheme when analyzing the WENO scheme. The third-order Runge-Kutta (RK3) scheme works by having two time steps between the actual time step giving the temporal stencil two additional points and is able to obtain a more accurate solution.

$$q^{(1)} = q^n + \Delta t L(q^n) \tag{2.16}$$

$$q^{(2)} = \frac{3}{4}q^n + \frac{1}{2}q^{(1)} + \frac{1}{4}\Delta t L(q^{(1)}) \tag{2.17}$$

$$q^{(n+1)} = \frac{1}{3}q^n + \frac{2}{3}q^{(2)} + \frac{2}{3}\Delta t L(q^{(2)}) \tag{2.18}$$

CHAPTER 3

ROE-MUSCL SCHEME

The Roe scheme can be improved by increasing the order of accuracy of the left- and right-cell face values used in Roe's flux. The order of accuracy at the cell face increase by extrapolating the values from cell-centered data to the right and left of the cell face. The most commonly used scheme for cell-face reconstruction is van Leer's MUSCL scheme.

3.1 Cell-Face Extrapolation

The accuracy of Riemann solvers is generally limited to the accuracy of the cell-face values. The assumption of piecewise-constant values for the cell faces is a first-order reconstruction about the cell face. Higher-order schemes must use higher-order reconstruction at the cell face. Developing higher-order reconstruction methods is non-trivial due to the nature of higher-order numerical methods for hyperbolic problems. Near strong gradients, higher-order methods have oscillations which are non-physical. Godunov's theorem states that "Linear numerical schemes for solving partial differential equations, having the property of not generating new extrema, can be at most first-order accurate" [4]. Godunov's theorem is the reason high-order

methods for solving hyperbolic problems is not trivial. A solution is considered TVD if Equation 3.1 holds true.

$$TVq^{n+1} \leq TVq^{n} \tag{3.1}$$

There are a number of ways to make higher-order schemes TVD. One commonly used method is to use numerical dissipation in regions of strong gradients. First-order schemes are TVD due to the large amount of numerical dissipation. Therefore, by adding numerical viscosity in regions of high gradients, those regions will become TVD, while the rest of the solution retains higher-order accuracy. The MUSCL scheme uses this method to ensure that the solution remains monotonic. Another way to achieve monoticity is via a non-linear scheme, which is similar to the WENO approach.

3.2 Monotonic Upwind Scheme for Conservation Laws

The MUSCL scheme of van Leer replaces the piecewise-constant data of Godunov's method with piecewise-linear functions shown in Figure 3.1. By assuming piecewise-linear functions, the Riemann problem is evaluated for second-order accuracy resulting in more precise solutions. There are several options available for piecewise-linear functions. Equation 3.2 and Equation 3.3 are general functions for linear reconstruction [21].

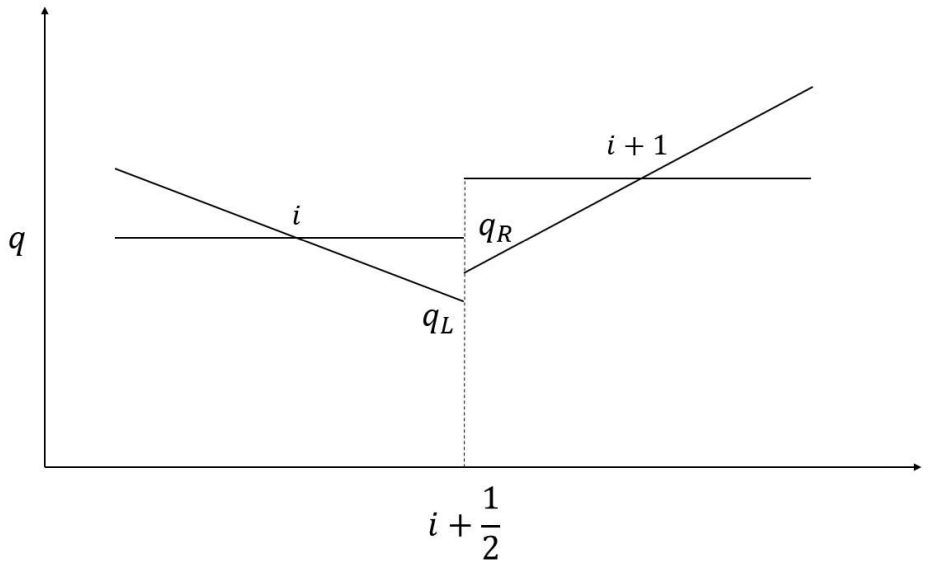

Figure 3.1: Configuration for the One-Dimensional MUSCL Scheme

$$q^L_{i+\frac{1}{2}} = q_i + \phi[\frac{1}{4}(1-\kappa)(q_i - q_{i-1}) + \frac{1}{4}(1+\kappa)(q_{i+1} - q_i)] \tag{3.2}$$

$$q^R_{i-\frac{1}{2}} = q_i + \phi[\frac{1}{4}(1+\kappa)(q_i - q_{i-1}) + \frac{1}{4}(1-\kappa)(q_{i+1} - q_i)] \tag{3.3}$$

Where ϕ is a limiter used to increase the numerical viscosity in regions with strong gradients, and modifications to κ changes the type of gradient (e.g. upwind or central). Different values of κ will result in different formulations based on how the gradients are determined. For $\kappa = -1$, the MUSCL scheme results in a fully upwind scheme; $\kappa = 0$, a second-order Fromm scheme; and $\kappa = 1$, a central-differencing scheme [21]. For the linear advection equation, the MUSCL scheme behaves best

when $\kappa = 1$. The finite-difference approach using $\kappa = 1$ results in piecewise-linear functions given by Equation 3.4 and Equation 3.5.

$$q^L_{i+\frac{1}{2}} = q_i + \frac{\phi}{2}(q_{i+1} - q_i) \tag{3.4}$$

$$q^R_{i-\frac{1}{2}} = q_i + \frac{\phi}{2}(q_i - q_{i-1}) \tag{3.5}$$

The limiter, ϕ, bounds the slope of the piecewise-constant function so that the solution remains TVD. There are a number of different limiters possible such as van Leer's [22] or van Albada's [23]. The most diffusive limiter is the minmod limiter developed by Roe [24]. The overly diffusive minmod limiter is better at keeping the solution TVD than other limiters, however at the cost of extra numerical diffusion. Equation 3.6 gives the mimmod limiter.

$$\phi = max(0, min(1, r)) \tag{3.6}$$

Where r is a ratio of forward and backward gradients and is a smoothness measurement of the gradient. For the finite-difference method, Equation 3.7 evaluates r.

$$r_{i+\frac{1}{2}} = \frac{q_i - q_{i-1}}{q_{i+1} - q_i} \tag{3.7}$$

Since r is a smoothness measurement and ϕ is dependent on r, it can clearly be seen that, if the solution is smooth, ϕ will be equal to one, and there will be no

limiting. If there is a strong discontinuity in the solution, then ϕ will trend toward zero making that part of the solution first-order accurate. The MUSCL scheme allows for high-order accuracy in smooth parts of the solution, and adds numerical diffusion in areas of strong gradients keeping the solution TVD and higher-order accurate.

3.3 Roe-MUSCL

By pairing the Roe scheme with the MUSCL scheme, the Roe scheme becomes second-order accurate. MUSCL-reconstructed values replace the left and right states resulting in more accurate cell-face values for the local Riemann problem. Therefore, the Roe-MUSCL scheme is a more accurate representation of the flux at the cell face. A variety of reconstruction schemes are available to increase the order of accuracy of the Roe scheme. For example, the WENO scheme extends the Roe scheme to third- and fifth-order accuracy.

CHAPTER 4

ROE-WENO SCHEME

Increased order of accuracy makes higher-order schemes more desirable. The MUSCL scheme achieves this by increasing the Roe scheme to second-order accurate. However, some applications require accuracy beyond second order. The WENO scheme was developed to be a high-resolution scheme and be extended to n^{th}-order accuracy.

4.1 Weighted Essentially Non-Oscillatory Scheme

The WENO scheme is a more robust and efficient ENO scheme. The ENO scheme evaluates stencils and determines which is the "smoothest". The smoothest stencil is chosen to approximate the fluxes at cell boundaries. The ENO scheme avoids spurious oscillations at discontinuities because it uses the smoothest stencil. The accuracy of the ENO scheme stems from the higher-order stencils used to evaluate fluxes. A disadvantage of the ENO scheme is that the stencil selection process is rather cumbersome. This is a result of the stencil selection process requiring multiple logic statements which are computational expensive. Also, in smooth regions of the

solution, two stencils are computed. The ENO scheme only uses one of these stencils while the other stencil is calculated without improvement of accuracy.

The original WENO scheme reduces the computational expense of the ENO scheme for cell-averaged values. Each cell is assigned a stencil and a convex combination of the stencils is performed to get a new stencil. Each stencil is assigned a weight that goes to zero if a discontinuity is present. If the stencil is located in a smooth region, the weight reaches its optimal value. The WENO scheme avoids the computationally inefficient logic statements of the ENO scheme as well as spurious oscillations near discontinuities. The convex combinations also cancel truncation errors to increase the order of the WENO scheme.

Further WENO refinement was achieved via Jiang and Shu [20] selecting a different smoothness measurement. The refined WENO scheme extrapolates cell-face quantities. The WENO scheme of Jiang and Shu is based on the cell-face fluxes and not the cell-centered values. The order of accuracy of the flux-based WENO scheme is $2r - 1$, where r is the number of points used in the stencil. This is a significant improvement of the cell-centered WENO scheme with an order of accuracy of $r + 1$.

The WENO scheme is essentially non-oscillatory which means any oscillations in the solution will be small and almost undetectable. In contrast to the MUSCL scheme, the WENO scheme reduces the dispersion error where as the MUSCL scheme adds numerical viscosity. Unlike the MUSCL scheme, the WENO scheme does not become first-order accurate when strong gradients are detected. Therefore, the WENO scheme provides better resolution of shocks than the MUSCL scheme.

The stencil points, S_k, are chosen by Equation 4.1, where $k = 0, 1, ..., r-1$, and r determines how many stencils.

$$S_k = (x_{j+k-r+1}, x_{j+k-r+2}, ..., x_{j+k}) \tag{4.1}$$

The order of accuracy scales as $2r-1$ so that increasing the number of stencils results in an additional two orders of accuracy. The r values used in this study are $r = 2$ for a third-order WENO scheme, and $r = 3$ for a fifth-order WENO scheme. The stencils are given by Equation 4.2.

$$q_k^r(g_0, ..., g_{r-1}) = \sum_{l=0}^{r-1} a_{k,l}^r g_l \tag{4.2}$$

Where $a_{k,l}^r$ is the coefficient and g_l is the quantity in the cell center. The number of stencils depends on the value of r, and the number of points in each stencil is a function of r. With Equation 4.2, the basis for making $2r-1$ order accurate schemes is complete. The coefficients for the stencils are given in Table 4.1 and the coefficients for the optimal weights are given in Table 4.2.

Table 4.1: Stencil Coefficients for WENO Scheme based on Paper by Shu

r	k	$l=0$	$l=1$	$l=2$
2	0	$-\frac{1}{2}$	$\frac{3}{2}$	
	1	$\frac{1}{2}$	$\frac{1}{2}$	
3	0	$\frac{1}{3}$	$-\frac{7}{6}$	$\frac{11}{6}$
	1	$-\frac{1}{6}$	$\frac{5}{6}$	$\frac{1}{3}$
	2	$\frac{1}{3}$	$\frac{5}{6}$	$-\frac{1}{6}$

Table 4.2: Coefficients for the Optimal Weights for WENO3 and WENO5 Scheme based on Paper by Shu

C_k^r	k=0	k=1	k=2
$r=2$	$\frac{1}{3}$	$\frac{2}{3}$	-
$r=3$	$\frac{1}{10}$	$\frac{6}{10}$	$\frac{3}{10}$

The WENO scheme differs from the MUSCL scheme in that the reconstructed states are polynomial and not linear functions. Figure 4.1 shows the structure of the WENO reconstruction stencils. The convex combination of the stencils creates non-linear functions. The use of these non-linear functions allows the WENO scheme to obtain higher-order accuracy. Because the functions are polynomials, additional stencils are added to increase the order of the scheme.

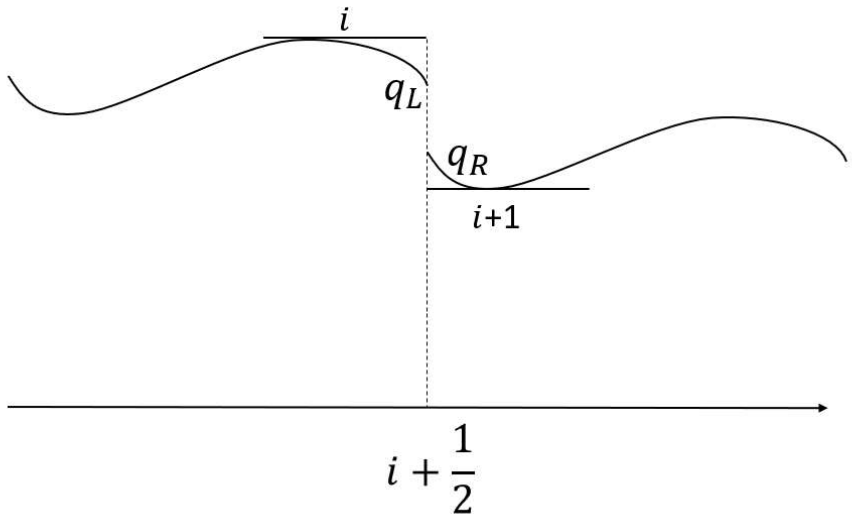

Figure 4.1: WENO Reconstruction

4.1.0.1 WENO3

The simplest WENO scheme used is the third-order WENO scheme (WENO3). The value of r is given as $r = 2$ resulting in two stencils. Figure 4.2 describes the stencils for the right-cell face. Here two stencils, with optimal weights, make up the complete stencil.

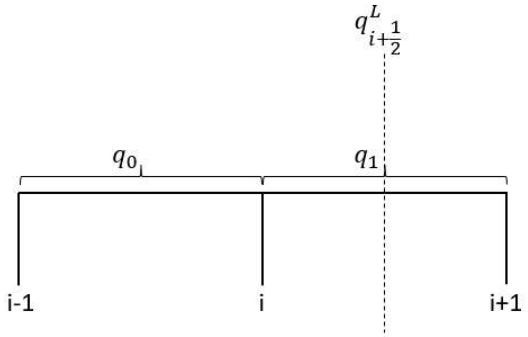

Figure 4.2: Configuration for the WENO3 Scheme

Equation 4.3 and Equation 4.4 gives the right face, q^R, of the WENO3 scheme smoothness indicator, IS.

$$IS_1 = (q_{i+2} - q_{i+1})^2 \tag{4.3}$$

$$IS_2 = (q_{i+1} - q_i)^2 \tag{4.4}$$

Equation 4.5 and Equation 4.6 give the attached stencil weights.

$$\omega_1 = \frac{\alpha_1}{\sum \alpha} \tag{4.5}$$

$$\omega_2 = \frac{\alpha_2}{\sum \alpha} \tag{4.6}$$

Equation 4.7 and Equation 4.8 give the α values.

$$\alpha_1 = \frac{C_1}{(\varepsilon + IS_1)^2} \tag{4.7}$$

$$\alpha_2 = \frac{C_2}{(\varepsilon + IS_2)^2} \tag{4.8}$$

Table 4.2 shows the values for the optimal weights, C. The parameter ε is used to avoid dividing by zero by setting it to $1e-6$. Equation 4.9 and Equation 4.10 give the stencils.

$$q_1 = \frac{3}{2}U_{i+1} - \frac{1}{2}U_{i+2} \tag{4.9}$$

$$q_2 = \frac{1}{2}U_i - \frac{1}{2}U_{i+1} \tag{4.10}$$

Multiplying the weights and the stencils yields the convex combination of stencils, q, in Equation 4.11.

$$q^R_{i+\frac{1}{2}} = \omega_1 q_1 + \omega_2 q_2 \tag{4.11}$$

The left cell-face is reconstructed symmetrically about the right. The WENO3 scheme has third-order accuracy, which means its order of convergence is higher than the MUSCL scheme. Increasing the accuracy of the WENO scheme is straight forward and detailed in the next section.

4.1.0.2 WENO5

The fifth-order WENO scheme (WENO5) uses one more stencil point and one more stencil than the WENO3 scheme. Since the order of accuracy scales as $(2r-1)$, the total order is fifth order. The Roe scheme used with the fifth-order WENO scheme is done in a similar manner as Shen, Wang, and Zha [25]. Figure 4.3 shows the stencil for the WENO5 scheme.

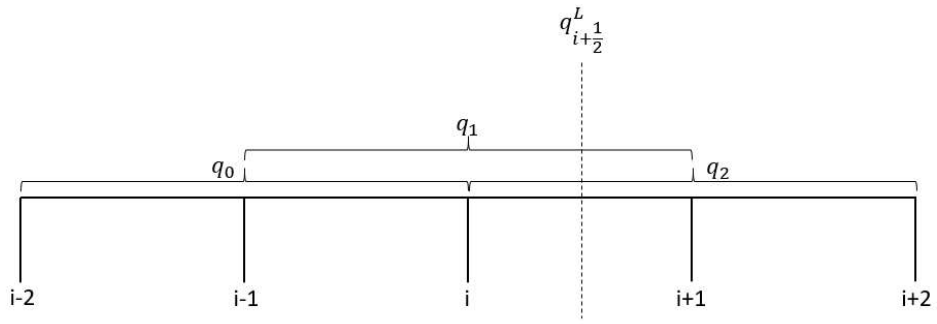

Figure 4.3: Configuration for the WENO5 Scheme

Equation 4.12 through Equation 4.14 show the three WENO5 scheme smoothness terms.

$$IS_0 = \frac{13}{12}(q_{i-1} - 2q_i + q_{i+1})^2 + \frac{1}{4}(q_{i-1} - 4q_i + 3q_{i+1})^2 \tag{4.12}$$

$$IS_1 = \frac{13}{12}(q_i - 2q_{i+1} + q_{i+2})^2 + \frac{1}{4}(q_i - q_{i+2})^2 \tag{4.13}$$

$$IS_2 = \frac{13}{12}(q_{i+1} - 2q_{i+2} + q_{i+3})^2 + \frac{1}{4}(3q_{i+1} - 4q_{i+2} + q_{i+3})^2 \tag{4.14}$$

Equation 4.15 through Equation 4.17 yield the smoothness measurements used to compute the α values in the optimal weights equations.

$$\alpha_0 = \frac{C_0}{(\varepsilon + IS_0)^2} \tag{4.15}$$

$$\alpha_1 = \frac{C_1}{(\varepsilon + IS_1)^2} \tag{4.16}$$

$$\alpha_2 = \frac{C_2}{(\varepsilon + IS_2)^2} \tag{4.17}$$

Equation 4.18 through Equation 4.20 show the optimal weights, C.

$$\omega_0 = \frac{\alpha_0}{\sum \alpha} \tag{4.18}$$

$$\omega_1 = \frac{\alpha_1}{\sum \alpha} \tag{4.19}$$

$$\omega_2 = \frac{\alpha_2}{\sum \alpha} \tag{4.20}$$

Equation 4.21 through Equation 4.23 yield the stencils.

$$q_0 = \frac{1}{6}(-U_{i-1} + 5U_i + 2U_{i+1}) \tag{4.21}$$

$$q_1 = \frac{1}{6}(2U_i + 5U_{i+1} - U_{i+2}) \tag{4.22}$$

$$q_2 = \frac{1}{6}(11U_{i+1} - 7U_{i+2} + 2U_{i+3}) \tag{4.23}$$

Equation 4.24 shows the convex combination of the right cell-face values. The left cell-face value is constructed symmetrically about the right.

$$q^R_{i+\frac{1}{2}} = \omega_0 q_0 + \omega_1 q_1 + \omega_2 q_2 \tag{4.24}$$

4.2 Roe-Weno

The WENO scheme increases the accuracy of the Roe scheme in the same manner as MUSCL reconstruction. Since WENO is higher order than the MUSCL scheme, it has less truncation error leading to more accurate results. In addition the solution to the Riemann problem at the cell face is a better representation of the flux through the cell, since the reconstructed values are higher-order accurate.

CHAPTER 5

MODIFICATION TO ROE'S SCHEME

An increase to the order of accuracy of the Roe scheme requires modifications. The most popular modification is to increase the order of accuracy of the extrapolated left- and right-state values similar to MUSCL and WENO. Most attempts to modify the scheme ignore the central-differencing term as the extrapolated values left and right states are absorbed in the fluxes. Since the central-differencing term is only order $O(\Delta x^2)$, improvements to the order of accuracy can be made with little computational cost.

5.1 Proof of Second-Order Accuracy

Equation 5.1 gives the second-order central differencing.

$$F_{CD2} = \frac{1}{2}(F_L + F_R) \tag{5.1}$$

If a first-order extrapolation is used such that the value of a cell is considered constant throughout the cell, then the equation is in the finite-difference form as shown in Equation 5.2.

$$F_{CD2} = \frac{1}{2}(F_i + F_{i+1}) \tag{5.2}$$

Taylor series expansion of the fluxes shows the order of accuracy of Equation 5.2 leads to Equation 5.3 and Equation 5.4

$$F_{i+1} = F_{i+\frac{1}{2}} + \frac{\Delta x}{2}\frac{\partial F}{\partial x} + O(\Delta x^2) \tag{5.3}$$

$$F_i = F_{i+\frac{1}{2}} - \frac{\Delta x}{2}\frac{\partial F}{\partial x} + O(\Delta x^2) \tag{5.4}$$

Using Equation 5.3 and Equation 5.4 to solve for $F_{i+\frac{1}{2}}$ leads to Equation 5.5.

$$F_{i+\frac{1}{2}} = \frac{1}{2}(F_i + F_{i+1}) + O(\Delta x^2) \tag{5.5}$$

Equation 5.5 indicates the CD term in the Roe scheme is of second-order accuracy. The same procedure shows $F_{i-\frac{1}{2}}$ is also second-order accurate. Since this is a central difference about the cell face, the order of accuracy is limited to even numbers. Reconstruction schemes do not modify the CD term in Roe's flux. Consequently, a new higher-order central differencing about the cell face will be developed.

5.2 Derivation of a Fourth-Order Central-Differencing Scheme

The next CD derived is a fourth-order CD which has the advantage of less dispersion error than the second-order scheme. A fourth-order accurate central dif-

ferencing is derived again by using Taylor series expansions given by Equation 5.6 through Equation 5.9.

$$F_{i+2} = F_{i+\frac{1}{2}} + \frac{3}{2}\Delta x\frac{\partial F}{\partial x} + \frac{9}{8}\Delta x^2\frac{\partial^2 F}{\partial x^2} + \frac{27}{48}\Delta x^3\frac{\partial^3 F}{\partial x^3} + O(\Delta x^4) \tag{5.6}$$

$$F_{i+1} = F_{i+\frac{1}{2}} + \frac{1}{2}\Delta x\frac{\partial F}{\partial x} + \frac{1}{8}\Delta x^2\frac{\partial^2 F}{\partial x^2} + \frac{1}{48}\Delta x^3\frac{\partial^3 F}{\partial x^3} + O(\Delta x^4) \tag{5.7}$$

$$F_{i} = F_{i+\frac{1}{2}} - \frac{1}{2}\Delta x\frac{\partial F}{\partial x} + \frac{1}{8}\Delta x^2 8\frac{\partial^2 F}{\partial x^2} - \frac{1}{48}\Delta x^3\frac{\partial^3 F}{\partial x^3} + O(\Delta x^4) \tag{5.8}$$

$$F_{i-1} = F_{i+\frac{1}{2}} - \frac{3}{2}\Delta x\frac{\partial F}{\partial x} + \frac{9}{8}\Delta x^2 8\frac{\partial^2 F}{\partial x^2} - \frac{27}{48}\Delta x^3\frac{\partial^3 F}{\partial x^3} + O(\Delta x^4) \tag{5.9}$$

There are four equations and four unknowns which leads to a system of algebraic equations. Solving the set of equations algebraically gives Equation 5.10.

$$F_{i+\frac{1}{2}} = \frac{1}{16}(-F_{i-1} + 9F_i + 9F_{i+1} - F_{i+2}) + O(\Delta x^4) \tag{5.10}$$

Equation 5.10 is a fourth-order CD about the cell face, $i + \frac{1}{2}$. F_{CD4} has a lower discretization error than its second-order counterpart and will have lower computational cost (at least for multi-component fluxes) than Roe's original CD flux. However, for the linear advection equation, the cost is slightly higher as there is no need to recalculate the flux.

5.3 Derivation of Sixth-Order Central-Differencing Scheme

A sixth-order CD is constructed in a similar manner to the fourth-order CD flux. The complete derivation is cumbersome and tedious and is not shown here, although it is given in Appendix A. Equation 5.11 gives the sixth-order flux.

$$F_{i+\frac{1}{2}} = \frac{1}{256}(3F_{i+3} - 25F_{i+2} + 150F_{i+1} + 150F_i - 25F_{i-1} + 3F_{i-2}) \tag{5.11}$$

The higher sixth-order CD flux uses a total of six stencil points whereas the fourth-order flux uses four and the second-order flux uses two. The further away the discrete points are from the cell face, the lower the weights are in front of the flux. This indicates that increasing the order of accuracy has a negative return in terms of accuracy versus computational cost.

5.4 Hybrid Roe-WENO Scheme

Roe's original CD flux is replaced with the second-, fourth-, and sixth-order CD flux and the accuracies are compared. The dissipation term is unmodified and upwind, MUSCL, WENO3, and WENO5 reconstruction are applied. Roe's original flux is also compared as a baseline of how the scheme was originally designed with the reconstructed values used in the central differencing. Chapter 6 shows the results of these modifications.

CHAPTER 6

NUMERICAL EXPERIMENTS

The linear advection equation, modeled in both one- and two-dimensions, is a very fundamental equation. The scalar quantity, q, advects quantities along the direction of propagation. As the solution changes, the shape of the solution remains constant, consequently the linear advection equation is often the first equation used to test a numerical scheme. A numerical scheme is defined or represented by how much the numerical solution diverges from the analytical solution. Over time the solution smears discontinuities and the maximum value of the solution decreases due to numerical viscosity. How much a solution smears is often a representation of the order of accuracy of the solution. The one-dimensional and two-dimensional linear advection equations are modeled with second-, fourth-, and sixth-order central differencing. Roe's original flux formulation serves as a baseline case. Reconstruction is applied to the cells using MUSCL, WENO3, and WENO5 extrapolation. The solution changes depending on the choice of central differencing applied.

6.1 Figure Legend Naming Scheme

The legend on the figures in this chapter describe the CD scheme and reconstruction scheme used to generate the curve. The left term of each legend indicates the CD scheme used for the central-difference term in Roe's flux. The higher-order CD schemes are CD2, CD4, and CD6. The higher-order CD schemes use the cell-center values of adjacent cells to calculate the CD flux. The curve labeled Roe refers to Roe's original CD where a simple averaging of the left- and right-cell face values is used. This is in contrast to the cell-centered values used in the other CD schemes. The left- and right-cell face values are made higher-order by reconstruction.

The right term of each legend indicates the type of reconstruction used. The reconstruction options used are upwind, MUSCL, WENO3, and WENO5. The reconstruction terms are used in the diffusive flux. When Roe's original CD flux is used, the reconstructed values at the cell face are used for the averging. The exact solution to the test case is also given by a dashed line.

6.2 One-Dimensional Cases

The one-dimensional linear advection equation is perhaps the simplest equation to test a numerical scheme. It is a partial differential equation with one spatial and one temporal dimension. The linear advection equation is a simplified version of the inviscid Burger's equation. The non-linear term in Burger's equation is made linear by assuming a constant advection velocity resulting in the one-dimensional linear advection equation given by Equation 6.1.

$$\frac{\partial q}{\partial t} + a\frac{\partial q}{\partial x} = 0 \tag{6.1}$$

The solution of Equation 6.1 is found by using the method of characteristics [6] with the analytical solution given by Equation 6.2.

$$q(x,t) = q_0(x - at) \tag{6.2}$$

The significance of Equation 6.2 is that the initial value of the problem remains constant. Any deviation from the initial value is a result of the numerical scheme. Three one-dimensional test cases are studied: a step profile, a half-sine wave, and a square wave. These were taken from Toro [6] as they are excellent one-dimensional test problems.

6.2.1 1-D Step Function

The first case tested consists of a 1-D step function advected with a constant velocity $a = 1$ on a uniform grid with 512 cells. The step function is initially placed at $x = 0$ with out flow boundary conditions located at $x = -1$ and $x = 1$. The test conditions and the mesh spacing are the same as those used in [5]. The step function is advected until time $t = 0.5$ at which point it has moved a distance of 0.5. Figure 6.1 shows the initial condition of the 1-D step. The solution to the 1-D step gives an indication of the ability of a numerical scheme to capture shocks.

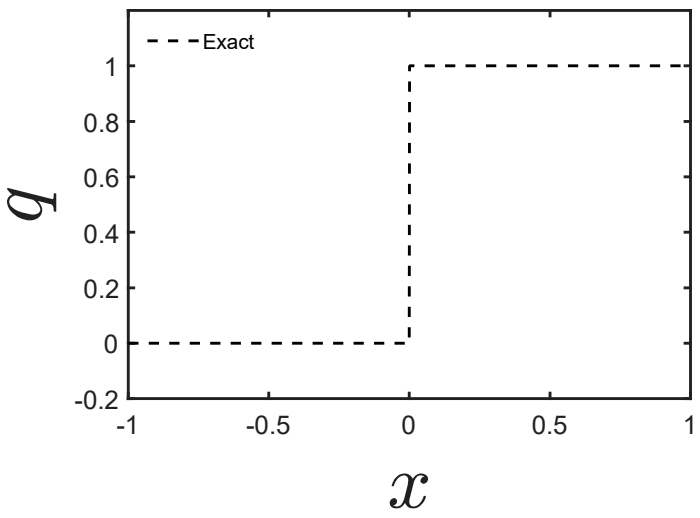

Figure 6.1: 1-D Step Initial Setup

6.2.2 1-D Step Function Results

Figure 6.2 shows the solution when using upwind reconstruction for the four central-differencing schemes. All of the curves fall on top of each other and give a poor representation of the shock. This is an indication that the diffusion term is dominating the flux calculation. As a result, increasing the order of accuracy of the central-difference terms will not increase the accuracy of the solution. In order to see the effect of high-order central differencing, higher-order reconstruction is used.

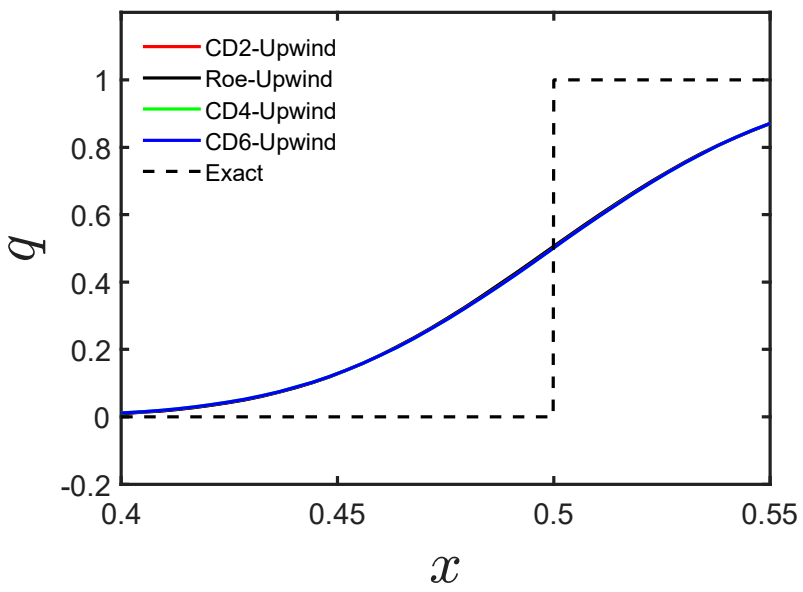

Figure 6.2: Upwind Reconstruction for the Diffusion Term with CD2, CD4, CD6, and Roe Central Differencing for the CD term for a 1-D Step

Figure 6.3 shows the effect of the four reconstruction schemes for the diffusive term with the CD2 scheme for the CD term. The CD2 flux paired with MUSCL reconstruction gave results that are more accurate and TVD. The WENO3 reconstruction scheme resolves the shock slightly better than the MUSCL scheme. However, the CD2 flux with WENO3 reconstruction is not TVD. As expected, the upwind scheme smears the shock the most out of all schemes tested which is a result of the scheme being only first-order accurate. The WENO5 scheme has unstable behavior with spurious oscillations near the discontinuity.

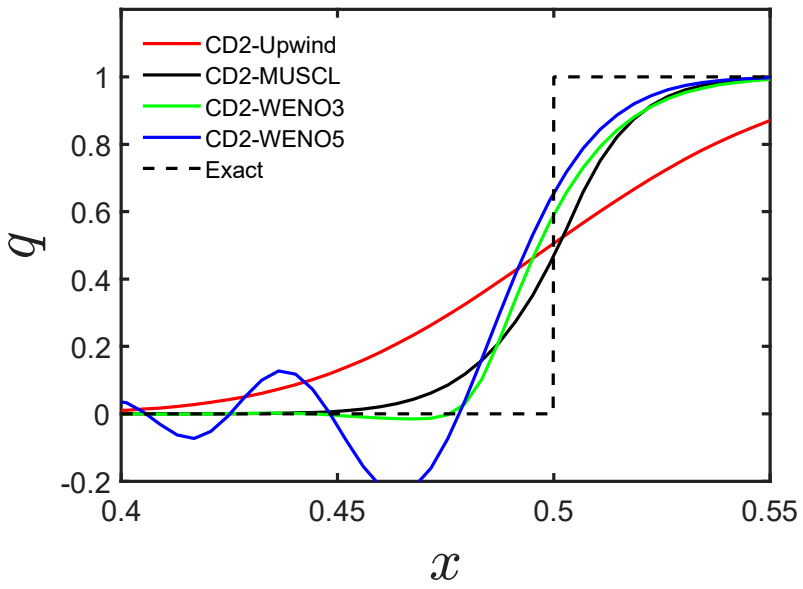

Figure 6.3: CD2 Central Differencing for the CD Term with Upwind, MUSCL, WENO3, and WENO5 Reconstruction for the Diffusion Term for a 1-D Step

Figure 6.4 shows the results for the four reconstruction schemes for the diffusive term with the CD4 scheme for the CD term. The fourth-order CD flux has reduced dispersion error compared to the CD2 scheme. Using WENO3 reconstruction gives a solution that is TVD and resolves the shock more accurately than the MUSCL scheme. The WENO5 scheme has small oscillations and a numerical feature in the shock front. The numerical feature is a result of reduced numerical viscosity compared to the WENO3 scheme. The upwind and MUSCL schemes are unaffected by the change in CD scheme.

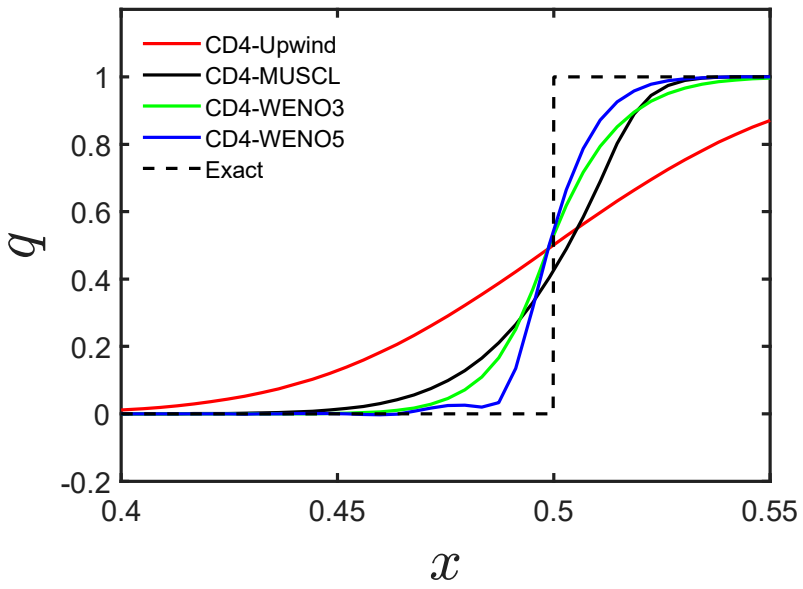

Figure 6.4: CD4 Central Differencing for the CD Term with Upwind, MUSCL, WENO3, and WENO5 Reconstruction for the Diffusion Term for a 1-D Step

Figure 6.5 shows the results of the four reconstruction schemes for the diffusive term with the CD6 scheme for the CD term. Using WENO5 reconstruction resolves the shock most accurately and the solution is TVD. All spurious oscillations are removed from the solution. Increasing the order of accuracy of the CD term using upwind, MUSCL, or WENO3 reconstruction does not change the accuracy of the solution. This is a result of the diffusion term in Roe's flux dominating over the CD term.

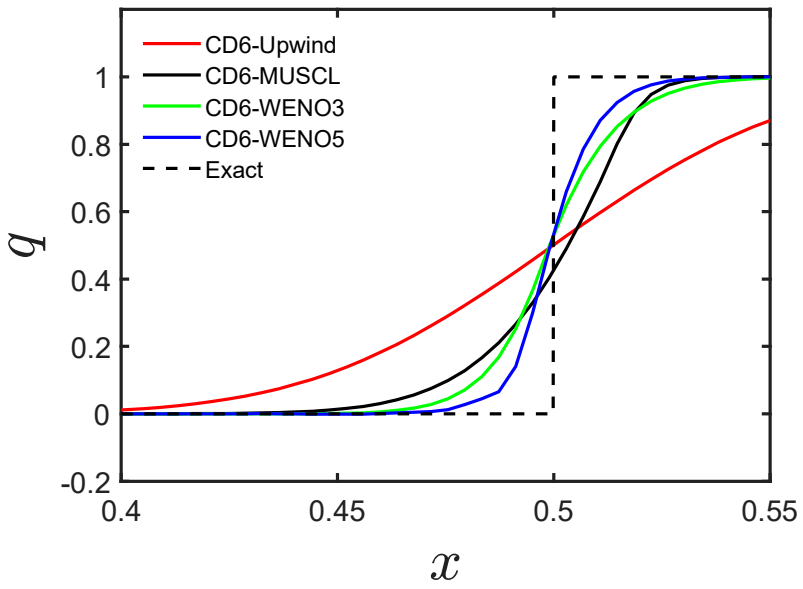

Figure 6.5: CD6 Central Differencing for the CD Term with Upwind, MUSCL, WENO3, and WENO5 Reconstruction for the Diffusion Term for a 1-D Step

Figure 6.6 shows the results of the four reconstruction schemes for the diffusive term with the Roe's original flux scheme for the CD term. All schemes are TVD and have a crisp representation of the discontinuity. The WENO5 scheme is the most accurate, followed by the WENO3 scheme then the MUSCL scheme. The results are reasonable as the MUSCL scheme is second-order accurate and the WENO3 scheme is third-order accurate. Roe's CD scheme uses second-order averaging of the higher-order left and right states. The higher the order of the reconstruction, the more accurate Roe's CD scheme becomes.

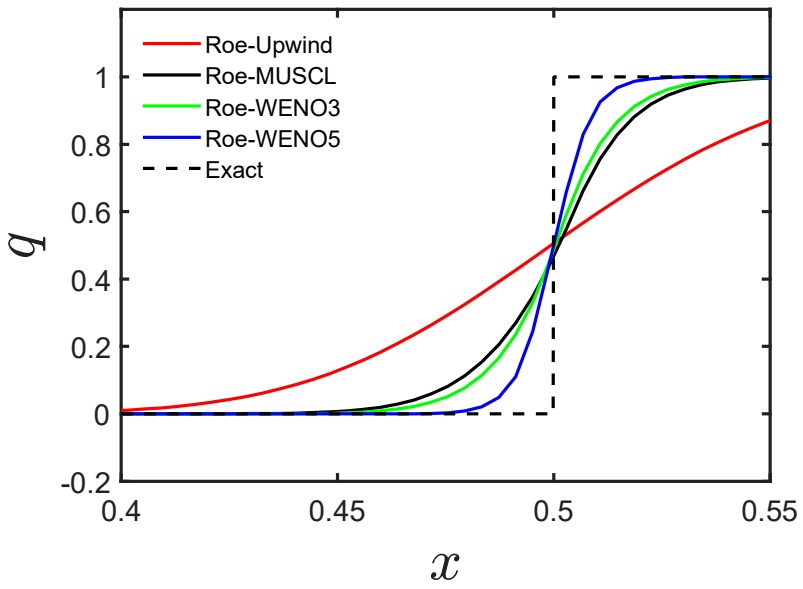

Figure 6.6: Roe Central Differencing for the CD Term with Upwind, MUSCL, WENO3, and WENO5 Reconstruction for the Diffusion Term for a 1-D Step

Figure 6.7 shows the results of the four CD schemes with WENO3 reconstruction for the diffusive term. The higher-order CD schemes (CD4 and CD6) are approximately identical to Roe's original flux. The CD2 scheme is not monotonic due to the reduced numerical diffusion with the higher-order reconstruction.

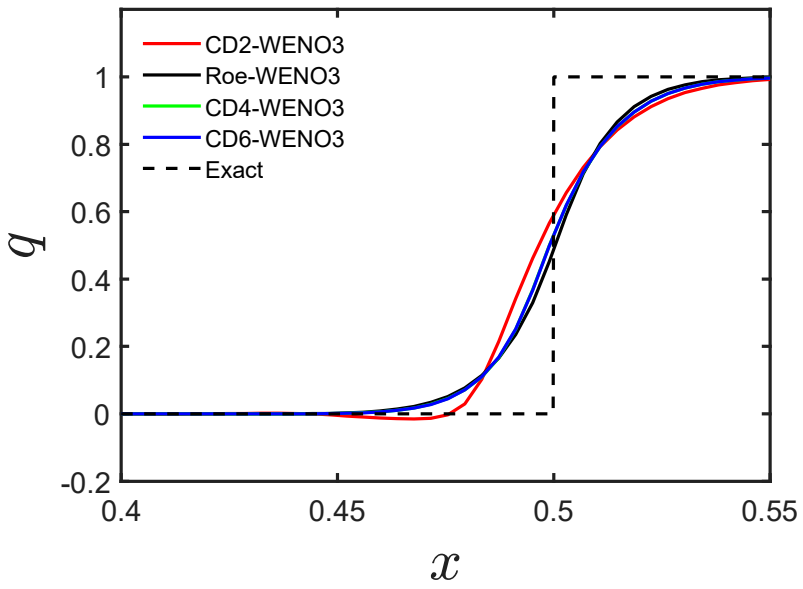

Figure 6.7: WENO3 Reconstruction for the Diffusion Term with CD2, CD4, CD6, and Roe Central Differencing for the CD term

Figure 6.8 shows the results of the four CD schemes with WENO5 reconstruction for the diffusive term. When using WENO5 reconstruction for the diffusion term, the differences between the different CD schemes become apparent. Roe's original CD term performs the best out of the four. The CD6 scheme is the most accurate scheme out of the four higher-order CD schemes. The CD2 scheme exhibits severe oscillations, as shown in Figure 6.8. The CD4 scheme has a numerical feature that is non-physical.

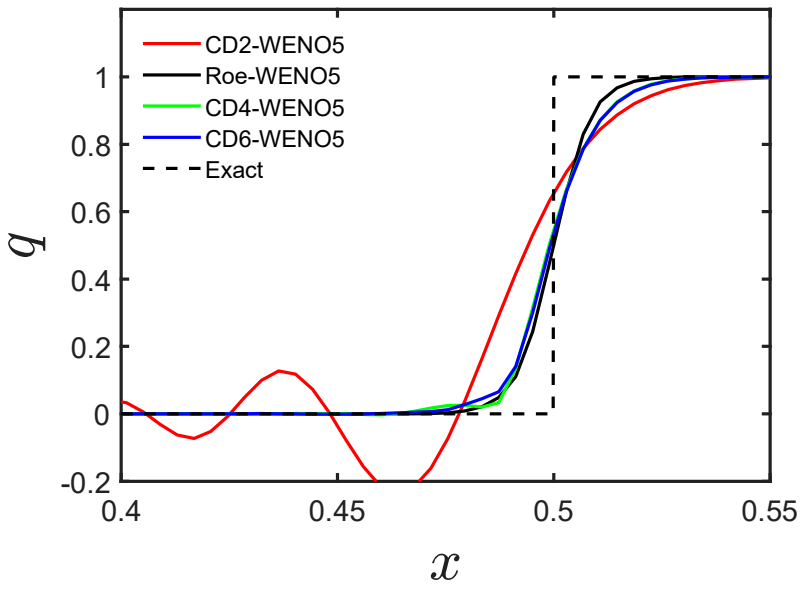

Figure 6.8: WENO5 Reconstruction for the Diffusion Term with CD2, CD4, CD6, and Roe Central Differencing for the CD term

6.2.3 1-D Half-Sine Wave

The second case tested consists of a 1-D half-sine wave advected with a constant velocity $a = 1$ on a uniform grid with the same cell spacing as the 1-D step case. The half-sine wave, given by Equation 6.3, is initially placed at $x = 0$ with out flow boundary conditions located at $x = -1$ and $x = 1$. The half-sine wave is advected until time $t = 50$ at which point it has moved a distance of 50.0. Figure 6.9 shows the initial condition of the 1-D half-sine wave. The solution to the 1-D half-sine wave gives an indication of a numerical scheme's numerical diffusion and phase lag.

$$q(x,0) = \alpha e^{-\beta x^2} \tag{6.3}$$

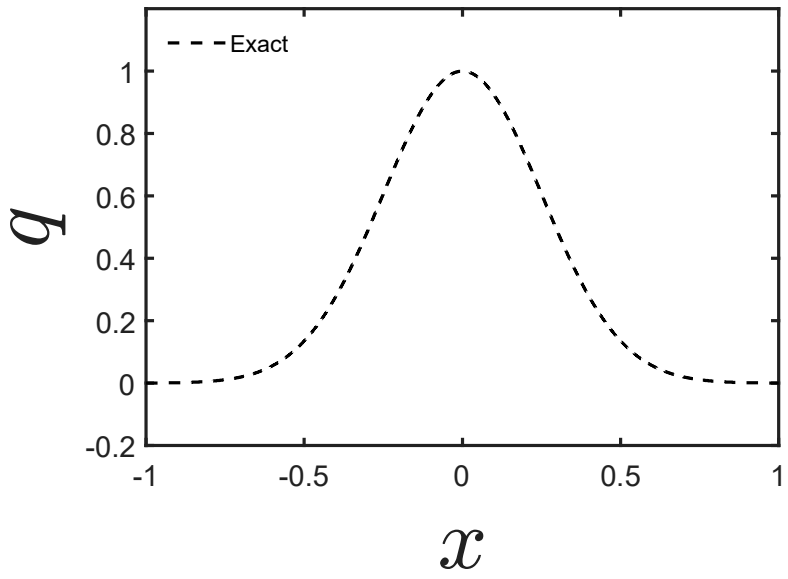

Figure 6.9: 1-D Half-Sine Wave Initial Setup

Equation 6.3 does not allow the sine wave to become negative therefore, the solution of the one-dimensional half-sine wave should not be negative. A resulting solution with a negative q is a result of the scheme. Additionally, an out-of-phase solution will lag behind or ahead of the analytical solution indicating how well a numerical scheme tracks flow features over time.

6.2.4 1-D Half-Sine Wave Results

Figure 6.10 shows the results for the four reconstruction schemes for the diffusive term with the CD2 scheme for the CD term. The upwind scheme has too much

numerical diffusion, erasing the shape of the half-sine wave. The MUSCL scheme demonstrates a significant amount of numerical diffusion although, the shape of the half-sine wave is discernible. The peak q value of 1 is reduced to a little under 0.7. The wave is dissipated more on the right side than the left. Also, the MUSCL scheme shows a slight phase lag as evident by the asymmetry on the right side.

The WENO3 scheme is also dissipative, but less so than the MUSCL scheme. The peak q value for the WENO3 scheme is approximately 0.8 and is more diffusive on the left side of the analytical solution. The WENO5 scheme keeps the shape of the half-sine wave, but experiences extreme phase lag to the left. There is also an oscillation on the left side leading to a negative q value. Lack of numerical viscosity generated from the diffusion term causes the solution to go negative.

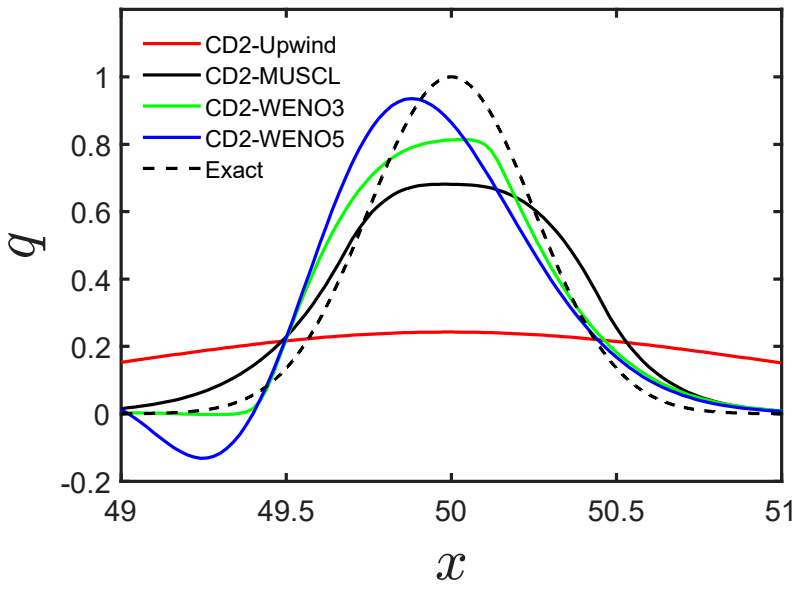

Figure 6.10: CD2 Central Differencing for the CD Term with Upwind, MUSCL, WENO3, and WENO5 Reconstruction for the Diffusion Term for a 1-D Half-Sine Wave

Figure 6.11 shows the results for the four reconstruction schemes for the diffusive term with the CD4 scheme for the CD term. The CD4 scheme has a lower dispersion error than the CD2 scheme. The amount of numerical diffusion is the same as the peaks of the waves remain essentially unchanged. Less phase lag is present in the WENO3 and WENO5 schemes. The MUSCL scheme appears to experience slightly more phase lag for CD4 than for CD2. The peak values of q remain largely unchanged for the MUSCL and WENO3 schemes, while the peak value for the WENO5 scheme is almost exactly equal to 1.

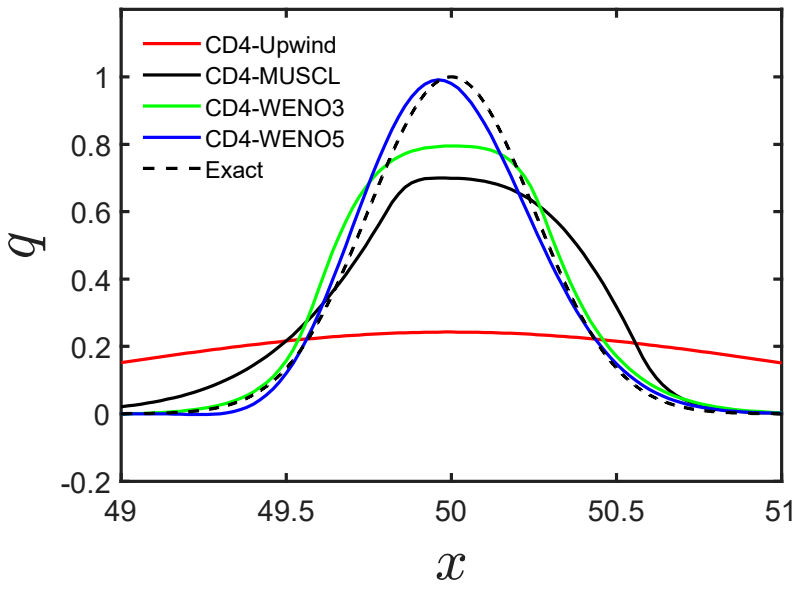

Figure 6.11: CD4 Central Differencing for the CD Term with Upwind, MUSCL, WENO3, and WENO5 Reconstruction for the Diffusion Term for a 1-D Half-Sine Wave

The values for the CD6 scheme are not shown, as the values for the CD4 scheme are identical. Consequently, increasing the order of accuracy for the CD term leads to negative returns. This is in contrast to the single discontinuity where increasing the CD scheme from fourth-order to sixth-order increased the accuracy of the WENO5 scheme.

Figure 6.12 shows the results for the four reconstruction schemes for the diffusive term with the Roe's original flux for the CD term. Roe's original CD flux was again better than CD2, CD4, and CD6. Both the MUSCL and WENO3 schemes show very little phase lag from the analytical solution. However, the maximum values of

q remain approximately the same between the MUSCL and WENO3 schemes. The WENO5 scheme shows no phase lag, but it overshoots the maximum value of q. This overshoot decreases with an increase in mesh density. The overshoot for the WENO5 scheme results from the diffusion term not providing enough numerical viscosity for the scheme to be TVD.

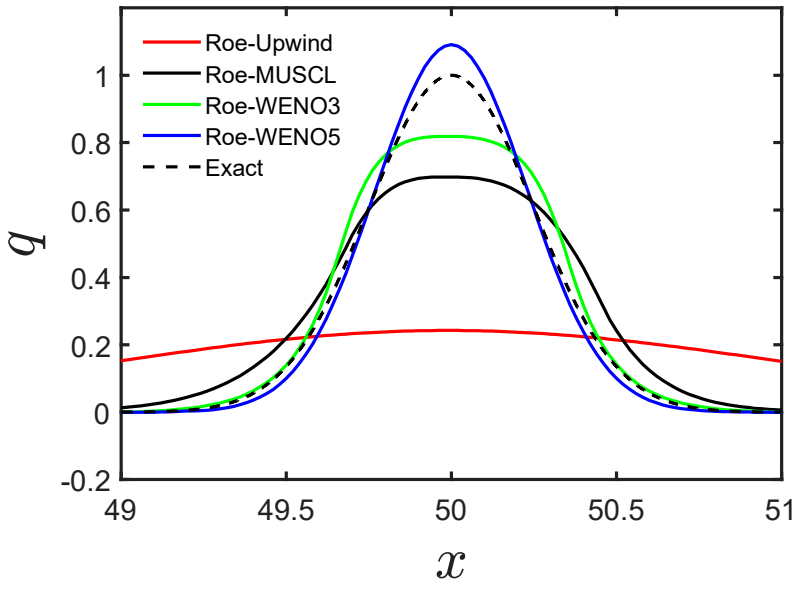

Figure 6.12: Roe Central Differencing for the CD Term with Upwind, MUSCL, WENO3, and WENO5 Reconstruction for the Diffusion Term for a 1-D Half-Sine Wave.

Figure 6.13 shows the results of WENO3 reconstruction for the diffusive term with CD2, CD4, CD6, and Roe central differencing for the CD term. The Roe CD is the most accurate central-differencing scheme followed by CD4 and CD6. All schemes present slight phase lag that is biased to the left side of the half-sine wave. The CD4

and CD6 schemes have no differences showing that a further reduction in dispersion error does not increase the accuracy.

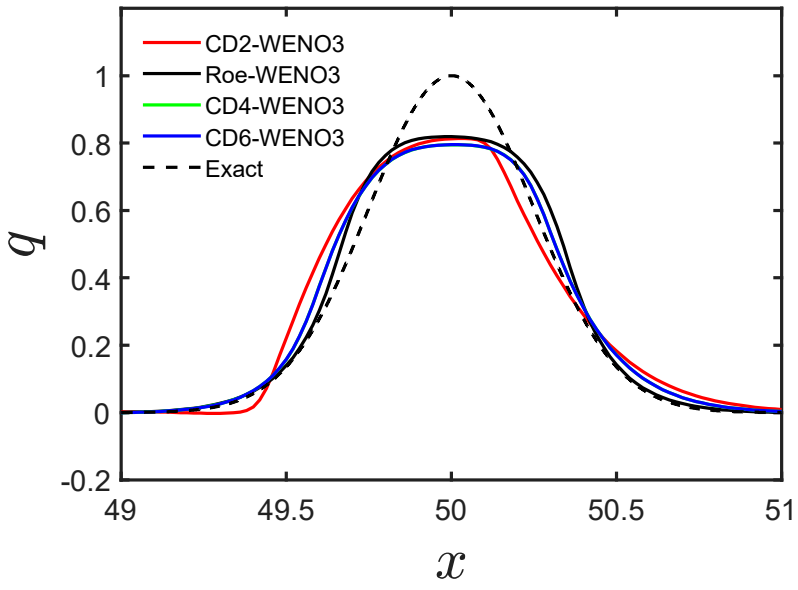

Figure 6.13: WENO3 Reconstruction for the Diffusion Term with CD2, CD4, CD6 and Roe Central Differencing for the CD Term for a 1-D Half-Sine Wave

Figure 6.14 shows the results of the WENO5 reconstruction for the diffusion term with the four CD schemes. Both the CD4 and CD6 schemes show the most accurate results. There is a slight bias to the left side indicating the CD2 scheme is not TVD as it shows severe phase lag. Roe's original flux overshoots the peak q value of the half-sine wave.

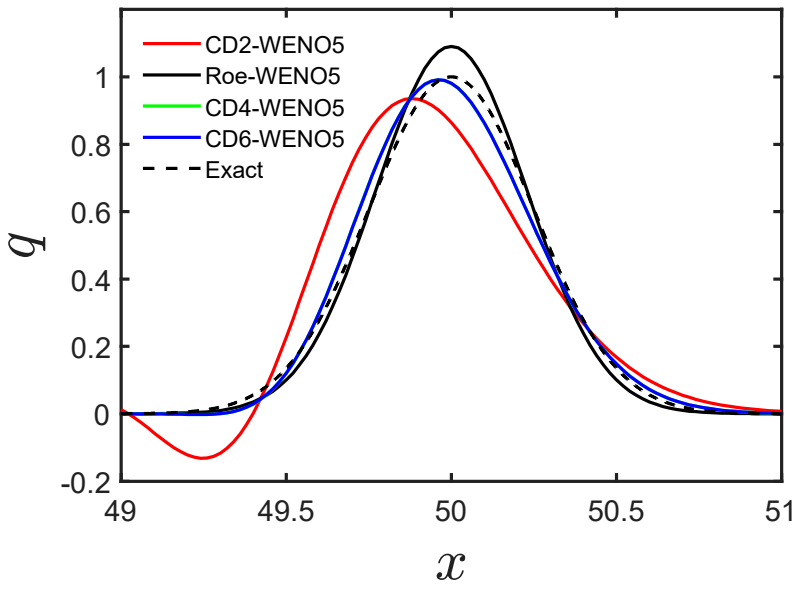

Figure 6.14: WENO5 Reconstruction for the Diffusion Term with CD2, CD4, CD6 and Roe Central Differencing for the CD Term for a 1-D Half-Sine Wave

6.2.5 1-D Square Wave

The third case tested consists of a 1-D square wave advected with a constant velocity $a = 1$ on a uniform grid with 512 cells. The half-sine wave has a value of $q = 0$ for $x < 0.3$ and $x > 0.7$, but is 1 when $x > 0.3$ and $x < 0.7$. The square wave is initially placed at $x = 0$ with out flow boundary conditions located at $x = 0$ and $x = 1.5$. The test conditions and the mesh spacing are the same as those used for the 1-D step. The square wave is advected until time $t = 0.5$ at which point it has moved a distance of 0.5. Figure 6.15 shows the initial condition of the 1-D square wave. The

solution to the 1-D square wave gives an indication of a numerical scheme's ability to handle consecutive shocks.

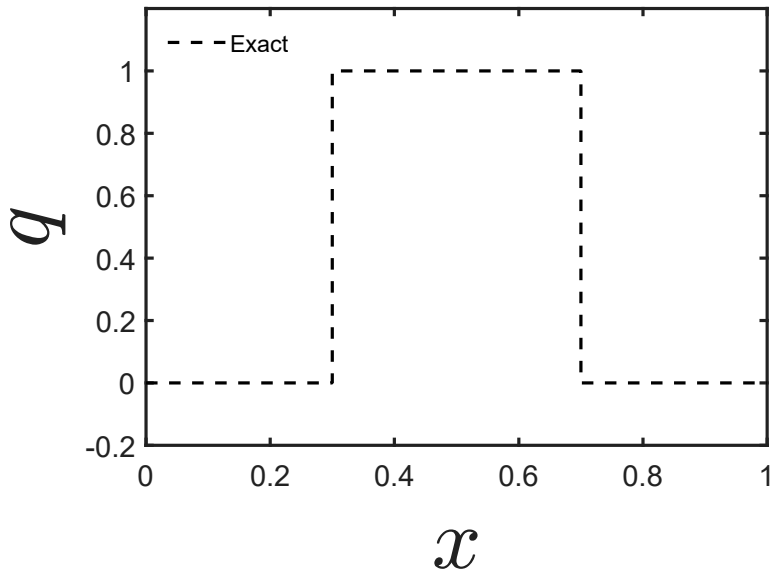

Figure 6.15: 1-D Square Wave Initial Setup

6.2.6 1-D Square Wave Results

Figure 6.16 shows the results for the four reconstruction schemes for the diffusive term with the CD2 scheme for the CD term. The CD2 scheme was TVD for the upwind scheme and the MUSCL scheme. However, for the WENO3 and WENO5 schemes, there are large oscillations near the discontinuities. This is a result of the diffusion term in Roe's CD not providing sufficient numerical viscosity for the amount of dispersion error present. For CD4 the WENO3 scheme no longer has the oscillations present when using CD2.

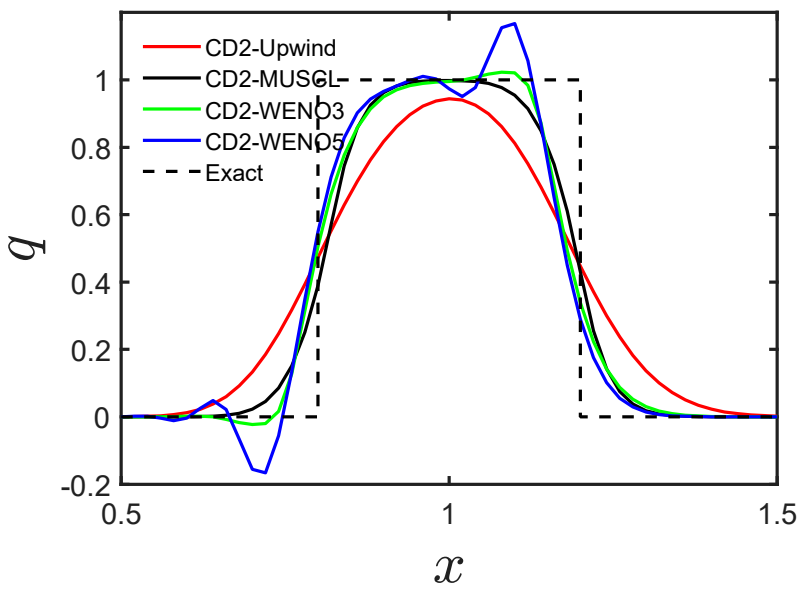

Figure 6.16: CD2 Central Differencing for the CD Term with Upwind, MUSCL, WENO3, and WENO5 Reconstruction for the Diffusion Term for a 1-D Square Wave

Figure 6.17 shows the results for the four reconstruction schemes for the diffusive term with the CD4 scheme for the CD term. The WENO5 scheme still has slight oscillations. The WENO3 scheme is TVD. The upwind and MUSCL schemes remain unchanged.

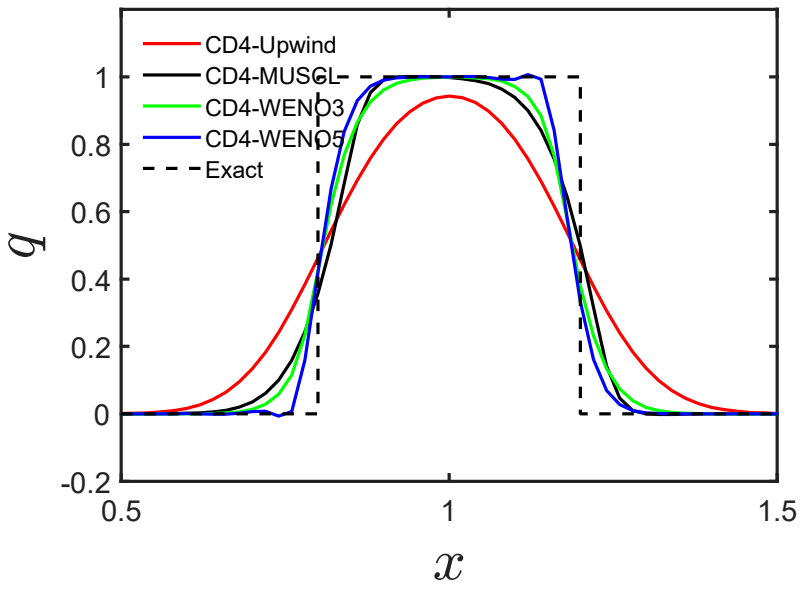

Figure 6.17: CD4 Central Differencing for the CD Term with Upwind, MUSCL, WENO3, and WENO5 Reconstruction for the Diffusion Term for a 1-D Square Wave

Figure 6.18 shows the results for the four reconstruction schemes for the diffusive term with the CD6 scheme for the CD term. Using the CD6 CD scheme fixes the oscillations in the WENO5 scheme making it TVD. The upwind, MUSCL, and WENO3 schemes remain unchanged from the CD4 scheme. This is a result of the dispersion error decreasing to a point that the diffusion term begins to dominate.

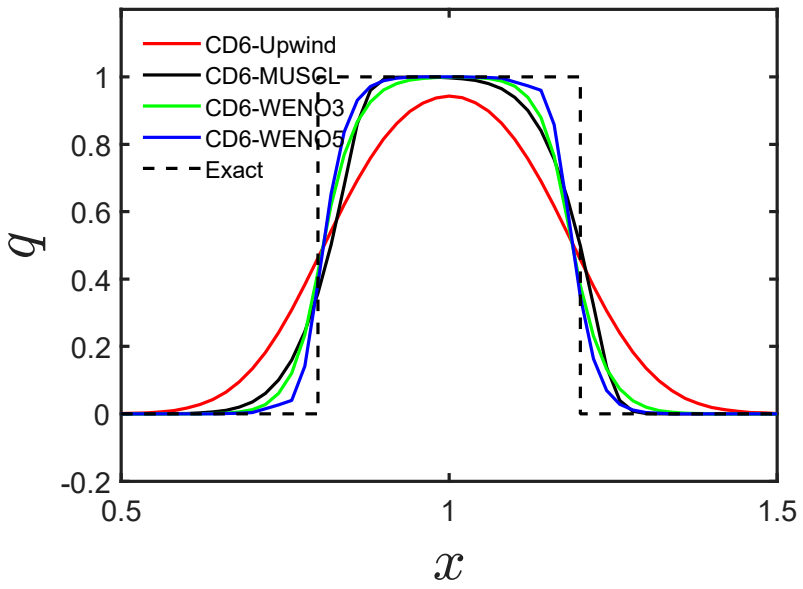

Figure 6.18: CD6 Central Differencing for the CD Term with Upwind, MUSCL, WENO3, and WENO5 Reconstruction for the Diffusion Term for a 1-D Square Wave

Figure 6.19 shows the Roe's CD for the 1-D square wave. The Roe CD works well for every reconstruction method. All cases are TVD with crisp representations of the discontinuities.

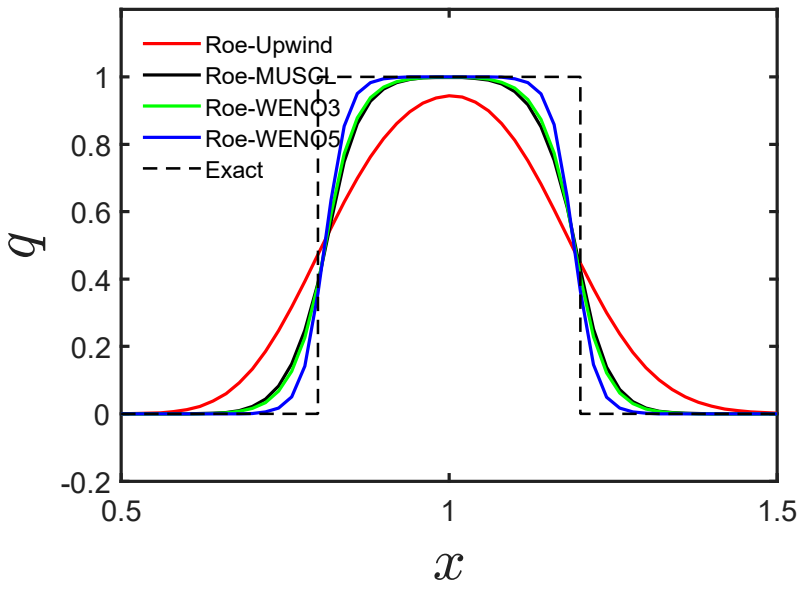

Figure 6.19: Roe Central Differencing for the CD Term with Upwind, MUSCL, WENO3, and WENO5 Reconstruction for the Diffusion Term for a 1-D Square Wave

Figure 6.20 shows the results of the four CD schemes with WENO3 reconstruction for the diffusive term. Every CD scheme besides CD2 are TVD and essentially identical. CD2 has oscillations resulting from the high dispersion error.

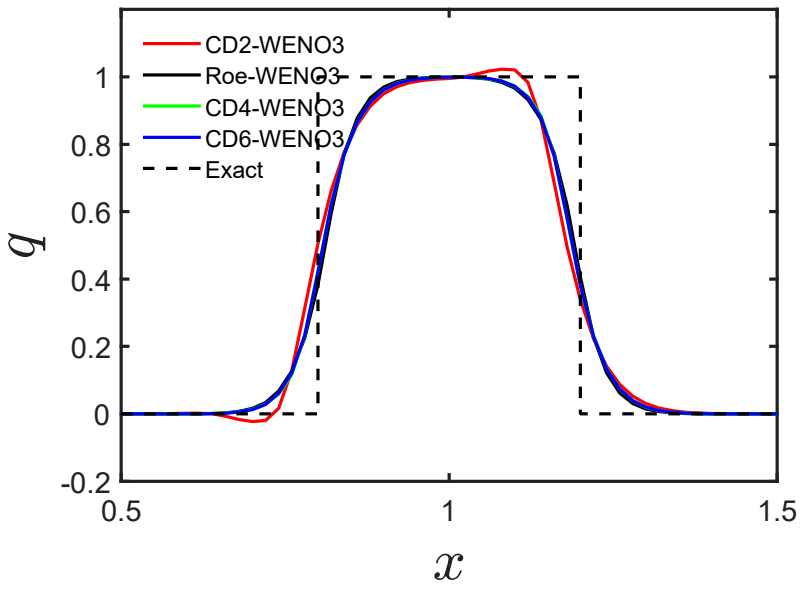

Figure 6.20: WENO3 Reconstruction for the Diffusion Term with CD2, CD4, CD6 and Roe Central Differencing for the CD Term for a 1-D Square Wave

Figure 6.21 shows the results of the four CD schemes with WENO5 reconstruction for the diffusive term. The Roe and CD6 schemes are nearly identical and TVD. The CD4 scheme has some small oscillations near the discontinuity. The CD2 scheme has more pronounced oscillations. The CD4 scheme has small oscillations as a result of the reduced numerical viscosity of the WENO5 scheme.

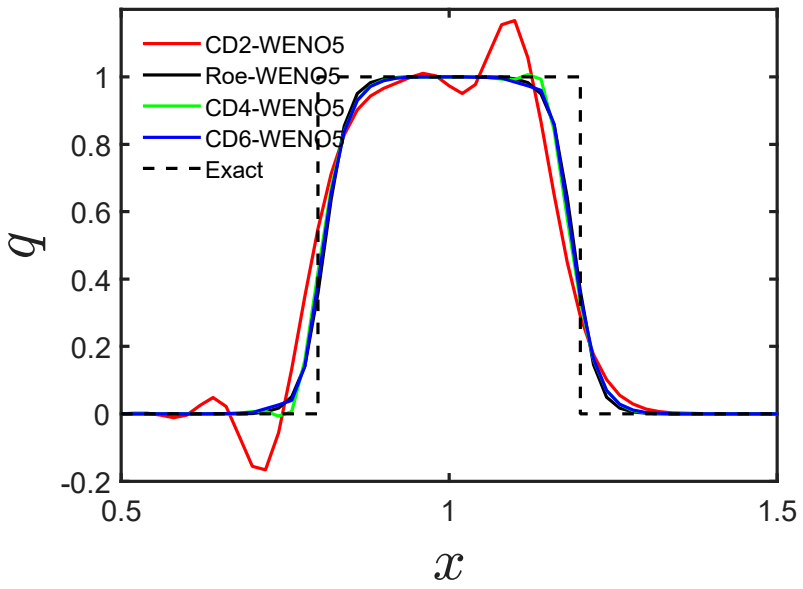

Figure 6.21: WENO5 Reconstruction for the Diffusion Term with CD2, CD4, CD6 and Roe Central Differencing for the CD Term for a 1-D Square Wave

6.3 Two-Dimensional Cases

The two-dimensional linear advection equation shows how a numerical scheme acts in two dimensions. Numerical schemes are usually derived for a one-dimensional case and then extended to multiple dimensions. It is important to study the same equation in multiple dimensions to ensure that the scheme behaves in the same manner as for one dimension. Here the two-dimensional linear advection is studied.

The two-dimensional linear advection equation has one temporal dimension, and two spatial dimensions. Unlike the one-dimensional case, the two-dimensional case has multiple boundary conditions. There are both inflow and outflow boundaries.

The inflow boundary conditions change depending on the test case. Equation 6.4 gives the two-dimensional linear advection equation.

$$\frac{\partial q}{\partial t} + a\frac{\partial q}{\partial x} + a\frac{\partial q}{\partial y} = 0 \tag{6.4}$$

Three test cases, taken from [26], are studied: a step profile, a sinusoidal profile, and a two-step profile. These test cases are based on the test cases for the one-dimensional analysis so direct comparisons can be made. Profiles of the advected scalar are presented for the four CD schemes with the various reconstruction schemes.

6.3.1 2-D Step

The first 2-D case tested consists of a 2-D step function advected with a constant velocity $a = (1, 1)$ on a uniform grid with 46x46 cells. The inflow boundary condition at $x = 0$ gives a q value of 1. The inflow boundary condition at $y = 0$ gives a q value of 0. Outflow boundary conditions are located at $x = 1$ and $y = 1$. The test conditions and the mesh spacing are the same as those used in [26]. The initial q values are set to zero throughout the domain. The solution is run for a time of $t = 1.5$ when steady state is reached. Figure 6.22 shows the boundary conditions of the 2-D step. A slice of the advective quantity q is taken at $y = 0.8$. The two-dimensional step demonstrates a numerical scheme's ability to resolve a shock in two dimensions.

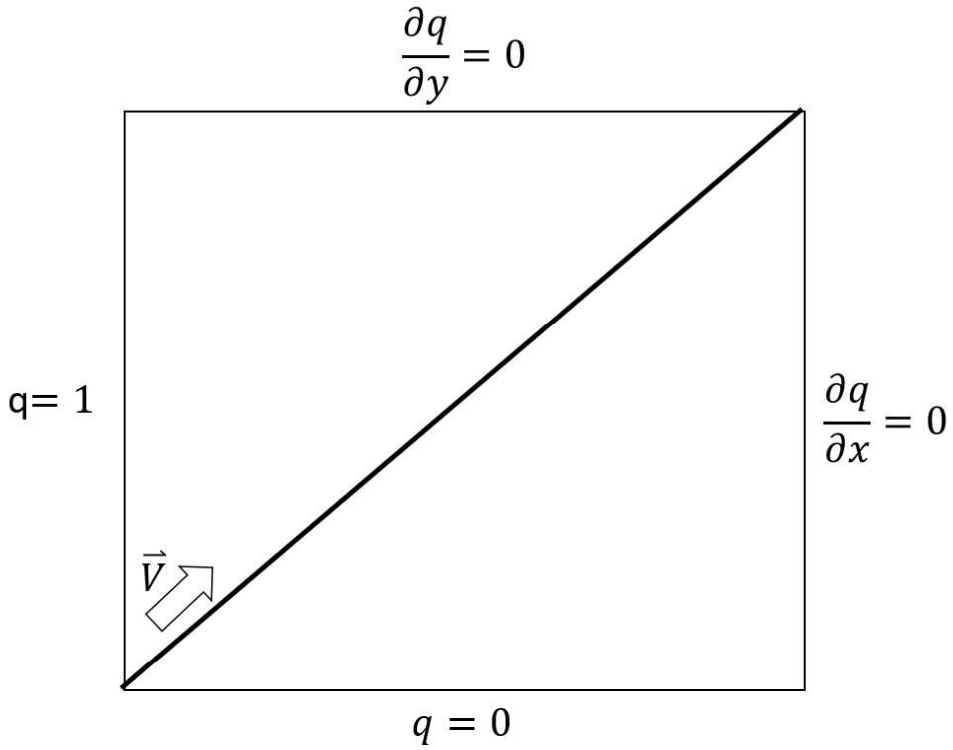

Figure 6.22: 2-D Step Initial Setup

6.3.2 2-D Step Results

Figure 6.23 shows the CD2 scheme for the CD term with upwind, MUSCL, WENO3, and WENO5 reconstruction for the diffusive term. The results for the 2-D step differ from the 1-D step. It can clearly be seen that the upwind, MUSCL, and WENO3 schemes are non-oscillatory and TVD. The WENO5 scheme has a large oscillation near the discontinuity.

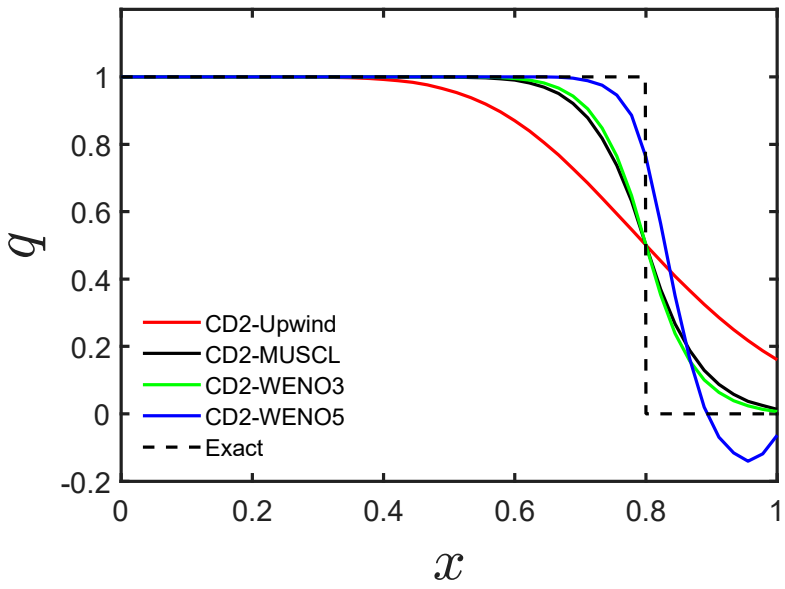

Figure 6.23: CD2 Central Differencing for the CD Term with Upwind, MUSCL, WENO3, and WENO5 Reconstruction for the Diffusion Term for a 2-D Step

Figure 6.24 shows the CD4 scheme for the CD term with upwind, MUSCL, WENO3, and WENO5 reconstruction for the diffusive term. The results of the CD4 scheme for the upwind, MUSCL, and WENO3 reconstruction schemes are identical to the CD2 scheme. In fact, using the CD2, CD4, CD6, and Roe's CD schemes have little effect on the solution for upwind, MUSCL, and WENO3 reconstructions due to the diffusion term in Roe's flux being dominant. Therefore, a reduction in dispersion error does not overcome the inherent numerical diffusion in the Roe scheme for this test case.

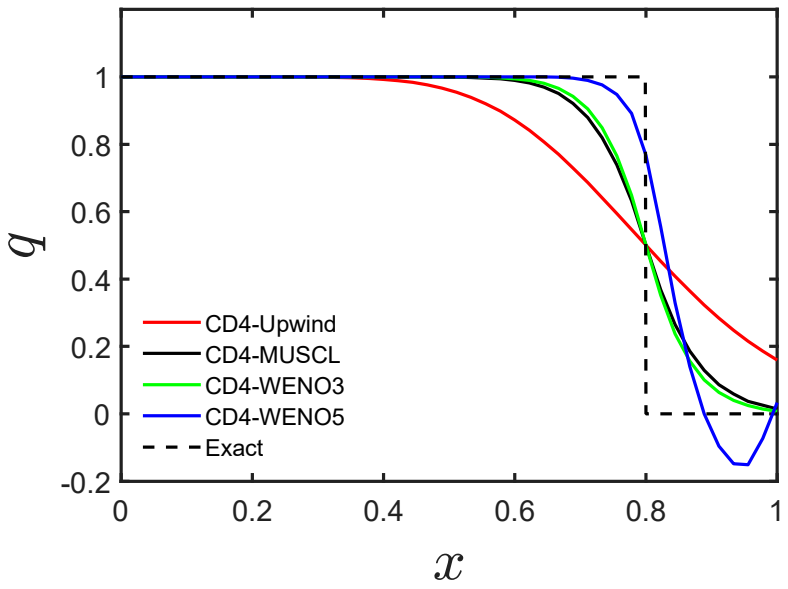

Figure 6.24: CD4 Central Differencing for the CD Term with Upwind, MUSCL, WENO3, and WENO5 Reconstruction for the Diffusion Term for a 2-D Step

Figure 6.25 shows Roe's original CD scheme with upwind, MUSCL, WENO3, and WENO5 reconstruction for the diffusive terms. Roe's CD scheme produced the best result with WENO5 reconstruction. The upwind, MUSCL, and WENO3 schemes are largely unaffected by the choice of CD used. The Roe CD flux is the only scheme that remains TVD. None of the other schemes properly resolve the discontinuity. The CD4 scheme is almost identical to the CD6 scheme indicating that an increase in the order of accuracy does not resolve the shock more accurately. The reason for this is that the WENO5 scheme is less diffusive than the other reconstruction schemes. This means that there is not enough numerical diffusion to rid the solution of oscillations.

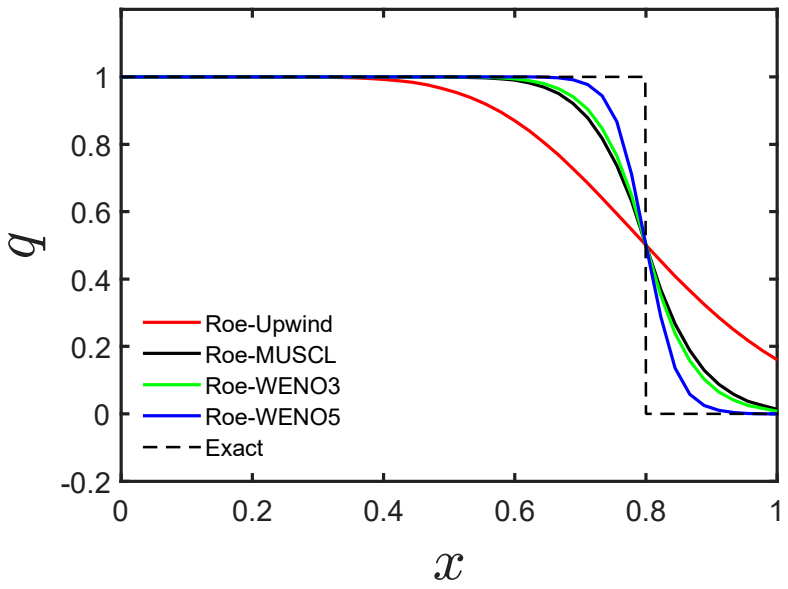

Figure 6.25: Roe Central Differencing for the CD Term with Upwind, MUSCL, WENO3, and WENO5 Reconstruction for the Diffusion Term for a 2-D Step

Figure 6.26 shows the results of the four CD schemes with WENO3 reconstruction for the diffusive term. All schemes are TVD and identical resulting from the diffusion term dominating the flux calculation. This feature is part of the WENO3 scheme for the two-dimensional linear advection equation.

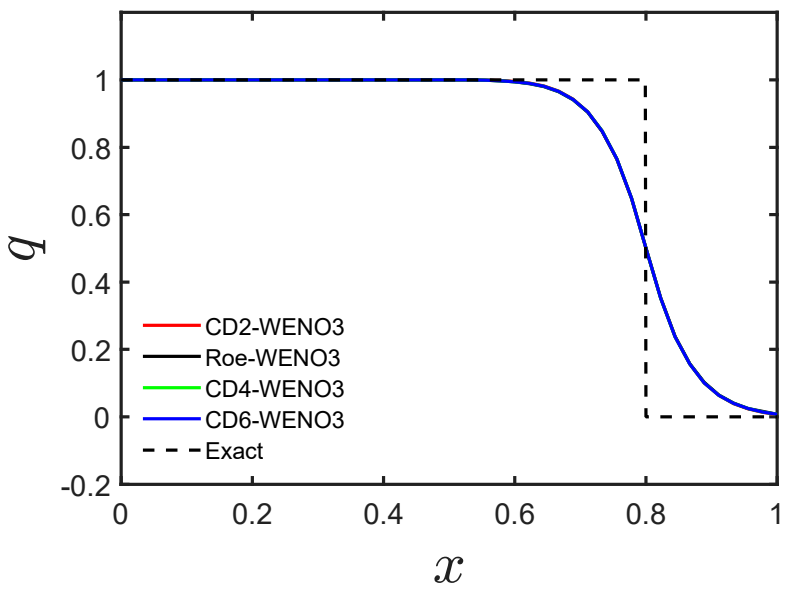

Figure 6.26: WENO3 Reconstruction for the Diffusion Term with CD2, CD4, CD6 and Roe Central Differencing for the CD Term for a 2-D Step

Figure 6.27 shows the results of the four CD schemes with WENO5 reconstruction for the diffusive term. Only Roe's original CD flux term gives a TVD solution. The other CD schemes have an oscillation near the discontinuity indicating improvements to the order of the CD does not alleviate the oscillation.

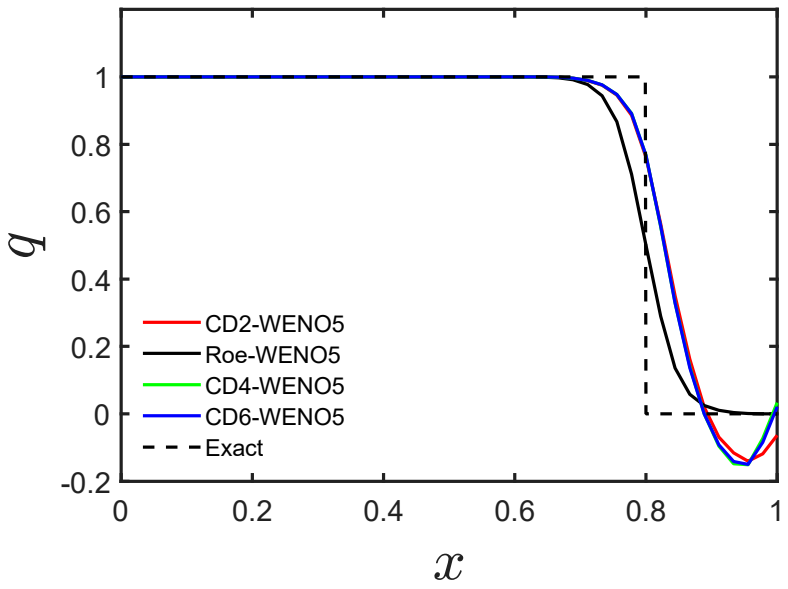

Figure 6.27: WENO5 Reconstruction for the Diffusion Term with CD2, CD4, CD6 and Roe Central Differencing for the CD Term for a 2-D Step

6.3.3 Two-Dimensional Half-Sine Wave

The second 2-D case tested consists of a 2-D half-sine wave advected with a constant velocity $a = (1, 1)$ on a uniform grid with 46x46 cells. The $x = 0$ inflow boundary is set to 0 for $y > 0.3414$, and when $y < 0.3414$ the inflow q value is given by Equation 6.5. The inflow boundary condition at $y = 0$ gives a q value of 0. Outflow boundary conditions are located at $x = 1$ and $y = 1$. The test conditions and the mesh spacing are the same as those used in [26]. The initial q values are set to zero throughout the domain. The solution is run for a time of $t = 1.5$ when steady state is reached. Figure 6.28 shows the boundary conditions of the 2-D half-sine wave. A

slice of the advective quantity q is taken at $y = 0.8$. The 2-D half-sine wave shows phase lag in the numerical solution as well as giving some insights into the dissipation of the various schemes.

$$q = sin[\frac{\pi}{2} max(1 - \frac{\mid y - 0.1707 \mid}{0.1707}, 0)] \tag{6.5}$$

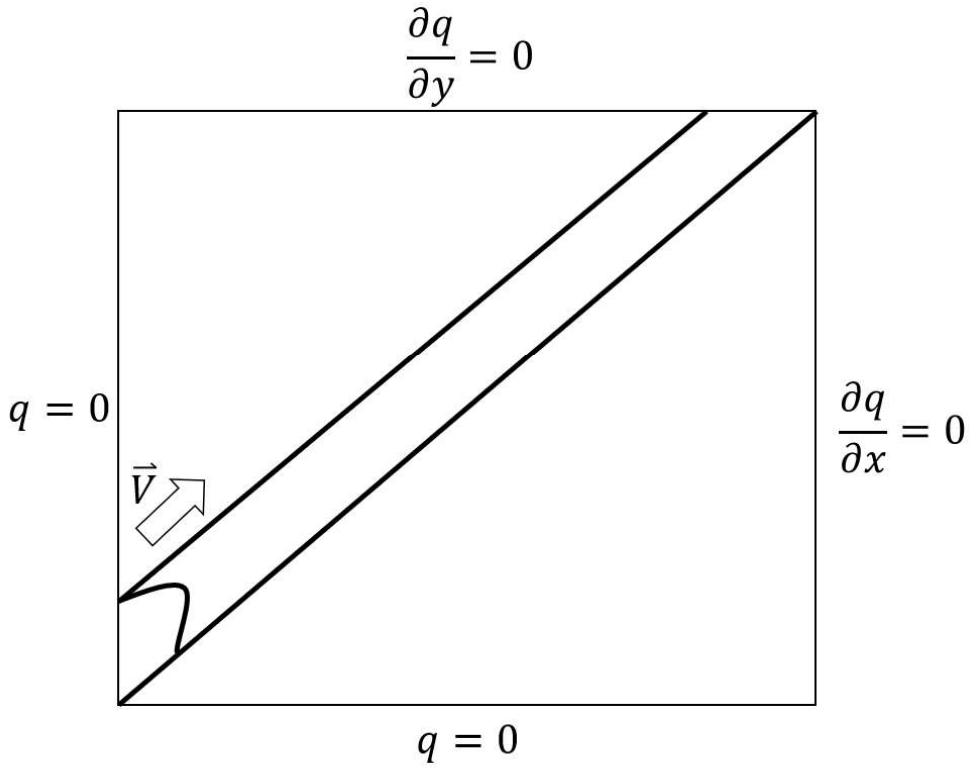

Figure 6.28: 2-D Half-Sine Wave Initial Setup

6.3.4 Two-Dimensional Half-Sine Wave Results

Figure 6.29 shows the CD2 scheme for the CD term with upwind, MUSCL, WENO3, and WENO5 reconstruction for the diffusive term. Similarly to the 2-

D step case, changing the CD flux has little effect for the upwind, MUSCL, and WENO3 reconstructions. The half-sine wave shape is changed due to numerical diffusion for the upwind, MUSCL, and WENO3 schemes. A slight phase lag appears as the peak value of the half-sine wave with the numerical schemes biasing to the right of the analytical solution. As with the 1-D cases, the WENO5 method overshoots the analytical solution and has more phase lag than the other reconstruction methods.

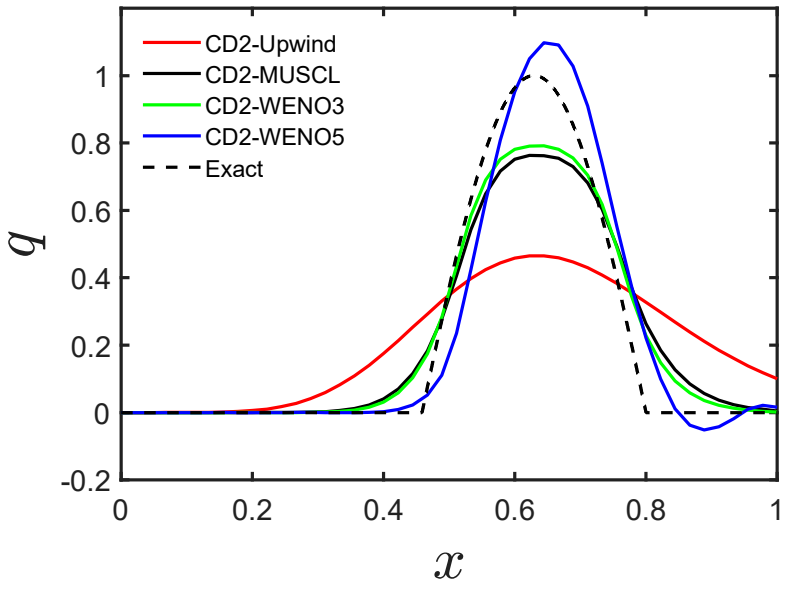

Figure 6.29: CD2 Central Differencing for the CD Term with Upwind, MUSCL, WENO3, and WENO5 Reconstruction for the Diffusion Term for a 2-D Half-Sine Wave

Figure 6.30 shows the CD4 scheme for the CD term with upwind, MUSCL, WENO3, and WENO5 reconstruction for the diffusive term. Increasing the order of the CD scheme from second to fourth does not change the answer of the solution.

Similarly to the 2-D step, using the CD2, CD4, CD6, and Roe's CD schemes have little effect on the solution for upwind, MUSCL, and WENO3 reconstructions due to the diffusion term in Roe's flux being dominant. Therefore, a reduction in dispersion error does not overcome the inherent numerical diffusion in the Roe scheme for this test case.

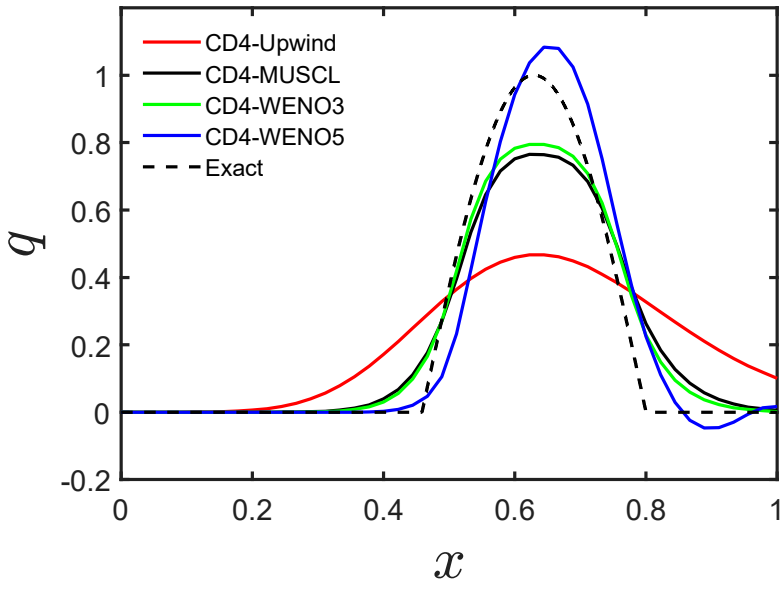

Figure 6.30: CD4 Central Differencing for the CD Term with Upwind, MUSCL, WENO3, and WENO5 Reconstruction for the Diffusion Term for a 2-D Half-Sine Wave

Figure 6.31 shows Roe's original CD scheme for the CD term with upwind, MUSCL, WENO3, and WENO5 reconstruction for the diffusive term. Using Roe's CD method alleviates the overshoot and decreases the amount of phase lag.

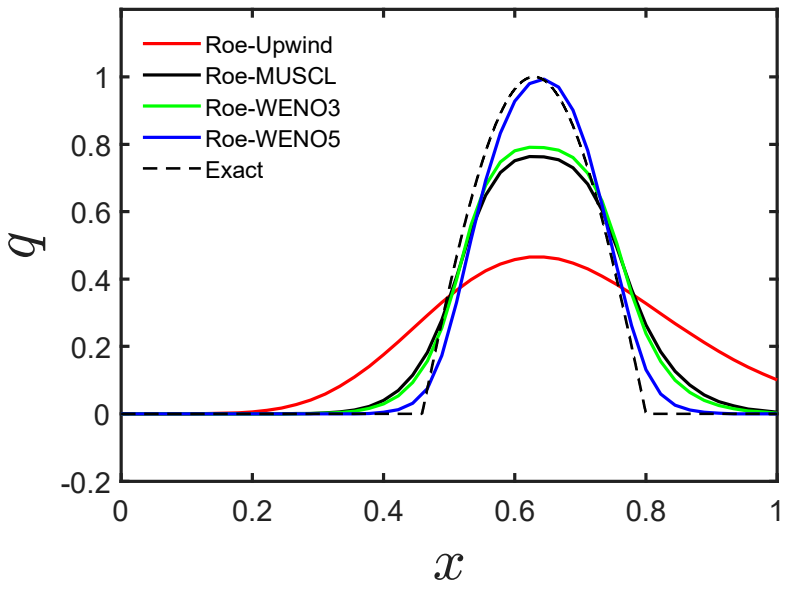

Figure 6.31: Roe Central Differencing for the CD Term with Upwind, MUSCL, WENO3, and WENO5 Reconstruction for the Diffusion Term for a 2-D Half-Sine Wave.

Figure 6.32 shows the results of the four CD schemes with WENO3 reconstruction for the diffusive term. Similarly to the 2-D step, the schemes are essentially identical. A slight phase lag biased to the right is present.

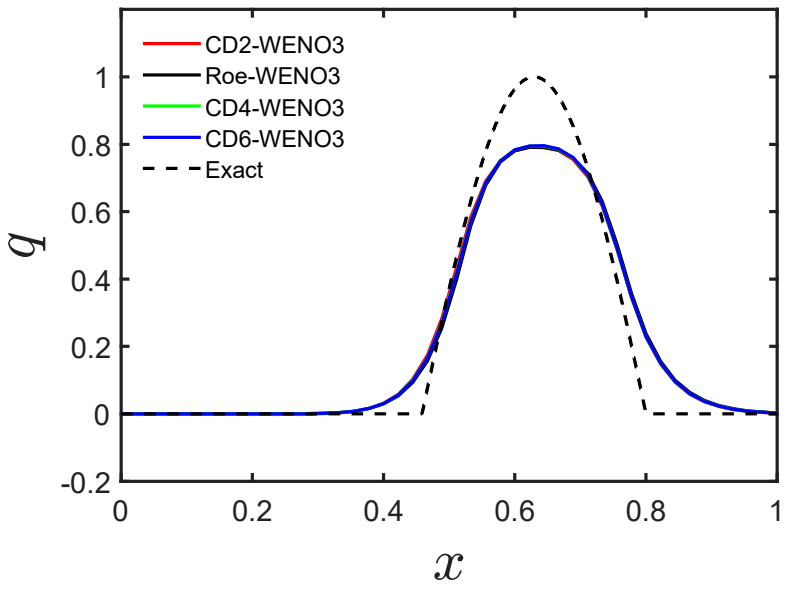

Figure 6.32: WENO3 Reconstruction for the Diffusion Term with CD2, CD4, CD6 and Roe Central Differencing for the CD Term for a 2-D Half-Sine Wave

Figure 6.33 shows the results of the four CD schemes with WENO5 reconstruction for the diffusive term. Only the Roe flux is TVD. All of the schemes have phase lag biased to the right and overshoot the exact solution. The higher-order CD schemes are not TVD and increasing the order of the CD does not reduce the oscillation.

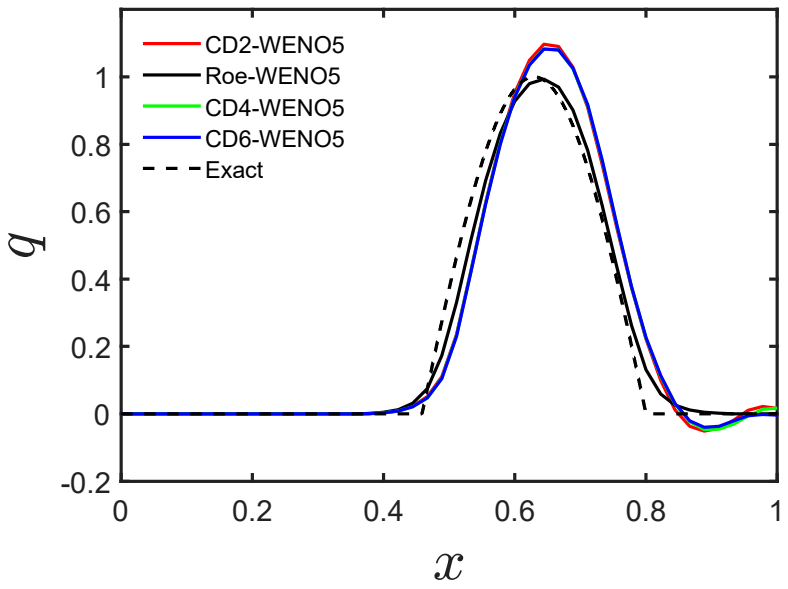

Figure 6.33: WENO5 Reconstruction for the Diffusion Term with CD2, CD4, CD6 and Roe Central Differencing for the CD Term for a 1-D Half-Sine Wave

6.3.5 Two-Dimensional Square Wave

The third 2-D test case consists of a 2-D square wave advected with a constant velocity $a = (1, 1)$ on a uniform grid with 46x46 cells. The $y = 1$ boundary is an inflow where q is set to zero. At the $x = 0$ inflow boundary, q is set to 1 when $x < 0.3$, and q is set to 0 when $x > 0.3$ creating a double-step profile that is similar to the square wave in one dimension. Outflow boundary conditions are located at $x = 1$ and $y = 1$. The test conditions and the mesh spacing are the same as those used in [26]. The initial q values are set to zero throughout the domain. The solution is run for a time of $t = 1.5$ when steady state is reached. Figure 6.34 shows the boundary

conditions of the 2-D half sine wave. A slice of the advective quantity q is taken at $y = 0.8$. The two-dimensional square wave shows how each scheme handles multiple discontinuities.

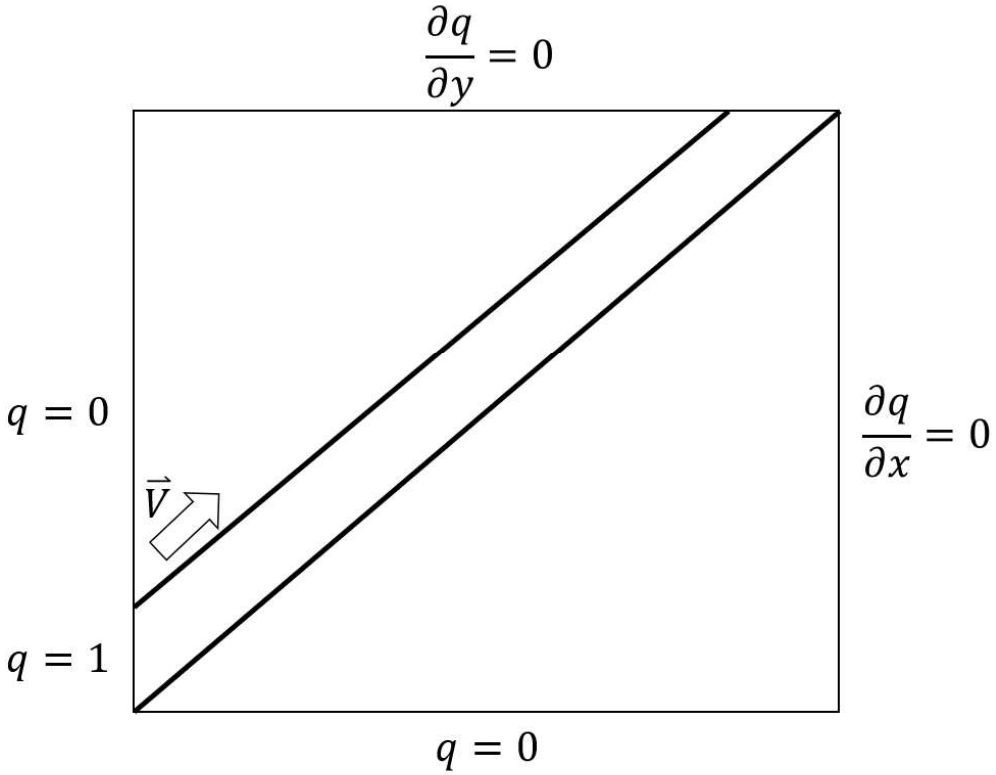

Figure 6.34: 2-D Square Wave Initial Setup

6.3.6 Two-Dimensional Square Wave Results

Figure 6.35 shows the CD2 scheme for the CD term with upwind, MUSCL, WENO3, and WENO5 reconstruction for the diffusive term. The results for the two-dimensional square wave are similar to the other two-dimensional test cases. The solution does not change depending on the choice of central differencing for

the upwind, MUSCL, and WENO3 schemes. All CD2 schemes are TVD with the exception of the WENO5 reconstruction.

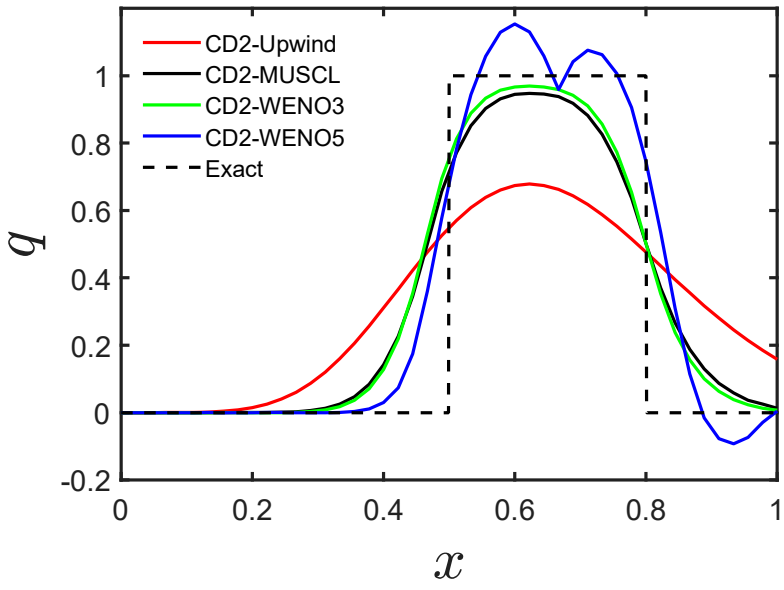

Figure 6.35: CD2 Central Differencing for the CD Term with Upwind, MUSCL, WENO3, and WENO5 Reconstruction for the Diffusion Term for a 2-D Square Wave

Figure 6.36 shows the CD4 scheme for the CD term with upwind, MUSCL, WENO3, and WENO5 reconstruction for the diffusive term. With the upwind, MUSCL, and WENO3 reconstruction schemes, increasing the accuracy of the CD term does not change the answer of the solution. The CD4 scheme with WENO5 reconstruction reduces the peak of the oscillation slightly, but the Gibbs phenomena is still prominent. The results for the CD6 scheme are identical to the CD4 scheme and, therefore, will not be shown.

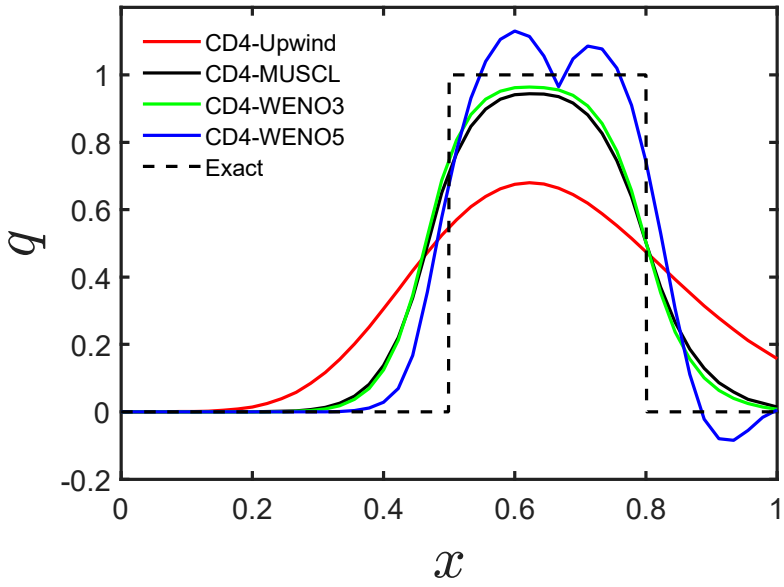

Figure 6.36: CD4 Central Differencing for the CD Term with Upwind, MUSCL, WENO3, and WENO5 Reconstruction for the Diffusion Term for a 2-D Square Wave

Figure 6.37 shows Roe's original CD scheme for the CD term with upwind, MUSCL, WENO3, AND WENO5 reconstruction for the diffusive term. Using Roe's CD has increasing ability to capture the square wave as the order of accuracy of the reconstruction scheme increases.

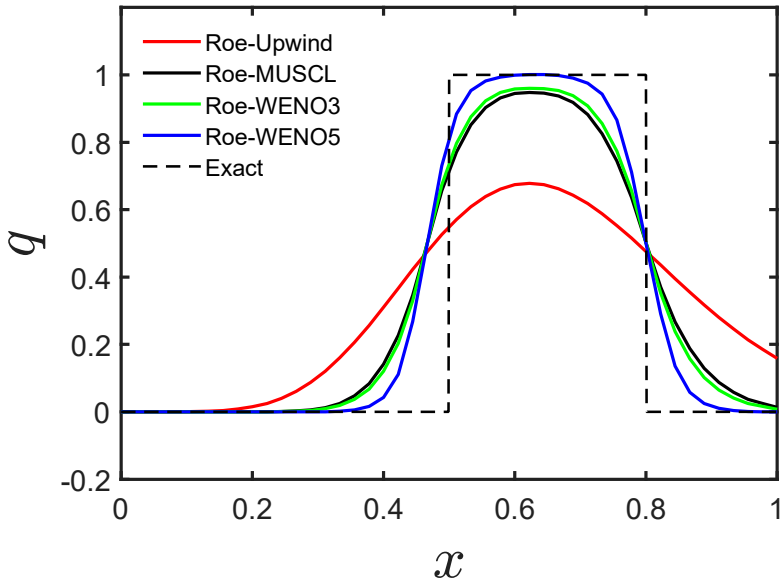

Figure 6.37: Roe Central Differencing for the CD Term with Upwind, MUSCL, WENO3, and WENO5 Reconstruction for the Diffusion Term for a 2-D Square Wave.

Figure 6.38 shows the results of the four CD schemes with WENO3 reconstruction for the diffusive term. Similar to the 2-D step and 2-D half-sine wave, the various CD schemes are essentially identical with a slight bias to the right.

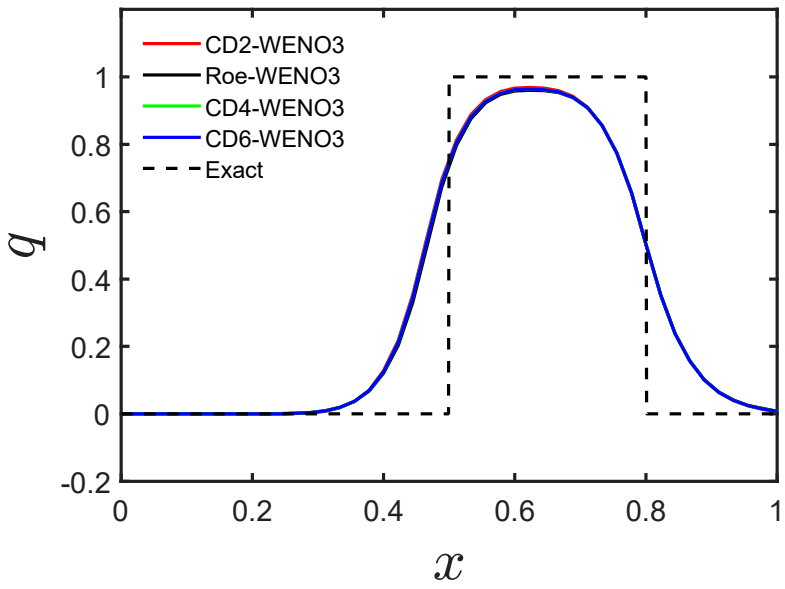

Figure 6.38: WENO3 Reconstruction for the Diffusion Term with CD2, CD4, CD6 and Roe Central Differencing for the CD Term for a 2-D Square Wave

Figure 6.39 shows the results of the four CD schemes with WENO5 reconstruction for the diffusive term. The WENO5 reconstruction has severe oscillations for the CD2, CD4, and CD6 fluxes. Only Roe's flux was TVD. Improving the order of the CD scheme appears to reduce the magnitude of the oscillations slightly. However, it is not enough to rid the solution of the oscillations completely. This is attributed to the diffusive term in Roe's flux not providing enough numerical viscosity for the higher-order scheme.

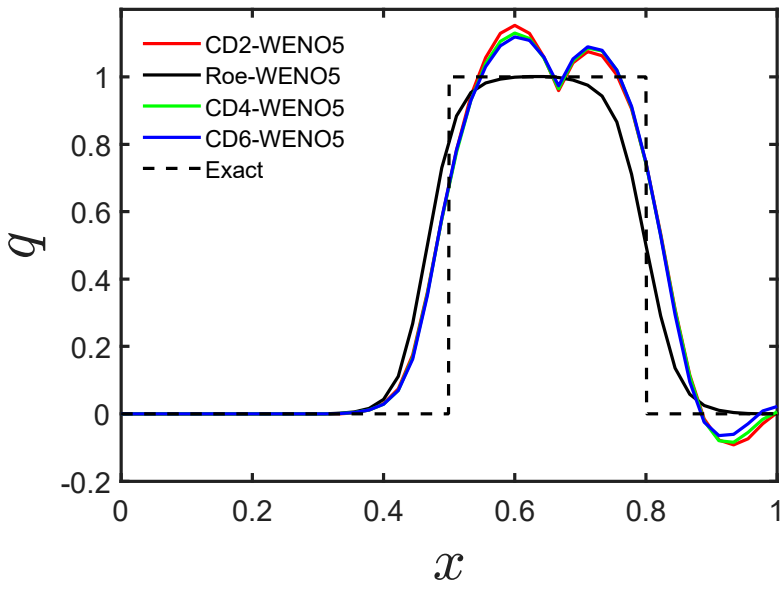

Figure 6.39: WENO5 Reconstruction for the Diffusion Term with CD2, CD4, CD6 and Roe Central Differencing for the CD Term for a 2-D Square Wave

CHAPTER 7

CONCLUSIONS

For all test cases, the WENO5 reconstruction is the most accurate and the best at capturing the solution features. The accuracy of the tested schemes show a direct correlation to their order of accuracy. The upwind scheme is the most diffusive and is only first-order accurate, the MUSCL scheme is second-order accurate, and WENO3 scheme is third-order accurate, and the WENO5 scheme is fifth-order accurate. This shows a direct correlation to the order of accuracy and the ability to capture flow features.

7.1 One-Dimensional Results

In one-dimension, the MUSCL scheme is largely unaffected by the choice of the CD flux and agrees with literature concerning the scheme [5]. The WENO3 scheme displays some unwanted features when the CD is second-order accurate. If the CD is increased to fourth-order accuracy, then the WENO3 scheme closely resembles Roe's CD. However, increasing the order of the CD to sixth order does not increase the accuracy of the WENO3 scheme.

The WENO5 scheme displays large oscillations at extrema for the CD2 flux. Increasing the accuracy of the CD flux to fourth order was insufficient to rid the solution of spurious oscillations. The CD had to be of sixth-order accuracy to obtain a solution of similar accuracy to Roe's original flux. As a result of the reduced numerical viscosity of the WENO5 scheme compared to the MUSCL and WENO3 schemes, the WENO5 scheme was prone to oscillations near strong gradients. This implies that Roe's flux could be replaced with a second-order accurate scheme for MUSCL, a fourth-order accurate scheme for WENO3, and a sixth-order accurate scheme for WENO5. This study showed using higher-order central differencing reduces dispersion error and increases the accuracy of the solution.

7.2 Two-Dimensional Results

The results from the two-dimensional linear advection contradicted some of the findings from the one-dimensional test cases. For the upwind, MUSCL, and WENO3 schemes, the choice of the CD flux had no effect on the solution. This implies that lower-order schemes have enough built-in diffusion to allow the second term in Roe's flux to dominate. Reducing dispersion error has little effect on the flux. However, the WENO5 scheme is more dependent on the choice of central differencing.

For all of the two-dimensional cases only Roe's original flux was TVD for the WENO5 scheme. The other CD flux's had oscillations at the extrema. This implies the WENO5 scheme has less diffusion and the choice of CD flux is more important for evaluating the flux. Reduction of dispersion has a positive effect on the schemes ability to accurately capture flow features. There were negative returns

when increasing the order of accuracy of the CD flux indicating that future increases to the order of accuracy to the CD flux will not result in a solution more accurate than Roe's original flux. Therefore, a TVD solution using WENO5 reconstruction is required to use Roe's CD flux.

7.3 Higher-Order Central Differencing

In one-dimension, increasing the order of accuracy of the CD term in Roe's flux increased the accuracy of the scheme. This is a result of the reduction of dispersion error. However, this is evident only when higher-order reconstruction is used on the diffusion term. The two-dimensional test cases showed that increasing the order of accuracy of the CD flux had a limited effect on the solution.

An unexpected discovery was made for the two-dimensional problem. The second-order CD gave the same accuracy as Roe's CD flux. This indicates reduced computation time for a system with more equations where recalculating the flux for every cell face is possible. For example, the flux of the Euler equations in three dimensions is a 5x1 vector. This could lead to less overhead cost and thus faster run times.

7.4 Future Work

The use of the CD2 and CD4 schemes need to be extended to more complicated problems. For example, the one-dimensional Euler equations for a shock tube is the next logical step. If the CD2 scheme produces the same solution for the Roe's original scheme, then it could lead to reduced computation time. Investigating

two-dimensional gas dynamic problems would be the next step. If the CD2 scheme succeeds, then application to the Navier-Stokes equations could be tested.

The most accurate flux is Roe's original flux because it uses the reconstructed values in the central differencing. The reconstructed values are of higher accuracy so Roe's flux is a second-order accurate CD. It has built-in higher-order reconstruction. If a higher-order CD using reconstructed values could be constructed, then this could be more accurate than Roe's original central difference.

APPENDICES

APPENDIX A

COMPLETE DERIVATION OF CD4 AND CD6

A.1 CD4

The beginning of the derivation of the fourth-order central-difference flux starts with taking Taylor series expansions at the cell centers with respect to $F_{i+\frac{1}{2}}$. The series is carried out until the order of accuracy desired is achieved. In this case, since fourth-order accuracy is required, the series is carried out until the third-order derivative is present.

$$F_{i+2} = F_{i+\frac{1}{2}} + \frac{3}{2}\Delta x\frac{\partial F}{\partial x} + \frac{9}{8}\Delta x^2\frac{\partial^2 F}{\partial x^2} + \frac{27}{48}\Delta x^3\frac{\partial^3 F}{\partial x^3} + O(4^{th}) \tag{A.1}$$

$$F_{i+1} = F_{i+\frac{1}{2}} + \frac{1}{2}\Delta x\frac{\partial F}{\partial x} + \frac{1}{8}\Delta x^2\frac{\partial^2 F}{\partial x^2} + \frac{1}{48}\Delta x^3\frac{\partial^3 F}{\partial x^3} + O(4^{th}) \tag{A.2}$$

$$F_{i} = F_{i+\frac{1}{2}} - \frac{1}{2}\Delta x\frac{\partial F}{\partial x} + \frac{1}{8}\Delta x^2 8\frac{\partial^2 F}{\partial x^2} - \frac{1}{48}\Delta x^3\frac{\partial^3 F}{\partial x^3} + O(4^{th}) \tag{A.3}$$

$$F_{i-1} = F_{i+\frac{1}{2}} - \frac{3}{2}\Delta x\frac{\partial F}{\partial x} + \frac{9}{8}\Delta x^2 8\frac{\partial^2 F}{\partial x^2} - \frac{27}{48}\Delta x^3\frac{\partial^3 F}{\partial x^3} + O(4^{th}) \tag{A.4}$$

The solution form is assumed in the form of Equation A.5.

$$F_{i+\frac{1}{2}} = aF_{i+2} + bF_{i+1} + cF_i + dF_{i-1} \tag{A.5}$$

In order to solve for the coefficients in Equation A.5, they are put into matrix form and solved where only $F_{i+\frac{1}{2}}$ is not zero.

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 3 & 1 & -1 & -3 \\ 9 & 1 & 1 & 9 \\ 27 & 1 & -1 & -27 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{A.6}$$

With the matrix solved, the constant coefficients are known and can be put back in Equation A.5 resulting in the fourth-order central-difference flux at cell face $i + \frac{1}{2}$, shown in Equation A.7.

$$F_{i+\frac{1}{2}} = \frac{1}{16}(-F_{i+2} + 9F_{i+1} + 9F_i - F_{i-1}) \tag{A.7}$$

A.2 CD6

The sixth-order central-difference flux is found in the same manner as the fourth-order central-difference flux. First, a Taylor series expansion is taken of the grid points about cell face, $i + \frac{1}{2}$, to the fifth derivative to get sixth-order accuracy.

$$F_{i+3} = F_{i+\frac{1}{2}} + \frac{5}{2}\Delta x\frac{\partial F}{\partial x} + \frac{25}{8}\Delta x^2\frac{\partial^2 F}{\partial x^2} + \frac{125}{48}\Delta x^3\frac{\partial^3 F}{\partial x^3} + \frac{625}{384}\Delta x^4\frac{\partial^4 F}{\partial x^4} + \frac{3125}{3840}\Delta x^5\frac{\partial^5 F}{\partial x^5} + O(6^{th}) \tag{A.8}$$

$$F_{i+2} = F_{i+\frac{1}{2}} + \frac{3}{2}\Delta x\frac{\partial F}{\partial x} + \frac{9}{8}\Delta x^2\frac{\partial^2 F}{\partial x^2} + \frac{27}{48}\Delta x^3\frac{\partial^3 F}{\partial x^3} + \frac{81}{384}\Delta x^4\frac{\partial^4 F}{\partial x^4} + \frac{243}{3840}\Delta x^5\frac{\partial^5 F}{\partial x^5} + O(6^{th}) \tag{A.9}$$

$$F_{i+1} = F_{i+\frac{1}{2}} + \frac{1}{2}\Delta x\frac{\partial F}{\partial x} + \frac{1}{8}\Delta x^2\frac{\partial^2 F}{\partial x^2} + \frac{1}{48}\Delta x^3\frac{\partial^3 F}{\partial x^3} + \frac{1}{384}\Delta x^4\frac{\partial^4 F}{\partial x^4} + \frac{1}{3840}\Delta x^5\frac{\partial^5 F}{\partial x^5} + O(6^{th}) \tag{A.10}$$

$$F_{i} = F_{i+\frac{1}{2}} - \frac{1}{2}\Delta x\frac{\partial F}{\partial x} + \frac{1}{8}\Delta x^2 8\frac{\partial^2 F}{\partial x^2} - \frac{1}{48}\Delta x^3\frac{\partial^3 F}{\partial x^3} + \frac{1}{384}\Delta x^4\frac{\partial^4 F}{\partial x^4} - \frac{1}{3840}\Delta x^5\frac{\partial^5 F}{\partial x^5} + O(6^{th}) \tag{A.11}$$

$$F_{i-1} = F_{i+\frac{1}{2}} - \frac{3}{2}\Delta x\frac{\partial F}{\partial x} + \frac{9}{8}\Delta x^2 8\frac{\partial^2 F}{\partial x^2} - \frac{27}{48}\Delta x^3\frac{\partial^3 F}{\partial x^3} + \frac{81}{384}\Delta x^4\frac{\partial^4 F}{\partial x^4} - \frac{243}{3840}\Delta x^5\frac{\partial^5 F}{\partial x^5} + O(6^{th}) \tag{A.12}$$

$$F_{i-2} = F_{i+\frac{1}{2}} - \frac{5}{2}\Delta x\frac{\partial F}{\partial x} + \frac{25}{8}\Delta x^2 8\frac{\partial^2 F}{\partial x^2} - \frac{125}{48}\Delta x^3\frac{\partial^3 F}{\partial x^3} + \frac{625}{384}\Delta x^4\frac{\partial^4 F}{\partial x^4} - \frac{3125}{3840}\Delta x^5\frac{\partial^5 F}{\partial x^5} + O(6^{th}) \tag{A.13}$$

The solution is assumed in the form of Equation A.14.

$$F_{i+\frac{1}{2}} = aF_{i+3} + bF_{i+2} + cF_{i+1} + dF_i + eF_{i-1} + fF_{i-2} \tag{A.14}$$

The values from the Taylor series are put into matrix form and solved.

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 5 & 3 & 1 & -1 & -3 & -5 \\ 25 & 9 & 1 & 1 & 9 & 25 \\ 125 & 27 & 1 & -1 & -27 & -125 \\ 625 & 81 & 1 & 1 & 81 & 625 \\ 3125 & 243 & 1 & -1 & -243 & -3125 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{A.15}$$

Finally, Equation A.16 is now known.

$$F_{i+\frac{1}{2}} = \frac{1}{256}(3F_{i+3} - 25F_{i+2} + 150F_{i+1} + 150F_i - 25F_{i-1} + 3F_{i-2}) \tag{A.16}$$

APPENDIX B

MATLAB CODE

```
clear,clc
close all
tic

t = 0.5;    % Time
scheme = 3; % Scheme (1=RK4, 2=TVD-RK3, 3=TVD-RK4)
flux = 2;   % flux (1=CD2, 2=WENO, 3=CD4, 4=CD6)
rec = 2;    % reconstruction (3=3rd order, 5=5th order)
type = 4;   % Test Problem (1=sin wave, 2=square wave, 3=single disconiunity, 4=3)
cfl = 0.01;  % CFL number (<1)

a = 1;

ep = 1e-6; % parameter used in alpha calculation

% setting up initial U
[U,actual,dx,x,xxx,n] = test_case(t,type);

time = 0; % Initial time
k = 0; % Initial k (used for debugging)

% WENO TVD-RK3
if scheme == 3
    while time<t

        [Uim3, Uim2, Uim1, Ui, Uip1, Uip2, Uip3] = U_setup(U);

        if rec == 0
            URip12 = 0;
            ULip12 = 0;
            URim12 = 0;
            ULim12 = 0;
        elseif rec == 1
            URip12 = Uip1;
            ULip12 = Ui;
            URim12 = Ui;
            ULim12 = Uim1;
        elseif rec == 2
            [URip12, ULip12, URim12, ULim12] = MUSCL_fast(U);
        elseif rec == 3
            [URip12, ULip12, URim12, ULim12] = WENO3_fast(U);
        elseif rec == 5
            [URip12, ULip12, URim12, ULim12] = WENO5_fast(U);
        end

        if flux == 1
            Fcdip12 = a*(Ui+Uip1)/2;
```

```
    Fcdim12 = a*(Uim1+Ui)/2;
elseif flux == 2
    Fcdip12 = a*(URip12+ULip12)/2;
    Fcdim12 = a*(URim12+ULim12)/2;
elseif flux == 3
    Fcdip12 = a/16*(-Uip2+9*Uip1+9*Ui-Uim1);
    Fcdim12 = a/16*(-Uip1+9*Ui+9*Uim1-Uim2);
elseif flux == 4
    Fcdip12 = a/256*(3*Uim2-25*Uim1+150*Ui+150*Uip1-25*Uip2+3*Uip3);
    Fcdim12 = a/256*(3*Uim3-25*Uim2+150*Uim1+150*Ui-25*Uip1+3*Uip2);
end

fip12 = Fcdip12-0.5*a*(URip12-ULip12);
fim12 = Fcdim12-0.5*a*(URim12-ULim12);

S = a;
dt = dx/S*cfl;
L0 = 1/dx*(fim12-fip12);
U1 = U+dt*L0;

[Uim3, Uim2, Uim1, Ui, Uip1, Uip2, Uip3] = U_setup(U1);

if rec == 0
    URip12 = 0;
    ULip12 = 0;
    URim12 = 0;
    ULim12 = 0;
elseif rec == 1
    URip12 = Uip1;
    ULip12 = Ui;
    URim12 = Ui;
    ULim12 = Uim1;
elseif rec == 2
    [URip12, ULip12, URim12, ULim12] = MUSCL_fast(U);
elseif rec == 3
    [URip12, ULip12, URim12, ULim12] = WENO3_fast(U);
elseif rec == 5
    [URip12, ULip12, URim12, ULim12] = WENO5_fast(U);
end

if flux == 1
    Fcdip12 = a*(Ui+Uip1)/2;
    Fcdim12 = a*(Uim1+Ui)/2;
elseif flux == 2
    Fcdip12 = a*(URip12+ULip12)/2;
    Fcdim12 = a*(URim12+ULim12)/2;
elseif flux == 3
    Fcdip12 = a/16*(-Uip2+9*Uip1+9*Ui-Uim1);
    Fcdim12 = a/16*(-Uip1+9*Ui+9*Uim1-Uim2);
elseif flux == 4
    Fcdip12 = a/256*(3*Uim2-25*Uim1+150*Ui+150*Uip1-25*Uip2+3*Uip3);
    Fcdim12 = a/256*(3*Uim3-25*Uim2+150*Uim1+150*Ui-25*Uip1+3*Uip2);
end

fip12 = Fcdip12-0.5*a*(URip12-ULip12);
fim12 = Fcdim12-0.5*a*(URim12-ULim12);

L1 = 1/dx*(fim12-fip12);
U2 = 3/4*U+1/4*U1+1/4*dt*L1;

[Uim3, Uim2, Uim1, Ui, Uip1, Uip2, Uip3] = U_setup(U2);

if rec == 0
    URip12 = 0;
```

```
            ULip12 = 0;
            URim12 = 0;
            ULim12 = 0;
        elseif rec == 1
            URip12 = Uip1;
            ULip12 = Ui;
            URim12 = Ui;
            ULim12 = Uim1;
        elseif rec == 2
            [URip12, ULip12, URim12, ULim12] = MUSCL_fast(U);
        elseif rec == 3
            [URip12, ULip12, URim12, ULim12] = WENO3_fast(U);
        elseif rec == 5
            [URip12, ULip12, URim12, ULim12] = WENO5_fast(U);
        end

        if flux == 1
            Fcdip12 = a*(Ui+Uip1)/2;
            Fcdim12 = a*(Uim1+Ui)/2;
        elseif flux == 2
            Fcdip12 = a*(URip12+ULip12)/2;
            Fcdim12 = a*(URim12+ULim12)/2;
        elseif flux == 3
            Fcdip12 = a/16*(-Uip2+9*Uip1+9*Ui-Uim1);
            Fcdim12 = a/16*(-Uip1+9*Ui+9*Uim1-Uim2);
        elseif flux == 4
            Fcdip12 = a/256*(3*Uim2-25*Uim1+150*Ui+150*Uip1-25*Uip2+3*Uip3);
            Fcdim12 = a/256*(3*Uim3-25*Uim2+150*Uim1+150*Ui-25*Uip1+3*Uip2);
        end

        fip12 = Fcdip12-0.5*a*(URip12-ULip12);
        fim12 = Fcdim12-0.5*a*(URim12-ULim12);
        L2 = 1/dx*(fim12-fip12);
        Unp = 1/3*U+2/3*U2+2/3*dt*L2;

        U = Unp;

        time = time+dt;
        k = k+1;
    end
    xx = x-a*time;
end

% Plotting
figure(1)
plot(xx,U,'k--.',xxx,actual,'k')
if type == 1
    axis([-1 1 0 1])
elseif type == 2
    axis([0 1 0 1.2])
elseif type == 3
    axis([-0.5 0.5 0 1.2])
elseif type == 4
    axis([-0.1 0.05 -0.4 1.2])
end
title('Q vs x')
xlabel('x')
ylabel('Q')

toc
```

WENO scheme for the 2D linear advection equation

```
clear,clc
close all
```

```
tic

t = 1.5;    % Time
scheme = 3; % Scheme (1=Euler, 3=TVD-RK3, 4=RK4)
flux = 2;   % flux (1=CD2, 2=WENO, 3=CD4, 4=CD6)
rec = 5;    % Order of WENO Scheme (3=3rd order, 5=5th order)
type = 1;   % Test Problem (1=2D step profile, 2=2D sinusoidal profile, 3=double step profile)
cfl = 0.01;  % CFL number (<1)

a = 1;  % Velocity
n = 46; % Number of cells (time dependent)
m = n;
dx = 1/(n-1);
dy = dx;

xstart = 0;
xend = 1;
x = linspace(xstart,xend,n);

ystart = 0;
yend = 1;
y = linspace(ystart,yend,n);

if rec == 1
    nn = rec+2;
    extra_gc = 2*nn;
elseif rec == 2
    nn = rec+1;
    extra_gc = 2*nn;
elseif rec == 3
    nn = rec+1;
    extra_gc = 2*nn;
elseif rec == 5
    nn = rec;
    extra_gc = 2*nn;
else
    fprintf('error order must be 3 or 5\n')
end

% setting up initial U
U = zeros(n+extra_gc,n+extra_gc);
rowL = max(length(U));

if type == 1
    rowL = max(length(U));
    for i = 1:nn
        U((1+nn):rowL,i) = 1;
    end
elseif type == 2
    rowL = max(find(y<0.3414));
    for i = 1:rowL
        for j = 1:nn
            U(i+nn-1,j) = sin(pi/2*max(1-abs(y(i)-0.1707)/0.1707,0));
        end
    end
elseif type == 3
    rowL = max(find(y<0.3));
    for i = 1:nn
        U((1+nn):rowL+1+nn,i) = 1;
    end
end

U1 = U;
U2 = U;
```

```
U3 = U;

time = 0; % Initial time
k = 0; % Initial k (used for debugging/Iteration number)

% Roe-WENO TVD-RK3
if scheme == 3
    while time<t
        for i = 1:n+extra_gc
            for j = 1:m+extra_gc
                if i <= nn
                    fim12(i,j) = 0;
                    fip12(i,j) = 0;
                    gjm12(i,j) = 0;
                    gjp12(i,j) = 0;
                elseif i > n-nn+extra_gc
                    fim12(i,j) = 0;
                    fip12(i,j) = 0;
                    gjm12(i,j) = 0;
                    gjp12(i,j) = 0;
                elseif j <= nn
                    fim12(i,j) = 0;
                    fip12(i,j) = 0;
                    gjm12(i,j) = 0;
                    gjp12(i,j) = 0;
                elseif j > n-nn+extra_gc
                    fim12(i,j) = 0;
                    fip12(i,j) = 0;
                    gjm12(i,j) = 0;
                    gjp12(i,j) = 0;
                else

                    if rec == 1
                        UR_ip12_j = U(i+1,j);
                        UL_ip12_j = U(i,j);
                        UR_i_jp12 = U(i,j+1);
                        UL_i_jp12 = U(i,j);
                        UR_im12_j = U(i,j);
                        UL_im12_j = U(i-1,j);
                        UR_i_jm12 = U(i,j);
                        UL_i_jm12 = U(i,j-1);
                    elseif rec == 2
                        [UR_ip12_j, UL_ip12_j, UR_i_jp12, UL_i_jp12] = MUSCL_2D(U,i,j);
                        [UR_im12_j, UL_im12_j, na1, na2] = MUSCL_2D(U,i-1,j);
                        [na3, na4, UR_i_jm12, UL_i_jm12] = MUSCL_2D(U,i,j-1);
                    elseif rec == 3
                        [UR_ip12_j, UL_ip12_j, UR_i_jp12, UL_i_jp12] = WENO3_2D(U,i,j);
                        [UR_im12_j, UL_im12_j, na1, na2] = WENO3_2D(U,i-1,j);
                        [na3, na4, UR_i_jm12, UL_i_jm12] = WENO3_2D(U,i,j-1);
                    elseif rec == 5
                        [UR_ip12_j, UL_ip12_j, UR_i_jp12, UL_i_jp12] = WENO5_2D(U,i,j);
                        [UR_im12_j, UL_im12_j, na1, na2] = WENO5_2D(U,i-1,j);
                        [na3, na4, UR_i_jm12, UL_i_jm12] = WENO5_2D(U,i,j-1);
                    end

                    if flux == 1
                        Fcdip12 = a*(U(i,j)+U(i+1,j))/2;
                        Fcdim12 = a*(U(i-1,j)+U(i,j))/2;
                        Gcdjp12 = a*(U(i,j)+U(i,j+1))/2;
                        Gcdjm12 = a*(U(i,j-1)+U(i,j))/2;
                    elseif flux == 2
                        Fcdip12 = a*(UR_ip12_j+UL_ip12_j)/2;
                        Fcdim12 = a*(UR_im12_j+UL_im12_j)/2;
                        Gcdjp12 = a*(UR_i_jp12+UL_i_jp12)/2;
```

```
                Gcdjm12 = a*(UR_i_jm12+UL_i_jm12)/2;
            elseif flux == 3
                Fcdip12 = a/16*(-U(i+2,j)+9*U(i+1,j)+9*U(i,j)-U(i-1,j));
                Fcdim12 = a/16*(-U(i+1,j)+9*U(i,j)+9*U(i-1,j)-U(i-2,j));
                Gcdjp12 = a/16*(-U(i,j+2)+9*U(i,j+1)+9*U(i,j)-U(i,j-1));
                Gcdjm12 = a/16*(-U(i,j+1)+9*U(i,j)+9*U(i,j-1)-U(i,j-2));
            elseif flux == 4
                Fcdip12 = a/256*(3*U(i-2,j)-25*U(i-1,j)+150*U(i,j)+150*U(i+1,j)-25*U(i+2,j)+3*U(i+3,j));
                Fcdim12 = a/256*(3*U(i-3,j)-25*U(i-2,j)+150*U(i-1,j)+150*U(i,j)-25*U(i+1,j)+3*U(i+2,j));
                Gcdjp12 = a/256*(3*U(i,j-2)-25*U(i,j-1)+150*U(i,j)+150*U(i,j+1)-25*U(i,j+2)+3*U(i,j+3));
                Gcdjm12 = a/256*(3*U(i,j-3)-25*U(i,j-2)+150*U(i,j-1)+150*U(i,j)-25*U(i,j+1)+3*U(i,j+2));
            end

            fip12(i,j) = Fcdip12-0.5*a*(UR_ip12_j-UL_ip12_j);
            fim12(i,j) = Fcdim12-0.5*a*(UR_im12_j-UL_im12_j);
            gjp12(i,j) = Gcdjp12-0.5*a*(UR_i_jp12-UL_i_jp12);
            gjm12(i,j) = Gcdjm12-0.5*a*(UR_i_jm12-UL_i_jm12);
        end
    end
end

S = a;
dt = dx/S*cfl;
L0 = 1/dx*(fim12-fip12)+1/dy*(gjm12-gjp12);
U1(1+nn:n+nn,1+nn:n+nn) = U(1+nn:n+nn,1+nn:n+nn)+dt*L0(1+nn:n+nn,1+nn:n+nn);

for i = 1:nn
    U1(n+extra_gc-i+1,:) = U1(n+extra_gc-nn,:);
    U1(:,n+extra_gc-i+1) = U1(:,n+extra_gc-nn);
end

for i = 1:n+extra_gc
    for j = 1:m+extra_gc
        if i <= nn
            fim12(i,j) = 0;
            fip12(i,j) = 0;
            gjm12(i,j) = 0;
            gjp12(i,j) = 0;
        elseif i > n-nn+extra_gc
            fim12(i,j) = 0;
            fip12(i,j) = 0;
            gjm12(i,j) = 0;
            gjp12(i,j) = 0;
        elseif j <= nn
            fim12(i,j) = 0;
            fip12(i,j) = 0;
            gjm12(i,j) = 0;
            gjp12(i,j) = 0;
        elseif j > n-nn+extra_gc
            fim12(i,j) = 0;
            fip12(i,j) = 0;
            gjm12(i,j) = 0;
            gjp12(i,j) = 0;
        else

            if rec == 1
                UR_ip12_j = U(i+1,j);
                UL_ip12_j = U(i,j);
                UR_i_jp12 = U(i,j+1);
                UL_i_jp12 = U(i,j);
                UR_im12_j = U(i,j);
                UL_im12_j = U(i-1,j);
                UR_i_jm12 = U(i,j);
                UL_i_jm12 = U(i,j-1);
```

```
            elseif rec == 2
                [UR_ip12_j, UL_ip12_j, UR_i_jp12, UL_i_jp12] = MUSCL_2D(U,i,j);
                [UR_im12_j, UL_im12_j, na1, na2] = MUSCL_2D(U,i-1,j);
                [na3, na4, UR_i_jm12, UL_i_jm12] = MUSCL_2D(U,i,j-1);
            elseif rec == 3
                [UR_ip12_j, UL_ip12_j, UR_i_jp12, UL_i_jp12] = WENO3_2D(U,i,j);
                [UR_im12_j, UL_im12_j, na1, na2] = WENO3_2D(U,i-1,j);
                [na3, na4, UR_i_jm12, UL_i_jm12] = WENO3_2D(U,i,j-1);
            elseif rec == 5
                [UR_ip12_j, UL_ip12_j, UR_i_jp12, UL_i_jp12] = WENO5_2D(U,i,j);
                [UR_im12_j, UL_im12_j, na1, na2] = WENO5_2D(U,i-1,j);
                [na3, na4, UR_i_jm12, UL_i_jm12] = WENO5_2D(U,i,j-1);
            end

            if flux == 1
                Fcdip12 = a*(U1(i,j)+U1(i+1,j))/2;
                Fcdim12 = a*(U1(i-1,j)+U1(i,j))/2;
                Gcdjp12 = a*(U1(i,j)+U1(i,j+1))/2;
                Gcdjm12 = a*(U1(i,j-1)+U1(i,j))/2;
            elseif flux == 2
                Fcdip12 = a*(UR_ip12_j+UL_ip12_j)/2;
                Fcdim12 = a*(UR_im12_j+UL_im12_j)/2;
                Gcdjp12 = a*(UR_i_jp12+UL_i_jp12)/2;
                Gcdjm12 = a*(UR_i_jm12+UL_i_jm12)/2;
            elseif flux == 3
                Fcdip12 = a/16*(-U1(i+2,j)+9*U1(i+1,j)+9*U1(i,j)-U1(i-1,j));
                Fcdim12 = a/16*(-U1(i+1,j)+9*U1(i,j)+9*U1(i-1,j)-U1(i-2,j));
                Gcdjp12 = a/16*(-U1(i,j+2)+9*U1(i,j+1)+9*U1(i,j)-U1(i,j-1));
                Gcdjm12 = a/16*(-U1(i,j+1)+9*U1(i,j)+9*U1(i,j-1)-U1(i,j-2));
            elseif flux == 4
                Fcdip12 = a/256*(3*U(i-2,j)-25*U(i-1,j)+150*U(i,j)+150*U(i+1,j)-25*U(i+2,j)+3*U(i+3,j));
                Fcdim12 = a/256*(3*U(i-3,j)-25*U(i-2,j)+150*U(i-1,j)+150*U(i,j)-25*U(i+1,j)+3*U(i+2,j));
                Gcdjp12 = a/256*(3*U(i,j-2)-25*U(i,j-1)+150*U(i,j)+150*U(i,j+1)-25*U(i,j+2)+3*U(i,j+3));
                Gcdjm12 = a/256*(3*U(i,j-3)-25*U(i,j-2)+150*U(i,j-1)+150*U(i,j)-25*U(i,j+1)+3*U(i,j+2));
            end

            fip12(i,j) = Fcdip12-0.5*a*(UR_ip12_j-UL_ip12_j);
            fim12(i,j) = Fcdim12-0.5*a*(UR_im12_j-UL_im12_j);
            gjp12(i,j) = Gcdjp12-0.5*a*(UR_i_jp12-UL_i_jp12);
            gjm12(i,j) = Gcdjm12-0.5*a*(UR_i_jm12-UL_i_jm12);
        end
    end
end

S = a;
dt = dx/S*cfl;
L1 = 1/dx*(fim12-fip12)+1/dy*(gjm12-gjp12);
U2(1+nn:n+nn,1+nn:n+nn) = 3/4*U(1+nn:n+nn,1+nn:n+nn)+1/4*U1(1+nn:n+nn,1+nn:n+nn)+1/4*dt*L1(1+nn:n+nn,1+nn:n+nn);

for i = 1:nn
    U2(n+extra_gc-i+1,:) = U2(n+extra_gc-nn,:);
    U2(:,n+extra_gc-i+1) = U2(:,n+extra_gc-nn);
end

for i = 1:n+extra_gc
    for j = 1:m+extra_gc
        if i <= nn
            fim12(i,j) = 0;
            fip12(i,j) = 0;
            gjm12(i,j) = 0;
            gjp12(i,j) = 0;
        elseif i > n-nn+extra_gc
            fim12(i,j) = 0;
            fip12(i,j) = 0;
```

```
    gjm12(i,j) = 0;
    gjp12(i,j) = 0;
elseif j <= nn
    fim12(i,j) = 0;
    fip12(i,j) = 0;
    gjm12(i,j) = 0;
    gjp12(i,j) = 0;
elseif j > n-nn+extra_gc
    fim12(i,j) = 0;
    fip12(i,j) = 0;
    gjm12(i,j) = 0;
    gjp12(i,j) = 0;
else

    if rec == 1
        UR_ip12_j = U(i+1,j);
        UL_ip12_j = U(i,j);
        UR_i_jp12 = U(i,j+1);
        UL_i_jp12 = U(i,j);
        UR_im12_j = U(i,j);
        UL_im12_j = U(i-1,j);
        UR_i_jm12 = U(i,j);
        UL_i_jm12 = U(i,j-1);
    elseif rec == 2
        [UR_ip12_j, UL_ip12_j, UR_i_jp12, UL_i_jp12] = MUSCL_2D(U,i,j);
        [UR_im12_j, UL_im12_j, na1, na2] = MUSCL_2D(U,i-1,j);
        [na3, na4, UR_i_jm12, UL_i_jm12] = MUSCL_2D(U,i,j-1);
    elseif rec == 3
        [UR_ip12_j, UL_ip12_j, UR_i_jp12, UL_i_jp12] = WENO3_2D(U,i,j);
        [UR_im12_j, UL_im12_j, na1, na2] = WENO3_2D(U,i-1,j);
        [na3, na4, UR_i_jm12, UL_i_jm12] = WENO3_2D(U,i,j-1);
    elseif rec == 5
        [UR_ip12_j, UL_ip12_j, UR_i_jp12, UL_i_jp12] = WENO5_2D(U,i,j);
        [UR_im12_j, UL_im12_j, na1, na2] = WENO5_2D(U,i-1,j);
        [na3, na4, UR_i_jm12, UL_i_jm12] = WENO5_2D(U,i,j-1);
    end

    if flux == 1
        Fcdip12 = a*(U2(i,j)+U2(i+1,j))/2;
        Fcdim12 = a*(U2(i-1,j)+U2(i,j))/2;
        Gcdjp12 = a*(U2(i,j)+U2(i,j+1))/2;
        Gcdjm12 = a*(U2(i,j-1)+U2(i,j))/2;
    elseif flux == 2
        Fcdip12 = a*(UR_ip12_j+UL_ip12_j)/2;
        Fcdim12 = a*(UR_im12_j+UL_im12_j)/2;
        Gcdjp12 = a*(UR_i_jp12+UL_i_jp12)/2;
        Gcdjm12 = a*(UR_i_jm12+UL_i_jm12)/2;
    elseif flux == 3
        Fcdip12 = a/16*(-U2(i+2,j)+9*U2(i+1,j)+9*U2(i,j)-U2(i-1,j));
        Fcdim12 = a/16*(-U2(i+1,j)+9*U2(i,j)+9*U2(i-1,j)-U2(i-2,j));
        Gcdjp12 = a/16*(-U2(i,j+2)+9*U2(i,j+1)+9*U2(i,j)-U2(i,j-1));
        Gcdjm12 = a/16*(-U2(i,j+1)+9*U2(i,j)+9*U2(i,j-1)-U2(i,j-2));
    elseif flux == 4
        Fcdip12 = a/256*(3*U(i-2,j)-25*U(i-1,j)+150*U(i,j)+150*U(i+1,j)-25*U(i+2,j)+3*U(i+3,j));
        Fcdim12 = a/256*(3*U(i-3,j)-25*U(i-2,j)+150*U(i-1,j)+150*U(i,j)-25*U(i+1,j)+3*U(i+2,j));
        Gcdjp12 = a/256*(3*U(i,j-2)-25*U(i,j-1)+150*U(i,j)+150*U(i,j+1)-25*U(i,j+2)+3*U(i,j+3));
        Gcdjm12 = a/256*(3*U(i,j-3)-25*U(i,j-2)+150*U(i,j-1)+150*U(i,j)-25*U(i,j+1)+3*U(i,j+2));
    end

    fip12(i,j) = Fcdip12-0.5*a*(UR_ip12_j-UL_ip12_j);
    fim12(i,j) = Fcdim12-0.5*a*(UR_im12_j-UL_im12_j);
    gjp12(i,j) = Gcdjp12-0.5*a*(UR_i_jp12-UL_i_jp12);
    gjm12(i,j) = Gcdjm12-0.5*a*(UR_i_jm12-UL_i_jm12);
end
```

```
            end
        end

        S = a;
        dt = dx/S*cfl;
        L2 = 1/dx*(fim12-fip12)+1/dy*(gjm12-gjp12);
        Unp(1+nn:n+nn,1+nn:n+nn) = 1/3*U(1+nn:n+nn,1+nn:n+nn)+2/3*U2(1+nn:n+nn,1+nn:n+nn)+2/3*dt*L2(1+nn:n+nn,1+nn:n+nn);

        U(1+nn:n+nn,1+nn:n+nn) = Unp(1+nn:n+nn,1+nn:n+nn);

        for i = 1:nn
            U(n+extra_gc-i+1,:) = U(n+extra_gc-nn,:);
            U(:,n+extra_gc-i+1) = U(:,n+extra_gc-nn);
        end

        time = time+dt;
        k = k+1;
    end
end

% Plotting
Unew = U((1+nn):n+nn,(1+nn):n+nn);
[X,Y] = meshgrid(x,y);
y08 = max(find(y<0.8));

for i = 1:n
    y0 = Unew(y08,i);
    y1 = Unew(y08+1,i);
    x0 = x(y08);
    x1 = x(y08+1);
    xx(i,1) = (y0*(x1-0.8)+y1*(0.8-x0))/(x1-x0);
end
figure(1)
contourf(X,Y,Unew)

figure(2)
plot(x,xx)

fprintf('*********************************************************************\n')
fprintf('Iteration = %1.0f\n',k)
fprintf('Run Time = %1.3f [s]\n',toc)
```

Functions

```
function [URip12,ULip12,URim12,ULim12] = MUSCL_fast(U)

phi_type = 1;
kappa = 1;
ep = 1e-10;

n = length(U);
Uim2 = U(2:n-5,1);
Uim1 = U(3:n-4,1);
Ui = U(4:n-3,1);
Uip1 = U(5:n-2,1);
Uip2 = U(6:n-1,1);

Ui = [0; 0; 0; Ui; 0; 0; 0];
Uim1 = [0; 0; 0; Uim1; 0; 0; 0];
Uim2 = [0; 0; 0; Uim2; 0; 0; 0];
Uip1 = [0; 0; 0; Uip1; 0; 0; 0];
Uip2 = [0; 0; 0; Uip2; 0; 0; 0];

rip1 = (Ui-Uim1+ep)./(Uip1-Ui+ep);
rim1 = (Uim1-Uim2+ep)./(Ui-Uim1+ep);
```

```
% limiter
if phi_type == 1 % Minmod
    phiip1 = max(0,min(1,rip1));
    phiim1 = max(0,min(1,rim1));
elseif phi_type == 2 % van Leer
    phiip1 = (rip1+abs(rip1))./(1+abs(rip1));
    phiim1 = (rim1+abs(rim1))./(1+abs(rim1));
elseif phi_type == 3 % Barth-Jesperson
    phiip1 = 0.5*(rip1+1).*min(min(1,4*rip1./(rip1+1)),min(1,4./(rip1+1)));
    phiim1 = 0.5*(rim1+1).*min(min(1,4*rim1./(rim1+1)),min(1,4./(rim1+1)));
elseif phi_type == 4 % Superbee
    phiip1 = max(max(0,min(1,2*rip1)),min(2,rip1));
    phiim1 = max(max(0,min(1,2*rim1)),min(2,rim1));
elseif phi_type == 5 % van albada
    phiip1 = 2*rip1./(rip1.^2+1);
    phiim1 = 2*rim1./(rim1.^2+1);
end

% Reconstruction
URip12 = Uip1-phiip1/4.*((1+kappa)*(Uip1-Ui)+(1-kappa)*(Uip2-Uip1));
ULip12 = Ui+phiip1/4.*((1-kappa)*(Ui-Uim1)+(1+kappa)*(Uip1-Ui));
URim12 = Ui-phiim1/4.*((1+kappa)*(Ui-Uim1)+(1-kappa)*(Uip1-Ui));
ULim12 = Uim1+phiim1/4.*((1-kappa)*(Uim1-Uim2)+(1+kappa)*(Ui-Uim1));
end

function [URip12, ULip12, URjp12, ULjp12] = MUSCL_2D(U,i,j)

phi_type = 1;
kappa = 1;
lam = 1;
ep = 1e-10;

% i's
% r's
rip1 = (U(i,j)-U(i-1,j)+ep)/(U(i+1,j)-U(i,j)+ep);
rim1 = (U(i-1,j)-U(i-2,j)+ep)/(U(i,j)-U(i-1,j)+ep);

% limiter
if phi_type == 1 % Minmod
    phiip1 = max(0,min(1,rip1));
    phiim1 = max(0,min(1,rim1));
elseif phi_type == 2 % van Leer
    phiip1 = (rip1+abs(rip1))/(1+abs(rip1));
    phiim1 = (rim1+abs(rim1))/(1+abs(rim1));
elseif phi_type == 3 % Barth-Jesperson
    phiip1 = 0.5*(rip1+1)*min(min(1,4*rip1/(rip1+1)),min(1,4/(rip1+1)));
    phiim1 = 0.5*(rim1+1)*min(min(1,4*rim1/(rim1+1)),min(1,4/(rim1+1)));
elseif phi_type == 4 % Superbee
    phiip1 = max(max(0,min(1,2*rip1)),min(2,rip1));
    phiim1 = max(max(0,min(1,2*rim1)),min(2,rim1));
elseif phi_type == 5 % van albada
    phiip1 = 2*rip1/(rip1^2+1);
    phiim1 = 2*rim1/(rim1^2+1);
end

% Reconstruction
URip12 = U(i+1,j)-phiip1/4/lam*((1+kappa)*(U(i+1,j)-U(i,j))+(1-kappa)*(U(i+2,j)-U(i+1,j)));
ULip12 = U(i,j)+phiip1/4/lam*((1-kappa)*(U(i,j)-U(i-1,j))+(1+kappa)*(U(i+1,j)-U(i,j)));

% j's
rjp1 = (U(i,j)-U(i,j-1)+ep)/(U(i,j+1)-U(i,j)+ep);
rjm1 = (U(i,j-1)-U(i,j-2)+ep)/(U(i,j)-U(i,j-1)+ep);
```

```
% limiter
if phi_type == 1 % Minmod
    phijp1 = max(0,min(1,rjp1));
    phijm1 = max(0,min(1,rjm1));
elseif phi_type == 2 % van Leer
    phijp1 = (rjp1+abs(rjp1))/(1+abs(rjp1));
    phijm1 = (rjm1+abs(rjm1))/(1+abs(rjm1));
elseif phi_type == 3 % Barth-Jesperson
    phijp1 = 0.5*(rjp1+1)*min(min(1,4*rjp1/(rjp1+1)),min(1,4/(rjp1+1)));
    phijm1 = 0.5*(rjm1+1)*min(min(1,4*rjm1/(rjm1+1)),min(1,4/(rjm1+1)));
elseif phi_type == 4 % Superbee
    phijp1 = max(max(0,min(1,2*rjp1)),min(2,rjp1));
    phijm1 = max(max(0,min(1,2*rjm1)),min(2,rjm1));
elseif phi_type == 5 % van albada
    phijp1 = 2*rjp1/(rjp1^2+1);
    phijm1 = 2*rjm1/(rjm1^2+1);
end

% Reconstruction
URjp12 = U(i,j+1)-phijp1/4/lam*((1+kappa)*(U(i,j+1)-U(i,j))+(1-kappa)*(U(i,j+2)-U(i,j+1)));
ULjp12 = U(i,j)+phijp1/4/lam*((1-kappa)*(U(i,j)-U(i,j-1))+(1+kappa)*(U(i,j+1)-U(i,j)));
end

function [URip12, ULip12, URim12, ULim12] = WENO3_fast(U)
% Hayden Arceneaux
% 10-11-2018
% WENO5 interpolation scheme for linear advection equation du/dt+a*du/dx=0
% Inputs:
%   U - advected quanity
%   i - cell face
% Outputs:
%   URip12 - reconstructed right cell face Ui+1/2
%   ULip12 - reconstructed left cell face Ui+i/2
%   URim12 - reconstructed right cell face Ui-i/2
%   ULim12 - reconstructed left cell face Ui-i/2

n = length(U);

C1 = 1/3;
C2 = 2/3;
ep = 1e-6;
p = 2;

Uim3 = U(1:n-6,1);
Uim2 = U(2:n-5,1);
Uim1 = U(3:n-4,1);
Ui = U(4:n-3,1);
Uip1 = U(5:n-2,1);
Uip2 = U(6:n-1,1);
Uip3 = U(7:n,1);

Ui = [0; 0; 0; Ui; 0; 0; 0];
Uim1 = [0; 0; 0; Uim1; 0; 0; 0];
Uim2 = [0; 0; 0; Uim2; 0; 0; 0];
Uim3 = [0; 0; 0; Uim3; 0; 0; 0];
Uip1 = [0; 0; 0; Uip1; 0; 0; 0];
Uip2 = [0; 0; 0; Uip2; 0; 0; 0];
Uip3 = [0; 0; 0; Uip3; 0; 0; 0];

% URi+1/2
IS1 = (Uip2-Uip1).^2;
IS2 = (Uip1-Ui).^2;

alpha1 = C1./(ep+IS1).^p;
```

```
alpha2 = C2./(ep+IS2).^p;
alphatot = alpha1+alpha2;

omega1 = alpha1./alphatot;
omega2 = alpha2./alphatot;

q1 = 3/2*Uip1-1/2*Uip2;
q2 = 1/2*Ui+1/2*Uip1;

URip12 = omega1.*q1+omega2.*q2;

% ULi+1/2
IS1 = (Ui-Uim1).^2;
IS2 = (Uip1-Ui).^2;

alpha1 = C1./(ep+IS1).^p;
alpha2 = C2./(ep+IS2).^p;
alphatot = alpha1+alpha2;

omega1 = alpha1./alphatot;
omega2 = alpha2./alphatot;

q1 = 3/2*Ui-1/2*Uim1;
q2 = 1/2*Ui+1/2*Uip1;

ULip12 = omega1.*q1+omega2.*q2;

% URi-1/2
IS1 = (Uip1-Ui).^2;
IS2 = (Ui-Uim1).^2;

alpha1 = C1./(ep+IS1).^p;
alpha2 = C2./(ep+IS2).^p;
alphatot = alpha1+alpha2;

omega1 = alpha1./alphatot;
omega2 = alpha2./alphatot;

q1 = 3/2*Ui-1/2*Uip1;
q2 = 1/2*Uim1+1/2*Ui;

URim12 = omega1.*q1+omega2.*q2;

% ULi-1/2
IS1 = (Uim1-Uim2).^2;
IS2 = (Ui-Uim1).^2;

alpha1 = C1./(ep+IS1).^p;
alpha2 = C2./(ep+IS2).^p;
alphatot = alpha1+alpha2;

omega1 = alpha1./alphatot;
omega2 = alpha2./alphatot;

q1 = 3/2*Uim1-1/2*Uim2;
q2 = 1/2*Uim1+1/2*Ui;

ULim12 = omega1.*q1+omega2.*q2;
end

function [URip12, ULip12, URim12, ULim12] = WENO5_fast(U)
% Hayden Arceneaux
% 10-10-2018
% WENO5 interpolation scheme for linear advection equation du/dt+a*du/dx=0
```

```
% Inputs:
%   U - advected quanity
% Outputs:
%   URip12 - reconstructed right cell face Ui+1/2
%   ULip12 - reconstructed left cell face Ui+i/2
%   URim12 - reconstructed right cell face Ui-i/2
%   ULim12 - reconstructed left cell face Ui-i/2

n = length(U);

C0 = 0.1;
C1 = 0.6;
C2 = 0.3;
ep = 1e-6;
p = 2;

Uim3 = U(1:n-6,1);
Uim2 = U(2:n-5,1);
Uim1 = U(3:n-4,1);
Ui = U(4:n-3,1);
Uip1 = U(5:n-2,1);
Uip2 = U(6:n-1,1);
Uip3 = U(7:n,1);

Ui = [0; 0; 0; Ui; 0; 0; 0];
Uim1 = [0; 0; 0; Uim1; 0; 0; 0];
Uim2 = [0; 0; 0; Uim2; 0; 0; 0];
Uim3 = [0; 0; 0; Uim3; 0; 0; 0];
Uip1 = [0; 0; 0; Uip1; 0; 0; 0];
Uip2 = [0; 0; 0; Uip2; 0; 0; 0];
Uip3 = [0; 0; 0; Uip3; 0; 0; 0];

% URi+1/2
IS0 = 13/12.*(Uip1-2*Uip2+Uip3).^2+1/4.*(3*Uip1-4*Uip2+Uip3).^2;
IS1 = 13/12.*(Ui-2*Uip1+Uip2).^2+1/4.*(Ui-Uip2).^2;
IS2 = 13/12.*(Uim1-2*Ui+Uip1).^2+1/4.*(Uim1-4*Ui+3*Uip1).^2;

alpha0 = C0./(ep+IS0).^p;
alpha1 = C1./(ep+IS1).^p;
alpha2 = C2./(ep+IS2).^p;
alphatot = alpha0+alpha1+alpha2;

omega0 = alpha0./alphatot;
omega1 = alpha1./alphatot;
omega2 = alpha2./alphatot;

q0 = (2*Uip3-7*Uip2+11*Uip1)/6;
q1 = (-Uip2+5*Uip1+2*Ui)/6;
q2 = (2*Uip1+5*Ui-Uim1)/6;

URip12 = omega0.*q0+omega1.*q1+omega2.*q2;

% ULi+1/2
IS0 = 13/12.*(Uim2-2*Uim1+Ui).^2+1/4.*(Uim2-4*Uim1+3*Ui).^2;
IS1 = 13/12.*(Uim1-2*Ui+Uip1).^2+1/4.*(Uim1-Uip1).^2;
IS2 = 13/12.*(Ui-2*Uip1+Uip2).^2+1/4.*(3*Ui-4*Uip1+Uip2).^2;

alpha0 = C0./(ep+IS0).^p;
alpha1 = C1./(ep+IS1).^p;
alpha2 = C2./(ep+IS2).^p;
alphatot = alpha0+alpha1+alpha2;

omega0 = alpha0./alphatot;
omega1 = alpha1./alphatot;
```

```
omega2 = alpha2./alphatot;

q0 = (2*Uim2-7*Uim1+11*Ui)/6;
q1 = (-Uim1+5*Ui+2*Uip1)/6;
q2 = (2*Ui+5*Uip1-Uip2)/6;

ULip12 = omega0.*q0+omega1.*q1+omega2.*q2;

% URi-1/2
IS0 = 13/12.*(Ui-2*Uip1+Uip2).^2+1/4.*(3*Ui-4*Uip1+Uip2).^2;
IS1 = 13/12.*(Uim1-2*Ui+Uip1).^2+1/4.*(Uim1-Uip1).^2;
IS2 = 13/12.*(Uim2-2*Uim1+Ui).^2+1/4.*(Uim2-4*Uim1+3*Ui).^2;

alpha0 = C0./(ep+IS0).^p;
alpha1 = C1./(ep+IS1).^p;
alpha2 = C2./(ep+IS2).^p;
alphatot = alpha0+alpha1+alpha2;

omega0 = alpha0./alphatot;
omega1 = alpha1./alphatot;
omega2 = alpha2./alphatot;

q0 = (2*Uip2-7*Uip1+11*Ui)/6;
q1 = (-Uip1+5*Ui+2*Uim1)/6;
q2 = (2*Ui+5*Uim1-Uim2)/6;

URim12 = omega0.*q0+omega1.*q1+omega2.*q2;

% ULi-1/2
IS0 = 13/12.*(Uim3-2*Uim2+Uim1).^2+1/4.*(Uim3-4*Uim2+3*Uim1).^2;
IS1 = 13/12.*(Uim2-2*Uim1+Ui).^2+1/4.*(Uim2-Ui).^2;
IS2 = 13/12.*(Uim1-2*Ui+Uip1).^2+1/4.*(3*Uim1-4*Ui+Uip1).^2;

alpha0 = C0./(ep+IS0).^p;
alpha1 = C1./(ep+IS1).^p;
alpha2 = C2./(ep+IS2).^p;
alphatot = alpha0+alpha1+alpha2;

omega0 = alpha0./alphatot;
omega1 = alpha1./alphatot;
omega2 = alpha2./alphatot;

q0 = (2*Uim3-7*Uim2+11*Uim1)/6;
q1 = (-Uim2+5*Uim1+2*Ui)/6;
q2 = (2*Uim1+5*Ui-Uip1)/6;

ULim12 = omega0.*q0+omega1.*q1+omega2.*q2;
end

function [URip12, ULip12, URjp12, ULjp12] = WENO3_2D(U,i,j)
% Hayden Arceneaux
% 1-7-2019
% WENO5 interpolation scheme for linear advection equation du/dt+a*du/dx=0
% Inputs:
%   U - advected quanity
%   i - cell face
% Outputs:
%   URip12 - reconstructed right cell face
%   ULip12 - reconstructed left cell face

C1 = 1/3;
C2 = 2/3;
ep = 1e-6;
p = 2;
```

```
% URi+1/2,j
IS1 = (U(i+2,j)-U(i+1,j))^2;
IS2 = (U(i+1,j)-U(i,j))^2;

alpha1 = C1/(ep+IS1)^p;
alpha2 = C2/(ep+IS2)^p;
alphatot = alpha1+alpha2;

omega1 = alpha1/alphatot;
omega2 = alpha2/alphatot;

q1 = 3/2*U(i+1,j)-1/2*U(i+2,j);
q2 = 1/2*U(i,j)+1/2*U(i+1,j);

URip12 = omega1*q1+omega2*q2;

% ULi+1/2,j
IS1 = (U(i,j)-U(i-1,j))^2;
IS2 = (U(i+1,j)-U(i,j))^2;

alpha1 = C1/(ep+IS1)^p;
alpha2 = C2/(ep+IS2)^p;
alphatot = alpha1+alpha2;

omega1 = alpha1/alphatot;
omega2 = alpha2/alphatot;

q1 = 3/2*U(i,j)-1/2*U(i-1,j);
q2 = 1/2*U(i,j)+1/2*U(i+1,j);

ULip12 = omega1*q1+omega2*q2;

% URi,j+1/2
IS1 = (U(i,j+2)-U(i,j+1))^2;
IS2 = (U(i,j+1)-U(i,j))^2;

alpha1 = C1/(ep+IS1)^p;
alpha2 = C2/(ep+IS2)^p;
alphatot = alpha1+alpha2;

omega1 = alpha1/alphatot;
omega2 = alpha2/alphatot;

q1 = 3/2*U(i,j+1)-1/2*U(i,j+2);
q2 = 1/2*U(i,j)+1/2*U(i,j+1);

URjp12 = omega1*q1+omega2*q2;

% ULi,j+1/2
IS1 = (U(i,j)-U(i,j-1))^2;
IS2 = (U(i,j+1)-U(i,j))^2;

alpha1 = C1/(ep+IS1)^p;
alpha2 = C2/(ep+IS2)^p;
alphatot = alpha1+alpha2;

omega1 = alpha1/alphatot;
omega2 = alpha2/alphatot;

q1 = 3/2*U(i,j)-1/2*U(i,j-1);
q2 = 1/2*U(i,j)+1/2*U(i,j+1);
```

```
ULjp12 = omega1*q1+omega2*q2;
end

function [URip12, ULip12, URjp12, ULjp12] = WENO5_2D(U,i,j)
% Hayden Arceneaux
% 1-24-2019
% WENO5 interpolation scheme for linear advection equation du/dt+a*du/dx=0
% Inputs:
%   U - advected quanity
%   i - cell face
% Outputs:
%   URip12 - reconstructed right cell face
%   ULip12 - reconstructed left cell face

C0 = 0.1;
C1 = 0.6;
C2 = 0.3;
ep = 1e-6;
p = 2;

% URi+1/2
IS0 = 13/12*(U(i+1,j)-2*U(i+2,j)+U(i+3,j))^2+1/4*(3*U(i+1,j)-4*U(i+2,j)+U(i+3,j))^2;
IS1 = 13/12*(U(i,j)-2*U(i+1,j)+U(i+2,j))^2+1/4*(U(i,j)-U(i+2,j))^2;
IS2 = 13/12*(U(i-1,j)-2*U(i,j)+U(i+1,j))^2+1/4*(U(i-1,j)-4*U(i,j)+3*U(i+1,j))^2;

alpha0 = C0/(ep+IS0)^p;
alpha1 = C1/(ep+IS1)^p;
alpha2 = C2/(ep+IS2)^p;
alphatot = alpha0+alpha1+alpha2;

omega0 = alpha0/alphatot;
omega1 = alpha1/alphatot;
omega2 = alpha2/alphatot;

q0 = (2*U(i+3,j)-7*U(i+2,j)+11*U(i+1,j))/6;
q1 = (-U(i+2,j)+5*U(i+1,j)+2*U(i,j))/6;
q2 = (2*U(i+1,j)+5*U(i,j)-U(i-1,1))/6;

URip12 = omega0*q0+omega1*q1+omega2*q2;

% ULi+1/2
IS0 = 13/12*(U(i-2,j)-2*U(i-1,j)+U(i,j))^2+1/4*(U(i-2,j)-4*U(i-1,j)+3*U(i,j))^2;
IS1 = 13/12*(U(i-1,j)-2*U(i,j)+U(i+1,j))^2+1/4*(U(i-1,j)-U(i+1,j))^2;
IS2 = 13/12*(U(i,j)-2*U(i+1,j)+U(i+2,j))^2+1/4*(3*U(i,j)-4*U(i+1,j)+U(i+2,j))^2;

alpha0 = C0/(ep+IS0)^p;
alpha1 = C1/(ep+IS1)^p;
alpha2 = C2/(ep+IS2)^p;
alphatot = alpha0+alpha1+alpha2;

omega0 = alpha0/alphatot;
omega1 = alpha1/alphatot;
omega2 = alpha2/alphatot;

q0 = (2*U(i-2,j)-7*U(i-1,j)+11*U(i,j))/6;
q1 = (-U(i-1,j)+5*U(i,j)+2*U(i+1,j))/6;
q2 = (2*U(i,j)+5*U(i+1,j)-U(i+2,j))/6;

ULip12 = omega0*q0+omega1*q1+omega2*q2;

% URj+1/2
IS0 = 13/12*(U(i,j+1)-2*U(i,j+2)+U(i,j+3))^2+1/4*(3*U(i,j+1)-4*U(i,j+2)+U(i,j+3))^2;
IS1 = 13/12*(U(i,j)-2*U(i,j+1)+U(i,j+2))^2+1/4*(U(i,j)-U(i,j+2))^2;
IS2 = 13/12*(U(i,j-1)-2*U(i,j)+U(i,j+1))^2+1/4*(U(i,j-1)-4*U(i,j)+3*U(i,j+1))^2;
```

```
alpha0 = C0/(ep+IS0)^p;
alpha1 = C1/(ep+IS1)^p;
alpha2 = C2/(ep+IS2)^p;
alphatot = alpha0+alpha1+alpha2;

omega0 = alpha0/alphatot;
omega1 = alpha1/alphatot;
omega2 = alpha2/alphatot;

q0 = (2*U(i,j+3)-7*U(i,j+2)+11*U(i,j+1))/6;
q1 = (-U(i,j+2)+5*U(i,j+1)+2*U(i,j))/6;
q2 = (2*U(i,j+1)+5*U(i,j)-U(i,j-1))/6;

URjp12 = omega0*q0+omega1*q1+omega2*q2;

% ULj+1/2
IS0 = 13/12*(U(i,j-2)-2*U(i,j-1)+U(i,j))^2+1/4*(U(i,j-2)-4*U(i,j-1)+3*U(i,j))^2;
IS1 = 13/12*(U(i,j-1)-2*U(i,j)+U(i,j+1))^2+1/4*(U(i,j-1)-U(i,j+1))^2;
IS2 = 13/12*(U(i,j)-2*U(i,j+1)+U(i,j+2))^2+1/4*(3*U(i,j)-4*U(i,j+1)+U(i,j+2))^2;

alpha0 = C0/(ep+IS0)^p;
alpha1 = C1/(ep+IS1)^p;
alpha2 = C2/(ep+IS2)^p;
alphatot = alpha0+alpha1+alpha2;

omega0 = alpha0/alphatot;
omega1 = alpha1/alphatot;
omega2 = alpha2/alphatot;

q0 = (2*U(i,j-2)-7*U(i,j-1)+11*U(i,j))/6;
q1 = (-U(i,j-1)+5*U(i,j)+2*U(i,j+1))/6;
q2 = (2*U(i,j)+5*U(i,j+1)-U(i,j+2))/6;

ULjp12 = omega0*q0+omega1*q1+omega2*q2;
end

function [U,actual,dx,x,xxx,n] = test_case(t,type)
% Hayden Arceneaux
% 9-7-2018
% Test Cases for linear advection du/dt+a*du/dx=0
% Inputs:
%   t - end time
%   type - test case
%       1 = sin wave
%       2 = square wave
%       3 = single discontinuity
%       4 = single discontinuity length of 1
% Outputs:
%   U - Initial Advected Quanity
%   actual - Analytical Solution
%   dx - distance between cell centers
%   x - grid
%   xxx - analytical solution grid
%   n - number of cells

% Values for smooth data
alpha = 1;
beta = 8;

% Setting up Smooth Data
if type == 1
%     n = t*100; % Number of cells (time dependent)
    dx = 0.02;
```

```
    xstart = -1;
    xend = 1;
    xxx = linspace(xstart,xend,10000);

    xstart = -1;
    xend = t+2;
%     n = (xstart-xend)/dx;
    x = xstart:dx:xend;
    n = length(x);
%     x = linspace(xstart,xend,n);
%     dx = (xend-xstart)/(n-1);
    for i = 1:n
        U(i,1) = alpha*exp(-beta*x(i)^2);
    end
    for i = 1:length(xxx)
        actual(i,1) = alpha*exp(-beta*xxx(i)^2);
    end
end

% Setting up Square Wave
if type == 2
%     n = t*100; % Number of cells (time dependent)
    dx = 0.02;
    xstart = 0;
    xend = 1;
    xxx = linspace(xstart,xend,10000);

    xstart = 0;
    xend = t+2;
    x = xstart:dx:xend;
    n = length(x);

    for i = 1:n
        if x(i) <= 0.3
            U(i,1) = 0;
        elseif x(i) > 0.3 && x(i) <= 0.7
            U(i,1) = 1;
        elseif x(i) > 0.7
            U(i,1) = 0;
        end
    end

    for i = 1:length(xxx)
        if xxx(i) <= 0.3
            actual(i,1) = 0;
        elseif xxx(i) > 0.3 && xxx(i) <= 0.7
            actual(i,1) = 1;
        elseif xxx(i) > 0.7
            actual(i,1) = 0;
        end
    end
end

% Setting up Single Discontinuity
if type == 3
%     n = t*1000; % Number of cells (time dependent)
    dx = 0.02;
    xstart = -1;
    xend = 1;
    xxx = linspace(xstart,xend,10000);

    xstart = -1;
    xend = t+2;
    x = xstart:dx:xend;
```

```
    n = length(x);

    for i = 1:n
        if x(i) < 0
            U(i,1) = 0;
        else
            U(i,1) = 1;
        end
    end

    for i = 1:length(xxx)
        if xxx(i) < 0
            actual(i,1) = 0;
        else
            actual(i,1) = 1;
        end
    end
end

% Setting up Single Discontinuity 2
if type == 4
    n = 512;
%     n = 2048;
    xstart = -1;
    xend = 1;
    xxx = linspace(xstart,xend,10000);

    xstart = -1;
    xend = 1;
    x = linspace(xstart,xend,n);
    dx = (xend-xstart)/(n-1);

    for i = 1:n
        if x(i) < 0
            U(i,1) = 0;
        else
            U(i,1) = 1;
        end
    end

    for i = 1:length(xxx)
        if xxx(i) < 0
            actual(i,1) = 0;
        else
            actual(i,1) = 1;
        end
    end
end
end

function [Uim3, Uim2, Uim1, Ui, Uip1, Uip2, Uip3] = U_setup(U)
% Hayden Arceneaux
% 10-11-2018
% setting up U for fast WENO
% Inputs:
%   U - advected quanity
% Outputs:
%   Uim3 - Ui-3
%   Uim2 - Ui-2
%   Uim1 - Ui-1
%   Ui   - Ui
%   Uip1 - Ui+1
%   Uip2 - Ui+2
%   Uip3 - Ui+3
```

```
n = length(U);

Uim3 = U(1:n-6,1);
Uim2 = U(2:n-5,1);
Uim1 = U(3:n-4,1);
Ui = U(4:n-3,1);
Uip1 = U(5:n-2,1);
Uip2 = U(6:n-1,1);
Uip3 = U(7:n,1);

Ui = [0; 0; 0; Ui; 0; 0; 0];
Uim1 = [0; 0; 0; Uim1; 0; 0; 0];
Uim2 = [0; 0; 0; Uim2; 0; 0; 0];
Uim3 = [0; 0; 0; Uim3; 0; 0; 0];
Uip1 = [0; 0; 0; Uip1; 0; 0; 0];
Uip2 = [0; 0; 0; Uip2; 0; 0; 0];
Uip3 = [0; 0; 0; Uip3; 0; 0; 0];

end
```

REFERENCES

[1] Peter L McCloud. Best practices for unstructured grid shock fitting. In *55th AIAA Aerospace Sciences Meeting*, page 1149, 2017.

[2] Theodore Kai Lee and Xiaolin Zhong. Spurious numerical oscillations in simulation of supersonic flows using shock-capturing schemes. *AIAA journal*, 37(3):313–319, 1999.

[3] Peter Gnoffo and Jeffery White. Computational aerothermodynamic simulation issues on unstructured grids. In *37th AIAA Thermophysics Conference*, page 2371, 2004.

[4] SK Godunov. A finite difference method for the computation of discontinuous solutions of the equations of fluid dynamics. *Sbornik: Mathematics*, 47(8-9):357–393, 1959.

[5] Shreyas Bidadi and Sarma L Rani. Quantification of numerical diffusivity due to tvd schemes in the advection equation. *Journal of Computational Physics*, 261:65–82, 2014.

[6] Eleuterio F Toro. *Riemann solvers and numerical methods for fluid dynamics: a practical introduction*. Springer Science & Business Media, 2013.

[7] Joseph L Steger and RF Warming. Flux vector splitting of the inviscid gasdynamic equations with application to finite-difference methods. *Journal of computational physics*, 40(2):263–293, 1981.

[8] Philip L Roe. Approximate riemann solvers, parameter vectors, and difference schemes. *Journal of computational physics*, 43(2):357–372, 1981.

[9] Ami Harten. High resolution schemes for hyperbolic conservation laws. *Journal of computational physics*, 49(3):357–393, 1983.

[10] Bram Van Leer. Towards the ultimate conservative difference scheme. v. a second-order sequel to godunov's method. *Journal of computational Physics*, 32(1):101–136, 1979.

[11] W Kyle Anderson and Daryl L Bonhaus. An implicit upwind algorithm for computing turbulent flows on unstructured grids. *Computers & Fluids*, 23(1):1–21, 1994.

[12] HQ Yang and Robert E Harris. High-order vertex-centered u-muscl schemes for turbulent flows. *COMMUNICATIONS IN COMPUTATIONAL PHYSICS*, 24(2):356–382, 2018.

[13] Ami Harten, Bjorn Engquist, Stanley Osher, and Sukumar R Chakravarthy. Uniformly high order accurate essentially non-oscillatory schemes, iii. In *Upwind and high-resolution schemes*, pages 218–290. Springer, 1987.

[14] Xu-Dong Liu, Stanley Osher, and Tony Chan. Weighted essentially non-oscillatory schemes. *Journal of computational physics*, 115(1):200–212, 1994.

[15] Rafael Borges, Monique Carmona, Bruno Costa, and Wai Sun Don. An improved weighted essentially non-oscillatory scheme for hyperbolic conservation laws. *Journal of Computational Physics*, 227(6):3191–3211, 2008.

[16] Xuliang Liu, Shuhai Zhang, Hanxin Zhang, and Chi-Wang Shu. A new class of central compact schemes with spectral-like resolution ii: Hybrid weighted nonlinear schemes. *Journal of Computational Physics*, 284:133–154, 2015.

[17] Oliver Friedrich. Weighted essentially non-oscillatory schemes for the interpolation of mean values on unstructured grids. *Journal of computational physics*, 144(1):194–212, 1998.

[18] Yuan Liu and Yong-Tao Zhang. A robust reconstruction for unstructured weno schemes. *Journal of Scientific Computing*, 54(2-3):603–621, 2013.

[19] Richard M Beam and Robert F Warming. An implicit finite-difference algorithm for hyperbolic systems in conservation-law form. *Journal of computational physics*, 22(1):87–110, 1976.

[20] Guang-Shan Jiang and Chi-Wang Shu. Efficient implementation of weighted eno schemes. *Journal of computational physics*, 126(1):202–228, 1996.

[21] Charles Hirsch. Numerical computation of internal and external flows. vol. 2-computational methods for inviscid and viscous flows(book). *Chichester, England and New York, John Wiley & Sons, 1990, 708*, 1990.

[22] Bram Van Leer. Towards the ultimate conservative difference scheme. *Journal of Computational Physics*, 135(2):229–248, 1997.

[23] GD Van Albada, Bram Van Leer, and WWjun Roberts. A comparative study of computational methods in cosmic gas dynamics. In *Upwind and High-Resolution Schemes*, pages 95–103. Springer, 1997.

[24] P.L. Roe. Some contributions to the modelling of discontinuous flows. In *Large-scale computations in fluid mechanics; Proceedings of the Fifteenth Summer Seminar on Applied Mathematics*, La Jolla, CA, 1983. American Mathematical Society.

[25] Yiqing Shen, Baoyuan Wang, and Gecheng Zha. Implicit weno scheme and high order viscous formulas for compressible flows. In *25th AIAA Applied Aerodynamics Conference*, page 4431, 2007.

[26] MS Darwish and F Moukalled. Tvd schemes for unstructured grids. *International Journal of heat and mass transfer*, 46(4):599–611, 2003.