2011

An Implicit High-Order Spectral Difference Method for the Compressible Navier-Stokes Equations Using Adaptive Polynomial Refinement

Caleb J. Barnes
Wright State University

Follow this and additional works at: http://corescholar.libraries.wright.edu/etd_all
Part of the Mechanical Engineering Commons

Repository Citation

This Thesis is brought to you for free and open access by the Theses and Dissertations at CORE Scholar. It has been accepted for inclusion in Browse all Theses and Dissertations by an authorized administrator of CORE Scholar. For more information, please contact corescholar@www.libraries.wright.edu.
An Implicit High-Order Spectral Difference Method for the Compressible Navier-Stokes Equations Using Adaptive Polynomial Refinement

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Engineering

By

CALEB JAMES BARNES
B.S., Wright State University, 2009

2011
Wright State University
August 29, 2011


______________________________
George Huang, Ph.D.
Thesis Director

______________________________
George Huang, Ph.D., Chair
Department of Mechanical and Materials Engineering

Committee on Final Examination

______________________________
George Huang, Ph.D.

______________________________
Joseph Shang, Ph.D.

______________________________
Miguel Visbal, Ph.D.

______________________________
Andrew T. Hsu, Ph.D.
Dean, Graduate School
Abstract


A high/variable-order numerical simulation procedure for gas dynamics problems was developed to model steep grading physical phenomena. Higher order resolution was achieved using an orthogonal polynomial Gauss-Lobatto grid, adaptive polynomial refinement and artificial diffusion activated by a pressure switch. The method is designed to be computationally stable, accurate, and capable of resolving discontinuities and steep gradients without the use of one-sided reconstructions or reducing to low-order. Solutions to several benchmark gas-dynamics problems were produced including a shock-tube and a shock-entropy wave interaction. The scheme’s 1st-order solution was validated in comparison to a 1st-order Roe scheme solution. Higher-order solutions were shown to approach reference values for each problem. Uniform polynomial refinement was shown to be capable of producing increasingly accurate solutions on a very coarse mesh. Adaptive polynomial refinement was employed to selectively refine the solution near steep gradient structures and results were nearly identical to those produced by uniform polynomial refinement. Future work will focus on improvements to the diffusion term, complete extensions to the full compressible Navier-Stokes equations, and multi-dimension formulations.
# Contents

1 Introduction ........................................................................... 1

2 Methodology .......................................................................... 8

2.1 Linearized Euler Equations .................................................. 9

2.2 Delta Formulation .............................................................. 13

2.3 Polynomial Approximation ................................................ 16

2.4 Lobatto Formulation .......................................................... 19

2.5 Global Domain Decomposition ........................................... 20

2.6 Boundary Conditions ........................................................ 20

2.7 Interface Flux Formulation ................................................ 22

2.7.1 Roe Scheme ............................................................... 23

2.7.2 Upwind Characteristics ................................................ 25

2.7.3 Rusanov Solver ........................................................... 27

2.8 Adaptive Polynomial Refinement ........................................ 28

2.8.1 Q-Refinement Switch ................................................... 28

2.8.2 Pressure Refinement Switch .......................................... 30

2.8.3 The Refinement Process ................................................. 31

2.9 Discontinuity Capturing ..................................................... 32

2.10 Artificial Diffusion ........................................................... 32

2.10.1 Applying Artificial Diffusion ........................................ 36
# List of Figures

1.1 Outward polynomial search .................................................. 2
1.2 Inward polynomial search .................................................... 2

2.1 Moving from single-domain to multi-domain ............................... 20
2.2 Extended cell stencil .......................................................... 21
2.3 Spectral difference stencils .................................................. 23
2.4 Node-centered cell stencil .................................................... 26
2.5 Adaptive polynomial refinement block diagram ........................... 31
2.6 Internal cell ................................................................. 34
2.7 Artificial diffusion behavior .................................................. 37
2.8 Low-pass filter applied to 4th degree stencil ............................... 43
2.9 Staggered-grid spectral difference stencil ................................. 44
2.10 Staggered-grid spectral difference stencil - derivative approximation . 44

3.1 Shock-tube configuration ..................................................... 49
3.2 Analytical shock-tube solution .............................................. 49
3.3 Reference Shu-Osher solution ............................................... 50

4.1 Initial high-order solutions .................................................. 53
4.2 Roe scheme vs. upwind characteristics .................................... 54
4.3 1st order vs. 1st degree polynomials ..................................... 56
4.4 Diffusion definition comparison ............................................ 58
4.5 Reduction of oscillations with increasing $K$ ........................................... 60
4.6 Effect of scaling the diffusion term for the shock-tube problem ......... 62
4.7 Comparison of 19th and 99th degree shock-tube solutions ............... 62
4.8 Effect of scaling diffusion on the 99th degree shock-tube solution .... 63
4.9 Effect of scaling diffusion for the Shu-Osher problem ......................... 64
4.10 Effect of scaling diffusion for Shu-Osher problem (magnified) ........... 65
4.11 Diffusion switch for 19th degree shock-tube solution ......................... 67
4.12 Diffusion switch for 49th degree shock-tube solution ......................... 68
4.13 Diffusion switch for 29th degree Shu-Osher solution ......................... 70
4.14 Diffusion switch for 49th degree Shu-Osher solution ......................... 71
4.15 Full vs. selective diffusion for the shock-tube problem .................... 72
4.16 Full vs. selective diffusion for the Shu-Osher problem ..................... 73
4.17 Reduction of oscillations with polynomial refinement ...................... 75
4.18 Uniform polynomial refinement - shock-tube problem ...................... 76
4.19 Uniform polynomial refinement - Shu-Osher problem ...................... 76
4.20 Shock-tube adaptive polynomial refinement ($n = 20, m_{\text{max}} = 30$) . 78
4.21 Shock-tube adaptive polynomial refinement ($n = 20, m_{\text{max}} = 50$) . 79
4.22 Uniform vs. adaptive refinement - shock-tube problem ($m_{\text{max}} = 30$) . 80
4.23 Uniform vs. adaptive refinement - shock-tube problem ($m_{\text{max}} = 50$) . 80
4.24 Shu-Osher adaptive polynomial refinement ($n = 100, m_{\text{max}} = 30$) . 81
4.25 Shu-Osher adaptive polynomial refinement ($n = 20, m_{\text{max}} = 30$) . 83
4.26 Shu-Osher adaptive polynomial refinement ($n = 20, m_{\text{max}} = 50$) . 84
4.27 Uniform vs. adaptive refinement - Shu-Osher problem ($m_{\text{max}} = 30$) . 85
4.28 Uniform vs. adaptive refinement - Shu-Osher problem ($m_{\text{max}} = 50$) . 86
4.29 Shu-Osher adaptive polynomial refinement ($n = 100, m_{\text{max}} = 30$) . 87
List of Tables

2.1  Filter coefficients for centered higher-order filters . . . . . . . . . . . . . 41
2.2  Filter coefficients for one-sided filter at point 2 . . . . . . . . . . . . . . . 42
2.3  Filter coefficients for one-sided filter at point 3 . . . . . . . . . . . . . . . 42
4.1  Guideline p-values . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 65
Acknowledgments

The author greatly appreciates the many discussions with Dr. George Huang and Dr. Joseph Shang at Wright State University. Their knowledge and experience has been a tremendous help in both my knowledge of computational fluid dynamics and professional growth. The sponsorship of Dr. F. Fahroo of AFOSR is sincerely appreciated. The author would also like to thank Dr. Z. J. Wang for providing numerical results for the shock-entropy wave problem.
Dedication

This work is dedicated to my wife, Margaret Barnes. Your encouragement and support has given me the perseverance to continue working hard and reach my goals. Thank you for your patience and uplifting spirit.
## List of Symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\vec{Q}$</td>
<td>vector of conservative variables</td>
</tr>
<tr>
<td>$\Delta \vec{Q}$</td>
<td>vector of conservative variables in delta form</td>
</tr>
<tr>
<td>$\vec{W}$</td>
<td>vector of characteristic variables</td>
</tr>
<tr>
<td>$\vec{F}$</td>
<td>vector of flux values</td>
</tr>
<tr>
<td>$\rho$</td>
<td>density</td>
</tr>
<tr>
<td>$u$</td>
<td>velocity</td>
</tr>
<tr>
<td>$e$</td>
<td>energy per unit mass</td>
</tr>
<tr>
<td>$E$</td>
<td>total energy</td>
</tr>
<tr>
<td>$M$</td>
<td>momentum</td>
</tr>
<tr>
<td>$P$</td>
<td>pressure</td>
</tr>
<tr>
<td>$R$</td>
<td>gas constant</td>
</tr>
<tr>
<td>$c_p$</td>
<td>specific heat at constant pressure</td>
</tr>
<tr>
<td>$c_v$</td>
<td>specific heat at constant volume</td>
</tr>
<tr>
<td>$\gamma$</td>
<td>ratio of specific heats</td>
</tr>
<tr>
<td>$h$</td>
<td>enthalpy</td>
</tr>
<tr>
<td>$c$</td>
<td>sound speed</td>
</tr>
<tr>
<td>$\beta$</td>
<td>Roe averaging weight</td>
</tr>
<tr>
<td>$p_k$</td>
<td>interpolation polynomial</td>
</tr>
<tr>
<td>$G_j$</td>
<td>known values of interpolated function</td>
</tr>
<tr>
<td>$L_j$</td>
<td>cardinal function</td>
</tr>
</tbody>
</table>
\( t \)  time
\( \Delta t \) time step
\( x \) x-position
\( z \) generic coordinate ranging from -1 to 1
\( \Delta x_R \) distance between right and center nodes
\( \Delta x_L \) distance between left and center nodes
\( [a_E] \) east coefficient matrix
\( [a_W] \) west coefficient matrix
\( [a_P] \) center coefficient matrix
\( \vec{S} \) source term
\( [A] \) flux Jacobian
\( [\hat{A}] \) averaged flux Jacobian
\( [\Lambda] \) Eigenvalues
\( [R] \) Matrix of left eigenvectors
\( [R^{-1}] \) Matrix of right eigenvectors
\( K \) artificial diffusion coefficient
\( \Delta x \) cell width
\( \Delta x_{\text{max}} \) maximum space between nodes in a cell
\( \lambda_{\text{max}} \) maximum flux Jacobian eigenvalues in a cell
\( [I] \) identity matrix
\( p \) diffusion scale factor
\( LHS \) left-hand-side
\( RHS \) right-hand-side
\( n \) number of cells
\( m \) number of unknowns per cell

**Superscripts**

* pseudo time level
n time level
Chapter 1

Introduction

Many approaches exist for computationally modeling fluid dynamics. These include the finite difference, finite element, and spectral methods to name a few. Finite element and finite difference methods are frequently used and offer a wide range of well-known numerical schemes. These schemes can vary in terms of computational accuracy but are typically of lower order. If a more accurate solution is desired, it is common practice to refine the mesh either globally or in a region of interest. This can often be a complicated or time consuming process as global mesh refinement will greatly increase the computation time while local refinement requires an elaborate refinement operation.

Alternatively, polynomial refinement has been used to improve the solution accuracy and has been shown to converge more quickly than mesh refinement in some cases [1, 2]. Figures 1.1 & 1.2 demonstrate two methods for constructing polynomial approximations. The first is an outward search which uses neighboring solution points as the unknowns which define the polynomial. This is a typical approach for finite difference methods [3]. One popular approach using the outward search for polynomial approximations are the compact difference schemes [4]. This outward search results in a series of overlapping polynomials across the domain which can become
somewhat cumbersome especially near the boundaries where nodes do not exist to construct the higher order polynomials [5]. Here, one-sided polynomial constructions must be used to maintain the order of accuracy near boundaries [6]. Also, there is little control over where the polynomial roots are located and, as will be seen later, a poor choice can result in spurious oscillations. Therefore, special techniques, such as low-pass spatial filters, must be used in order to relieve this problem [7].

Figure 1.1: Outward polynomial search.

Figure 1.2: Inward polynomial search.

Finite element methods instead increase the number of unknown values within the cell itself to construct a higher order solution using the inward search technique shown in Figure 1.2 resulting in a series of non-overlapping polynomials limited to each subdomain [8]. The internal search technique does not have the disadvantage of poorly placed polynomial roots or have the need for one-sided approximations of the outward search methods. The current study will focus on developing an internal search polynomial construction method and discuss its performance.

The solution accuracy of a method is often judged by its formal truncation error. However, a scheme with a very high formal order of accuracy will not necessarily
always produce the highest resolution [4]. Lele demonstrated a spectral-like scheme with a formal 4th order accuracy produced a much more highly resolved solution than schemes with higher formal orders of accuracy when comparing modified wave numbers. Therefore, formal order of accuracy does not provide a comprehensive basis for selecting the best solution procedure. State-of-the-art methods such as the spectral difference and discontinuous Galerkin methods fall into this category [9, 10].

Spectral methods are considered a class of solution techniques using sets of known functions to solve differential equations [11]. Basic spectral methods solve series expansions of trial functions using the method of weighted residuals (MWR). These trial functions can be truncated to the desired order of accuracy. Either collocation or Galerkin methods may be used in the method of weighted residuals, where collocation uses Dirac delta functions at collocation points as trial functions while the Galerkin method utilizes the test function as the trial function [12].

A number of drawbacks plague the practicality of spectral methods. For instance, spectral methods require a simple domain and increased resolution can only be obtained by increasing the approximation order. However, computational efficiency greatly decreases as the approximation order is increased. Additionally, increased order reduces the size of the time step that may be used to a large degree [13]. To this end researchers investigated applying spectral methods to globally decomposed domains.

Such methods are capable of obtaining a high level of accuracy and converge exponentially [11]. In the past, several spectral schemes have been proposed and demonstrated such as the spectral element methods [14], multi-domain spectral methods [13, 15], spectral volume methods [16, 17, 18, 19], and more recently the spectral difference methods [9, 20].

Patera identified that spectral and finite element methods were related and combined the two in order to obtain a generalized solution procedure with the high level
of accuracy obtained from spectral methods [14]. Here, spectral approximations were applied locally using global domain-decomposition instead of being limited to global solution procedures. Additionally, upwind values were chosen at the element interface in order to preserve a continuous solution. The spectral element procedure was applied to advection, advection-diffusion, and incompressible channel flow problems.

The implication of tying domain-decomposition together with spectral solutions increases the flexibility of the solution procedure. More complicated domains may be solved while lower order approximations are used locally increasing computational efficiency and increasing the size of the time step [13]. Although Patera’s method increased the applicability of spectral methods it was not conservative.

Kopriva and Kolias proposed a conservative staggered-grid multi-domain spectral method [21]. Here, flux values were computed on the Lobatto points while conservative quantities were defined on the Gauss points. The method defines solutions on the Lobatto and Gauss points using the Lagrange interpolating polynomial. The domain is subdivided into smaller domains which are then meshed with the staggered grid described above. Subsequently, the conservative variables and flux values are approximated using the Lagrange interpolating polynomial and substituted into the conservation equation. The flux derivative is determined by multiplying the flux variables by a derivative matrix obtained by differentiating the Lagrange polynomial. Subdomains interacted using an approximate Riemann solver such as the Roe scheme [22]. In the case of non-conforming subdomains a mortar method was applied. When refinement is used locally, it is necessary to also refine the adjacent cells for interfacial nodes to align. By extension, this required polynomial refinement across the entire domain. Kopriva’s goal was to improve flexibility by allowing for non-continuous fluxes at the cell interfaces which would eliminate the requirement that the interfacial points line up exactly allowing for purely local polynomial refinement.

To this end, an extra element (mortar cell) was introduced to reconcile the interfa-
cial values [13]. The mortar cell was used by choosing a polynomial order sufficiently large for both sides of the interface. Interface fluxes from each side were projected onto the mortar space and used to evaluate and define a new unique flux value consistent across the mortar to be transferred back to the original cell interfaces. It was necessary to determine the interfacial flux value using values upwind from the current location to satisfy outflow conditions. Projections to the mortar and back to the subdomain were performed using a least-squares procedure in order to satisfy orthogonal properties of the approximation and maintain conservation properties across the entire domain. Both methods were found to reduce the error exponentially as the grid was refined. However, the non-conforming method produced the same solution in half the computation time due to requiring only local refinement.

While the previous multi-domain method successfully produced highly refined solutions, it is difficult to extend to unstructured elements. A spectral volume method was developed in order to address this issue [16, 17, 18, 19]. This method is similar to the finite volume method in that it breaks the domain down into smaller finite volume elements. However, each finite volume was further decomposed into control volumes which allow high order polynomial reconstructions within each finite volume. The unknowns within the finite volume are the cell averaged values within the smaller control volumes. Approximate Riemann solvers were used to model the flux at the overall cell surface. However, flux values for each of the smaller control volumes are computed using the reconstruction for the entire finite volume.

The high-order unstructured solver previously discussed is somewhat complicated to implement and fairly inefficient [9]. Another method was developed that can be applied to both structured and unstructured meshes known as the spectral difference method [9, 23, 24, 20]. The procedure involves dividing the domain into smaller subdomains as in Kopriva’s method and refining the solution locally using polynomial refinement based on Gauss quadrature. Finite difference formulations were used
within each cell in order construct local high-order polynomial solutions. The scheme was shown to be conservative and solves the differential form of the conservation laws. Flux values and the unknowns were defined using polynomial interpolation at solution points. Both the flux and unknown values can be placed at the Gauss-Lobatto quadrature points in order to avoid having to perform two polynomial reconstructions, but an order of accuracy is lost. A staggered grid formulation similar to [13, 15] maintains the order of accuracy associated with the number of unknowns. Additionally flux values at cell surfaces were allowed to be discontinuous but are replaced with numerical approximations that allow for interaction between cells. The procedure for refinement using various degrees of polynomial approximations for different cells as seen in Liu et. al. [9] was used as far back as Kopriva’s work [13, 15].

The spectral difference method developed by Liu et. al. [9] provided a high level of flexibility, accuracy, and efficiency with a lower complexity than other advanced schemes such as the discontinuous Galerkin methods [10, 25, 26]. Spectral difference methods are particularly promising due to their simple implementation and ability to produce high-order solutions for unstructured grids.

Huang et. al. [1, 2] expanded the spectral difference method in a number of ways. First, spectral difference methods were applied to time integration in order to treat time in the same fashion as space and fourth-order time integration using this method was demonstrated. Additionally, the method was implicitly discretized using the delta formulation resulting in the ability to take larger time steps. A self-adaptive polynomial refinement procedure allowed the method to change levels of refinement in order to better match the exact solution. Finally, discontinuity capturing was introduced using artificial diffusion limited to regions of spurious oscillation. These improvements created a more robust polynomial refinement procedure, sharper discontinuity resolution, and allowed for larger time steps.

The method as developed by Huang et. al. [1, 2] has only been applied to scalar
one and two-dimensional problems such as Burger’s equation [1] or more recently combustion problems [27]. For the current article, Huang’s method is further expanded for a system of equations and solves gas dynamics problems using the Euler equations. A shock capturing method for the Euler equations is incorporated by defining artificial diffusion guidelines and selectively applying the diffusion term where it is needed. Additionally, an alternative to the Roe Scheme was used for cell interfaces by upwinding the characteristic values crossing the interface. Adaptive polynomial refinement is employed. The objective of the current study is to develop a state-of-the-art scheme for the compressible Navier-Stokes equations that is capable of an infinite order of accuracy and adaptively refines polynomial approximations where needed. High-order polynomial reconstructions are achieved across discontinuities without resorting to one-sided stencils or reducing to low order [28].
Chapter 2

Methodology

Each step of scheme’s development is thoroughly addressed in order for this work to be self-contained and easily followed. The discussion will begin with a brief description of the model equation and how it is modified and incorporated in the present work and then components of the method will be addressed in detail.

The compressible Navier-Stokes equations may be greatly simplified in certain circumstances by ignoring viscous interactions reducing the system of equations to the Euler equations which are a coupled system of nonlinear hyperbolic partial differential equations. The Euler equations are often used to model high-Reynolds number flows where boundary layer development is considered insignificant to the overall flowfield. The Euler equations may be written in conservation form for one-dimension as shown in Eqns. 2.1 and 2.2. The inviscid Navier-Stokes equations are heavily investigated in the current study because the hyperbolic limit of the compressible Navier-Stokes is equations is often the most difficult to tackle.

\[
\frac{\partial \vec{Q}}{\partial t} + \frac{\partial \vec{F}}{\partial x} = 0
\]

(2.1)
\[
\vec{Q} = \begin{bmatrix}
\rho \\
\rho u \\
\rho e
\end{bmatrix}, \quad \vec{F} = \begin{bmatrix}
\rho u \\
P + \rho u^2 \\
\rho (e + P/\rho)
\end{bmatrix}
\] (2.2)

2.1 Linearized Euler Equations

Eqn. 2.1 is a non-linear hyperbolic system of equations, but can be greatly simplified using a linear approximation given in Eqn. 2.3,

\[
\frac{\partial \vec{Q}}{\partial t} + [A] \frac{\partial \vec{Q}}{\partial x} = 0 \tag{2.3}
\]

where \([A]\) is the Jacobian of the flux vector given as,

\[
[A] = \frac{\partial F_i}{\partial Q_j} \tag{2.4}
\]

and the flux vector can be written,

\[
\vec{F} = [A] \vec{Q} \tag{2.5}
\]

To find the flux Jacobian, the conservative variables may be rewritten as,

\[
\vec{Q} = \begin{bmatrix}
\rho \\
M \\
E
\end{bmatrix} \tag{2.6}
\]

Eqns. 2.7-2.14 are used to write the flux values in terms of the conservative variables only.

\[
P = \rho RT \tag{2.7}
\]
\[ R = c_P - c_v \]  
\[ (2.8) \]

\[ c_v = \frac{R}{(\gamma - 1)} \]  
\[ (2.9) \]

\[ c_p = \frac{\gamma R}{(\gamma - 1)} \]  
\[ (2.10) \]

\[ \gamma = \frac{c_P}{c_v} \]  
\[ (2.11) \]

\[ e = c_v T + \frac{1}{2} u^2 \]  
\[ (2.12) \]

\[ E = \rho e \]  
\[ (2.13) \]

\[ h = e + \frac{P}{\rho} \]  
\[ (2.14) \]

Resulting in,

\[
\vec{F} = \begin{bmatrix}
M \\
(\gamma - 1) E + \frac{M^2}{2\rho} (3 - \gamma) \\
\frac{ME\gamma}{\rho} - \frac{M^3}{2\rho^2} (\gamma - 1)
\end{bmatrix}
\]  
\[ (2.15) \]

The partial derivative of each flux value is taken with respect to each of the
conservative variables yielding the elements of the Jacobian matrix in Eq. 2.4.

\[
[A] = \begin{bmatrix}
0 & 1 & 0 \\
-\frac{M^2}{2\rho^2} (3 - \gamma) & \frac{M}{\rho} (3 - \gamma) & (\gamma - 1) \\
-\frac{ME\gamma}{\rho^2} + \frac{M^3}{\rho^2} (\gamma - 1) & \frac{E\gamma}{\rho} - \frac{3}{2} \frac{M^2}{\rho^2} (\gamma - 1) & \frac{M\gamma}{\rho}
\end{bmatrix}
\]  (2.16)

The sound speed is defined in Eqn. 2.17 which may be converted to Eqn. 2.18 for convenience using Eqns. 2.7-2.14.

\[
c = \sqrt{\frac{P}{\rho}}
\]  (2.17)

\[
c = \sqrt{(\gamma - 1) \left( h - \frac{1}{2} u^2 \right)}
\]  (2.18)

The Jacobian is then written in terms of only two state variables by simplifying each term to functions of \( u \) and \( c \) using Eqn. 2.18. This results in,

\[
[A] = \begin{bmatrix}
0 & 1 & 0 \\
\frac{1}{2} (\gamma - 3) u^2 & -(\gamma - 3) u & (\gamma - 1) \\
-\frac{u^2}{\gamma - 1} + \frac{u^3}{2} (\gamma - 2) & \frac{c^2}{\gamma - 1} + \frac{u^2}{2} (3 - 2\gamma) & u\gamma
\end{bmatrix}
\]  (2.19)

The matrix \([A]\) is diagonalized by splitting the matrix into its left and right eigenvectors and eigenvalues,

\[
[A] = [R] [\Lambda] [R]^{-1}
\]  (2.20)

The left Eigenvectors are given as,

\[
[R] = \begin{bmatrix}
-\frac{1}{c^2} & \frac{1}{2c^2} & \frac{1}{2c^2} \\
-\frac{u}{c^2} & \frac{u+c}{2c^2} & \frac{u-c}{2c^2} \\
-\frac{u^2}{2c^2} + \frac{u^2}{4c^2} + \frac{1}{2(\gamma - 1)} & \frac{u^2}{4c^2} - \frac{u}{2c} + \frac{1}{2(\gamma - 1)}
\end{bmatrix}
\]  (2.21)
where \( [R] \) can be further decomposed into two submatrices \([R_1]\) and \([R_2]\). This decomposition decreases the number of calculations required in matrix multiplication operations and improves computational efficiency [29].

\[
[R] = [R_1] [R_2]
\] (2.22)

\[
[R_1] = \begin{bmatrix}
1 & 0 & 0 \\
u & 1 & 0 \\
\frac{u^2}{2} & u & \frac{1}{(\gamma-1)}
\end{bmatrix}
\] (2.23)

\[
[R_2] = \begin{bmatrix}
-\frac{1}{c^2} & \frac{1}{2c^2} & \frac{1}{2c^2} \\
0 & \frac{1}{2c} & -\frac{1}{2c} \\
0 & \frac{1}{2} & \frac{1}{2}
\end{bmatrix}
\] (2.24)

Similarly, the right eigenvectors are given as,

\[
[R]^{-1} = \begin{bmatrix}
\frac{1}{2} (\gamma - 1) u^2 - c^2 & u (1 - \gamma) & (\gamma - 1) \\
\frac{1}{2} (\gamma - 1) u^2 - uc & u (1 - \gamma) + c & (\gamma - 1) \\
\frac{1}{2} (\gamma - 1) u^2 + uc & u (1 - \gamma) - c & (\gamma - 1)
\end{bmatrix}
\] (2.25)

where \([R]^{-1}\) can be decomposed into submatrices as well.

\[
[R]^{-1} = [T_2] [T_1]
\] (2.26)

\[
[T_1] = \begin{bmatrix}
1 & 0 & 0 \\
-u & 1 & 0 \\
\frac{1}{2} u^2 (\gamma - 1) & u (1 - \gamma) & (\gamma - 1)
\end{bmatrix}
\] (2.27)
\[ [T_2] = \begin{bmatrix} -c^2 & 0 & 1 \\ 0 & c & 1 \\ 0 & -c & 1 \end{bmatrix} \] (2.28)

The eigenvalues resulting from the diagonalization are given in Eqn. 2.29.

\[ [\Lambda] = \begin{bmatrix} u & 0 & 0 \\ 0 & u + c & 0 \\ 0 & 0 & u - c \end{bmatrix} \] (2.29)

### 2.2 Delta Formulation

The linearized equations are now cast into delta form using the implicit methods of Beam and Warming [30] and Newton-like subiterations to preserve time-accurate solutions. The value of \( \Delta Q \) is given in Eqn. 2.30 where the superscript \( * \) denotes the iterative approximation of the terms where it is applied. When \( * = 1 \), \( \bar{Q}^* = \bar{Q}^n \) and \( \bar{Q}^* \to \bar{Q}^{n+1} \) as \( * \to \infty \).

\[
\Delta \bar{Q} = \bar{Q}^{*+1} - \bar{Q}^* 
\] (2.30)

\[
\frac{\partial \bar{Q}}{\partial t} = \frac{\bar{Q}^{n+1} - \bar{Q}^n}{\Delta t} 
\] (2.31)

\[
\frac{\partial \bar{Q}}{\partial t} = -\frac{\partial \bar{F}}{\partial x} 
\] (2.32)

For 1st-order accuracy in time, 1st-order backward-Euler time integration may be used (Eqn. 2.31). Substituting Eqn. 2.30 into Eqns. 2.31 \& 2.32 produces Eqns. 2.33 and 2.34. Setting these two equations equal to each other results in the delta
formulation of the Euler equations for 1st-order accuracy in time 2.35.

\[
\frac{\partial \vec{Q}}{\partial t} = \frac{\Delta \vec{Q} + \vec{Q}^* - \vec{Q}^n}{\Delta t} 
\] (2.33)

\[
\frac{\partial \vec{Q}}{\partial t} = \left( \frac{\partial \Delta \vec{Q}}{\partial x} + \frac{\partial F^*}{\partial x} \right) 
\] (2.34)

\[
\frac{\Delta \vec{Q}}{\Delta t} + [A] \frac{\partial \Delta \vec{Q}}{\partial x} = - \left( \frac{\vec{Q}^* - \vec{Q}^n}{\Delta t} + \frac{\partial F^*}{\partial x} \right) 
\] (2.35)

The flux derivative on the RHS of 2.35 is solved explicitly and represents the physics of the original PDE where high-order approximations to this term are easily applied. The LHS is solved implicitly and discretization only requires numerical stability. Here, a 1st-order upwind scheme is applied for the convective fluxes resulting in Eqn. 2.36 and 2nd-order central differences will be applied later for viscous flux terms.

\[
[A] \frac{\partial \Delta \vec{Q}}{\partial x} = \frac{1}{2} ([A_P] + ||A_P||) \frac{\Delta \vec{Q}_P - \Delta \vec{Q}_W}{\Delta x_L} + \frac{1}{2} ([A_P] - ||A_P||) \frac{\Delta \vec{Q}_E - \Delta \vec{Q}_P}{\Delta x_R} 
\] (2.36)

\[
[a_P] \Delta \vec{Q}_P + [a_W] \Delta \vec{Q}_W + [a_E] \Delta \vec{Q}_E = \vec{S} 
\] (2.37)

The coefficients given above are derived by grouping like terms from Eqn. 2.36 substituted into Eqn. 2.35. The resulting coefficient values are given below in Eqns. 2.38-2.41 where the absolute value of the flux Jacobian is determined using the magnitude of the eigenvalues.

\[
[a_W] = -\frac{1}{2\Delta x_L} [R] ([A_P] + ||A_P||) [R^{-1}] 
\] (2.38)
\[
[a_E] = \frac{1}{2\Delta x_R} [R] ([\Delta_P] - [|\Delta_P|]) [R^{-1}]
\] (2.39)

\[
[a_P] = \frac{[I]}{\Delta t} - ([a_W] + [a_E])
\] (2.40)

\[
\vec{S} = - \left( \frac{\vec{Q}^* - \vec{Q}^n}{\Delta t} \right) - \frac{\partial \vec{F}^*}{\partial x}
\] (2.41)

The nature of the delta scheme allows the LHS of the above equations to be 1st-order accurate without affecting the overall accuracy of the solution which is determined by how the RHS of the equation is treated. The formulation presented above is diagonally dominant and unconditionally stable for iterative solution procedures. The RHS contains the real physics and is updated in an iterative fashion through changes in the primitive variables. The flux gradients are determined using the methods described later in this paper.

A 2nd-order in time scheme is obtained by replacing Eqn. 2.31 with the 2nd-order backward Euler approximation (Eqn. 2.42) and applying the same steps as above. This results in Eqn. 2.43 where only the coefficients \([a_p]\) and \(\vec{S}\) are changed to Eqns. 2.44 & 2.45 respectively. At this point the discussion for the LHS is complete and we will now discuss the high-order treatment of the RHS.

\[
\frac{\partial \vec{Q}}{\partial t} = \frac{3\vec{Q}^{n+1} - 4\vec{Q}^n + \vec{Q}^{n-1}}{2\Delta t}
\] (2.42)

\[
\frac{3}{2} \Delta \vec{Q} + [A] \frac{\partial A\vec{Q}}{\partial x} = - \left( \frac{3\vec{Q}^* - 4\vec{Q}^n + \vec{Q}^{n-1}}{2\Delta t} \right) - \frac{\partial \vec{F}^*}{\partial x}
\] (2.43)

\[
[a_P] = \frac{3}{2} \frac{[I]}{\Delta t} - ([a_W] + [a_E])
\] (2.44)

15
\[ S = - \left( \frac{3\bar{Q}^* - 4\bar{Q}^n - \bar{Q}^{n-1}}{2\Delta t} \right) - \frac{\partial \bar{F}^*}{\partial x} \] (2.45)

### 2.3 Polynomial Approximation

Previously, a number of methods for achieving high-order approximations were discussed in Chapter 1. These methods incorporated the use of larger polynomial stencils to achieve the high-order approximation by either searching outside of the cell or increasing the number of unknowns within the cell. For the highest order of approximation, every node is included for the solution of each unknown as is done in basic spectral methods. This is accomplished here by reconstructing the solution using a series of polynomials shown in Eqn. 2.46.

\[ p_{k-1}(x) = \sum_{j=1}^{k} [L_j(x)G_j] \] (2.46)

\( L_j(x) \) is the shape function while \( G_j \) consists of known values located on the shape function roots, and \( k \) refers to the number of points used in the approximation. This definition results in a \( k-1 \) degree approximation for the polynomial at the given value for \( x \). The shape function is formed in Eqn. 2.47 using Newton’s divided differences, while the placement of the roots uniquely defines the shape function [31].

\[ L_j(x) = \prod_{m=1, m\neq j}^{k} \left( \frac{x - x_m}{x_j - x_m} \right) \] (2.47)

The conservative values are reconstructed in a similar manner using Eqn. 2.48 for the flux values using Eqn. 2.49 and rewriting \( \bar{Q} \) as \( Q_i \) (\( i = 1, 2, 3 \)).

\[ Q_i(x) = \sum_{j=1}^{k} \left[ L_j(x) (Q_i)_j \right] \] (2.48)
\[ F_i(x) = \sum_{j=1}^{k} \left[ L_j(x) (F_i)_j \right] \quad (2.49) \]

The shape function can be differentiated as shown in Eqn. 2.50 and used to replace \( L_j(x) \) in Eqn. 2.46 to approximate the polynomial derivative as shown in Eqn. 2.51[1].

\[
\frac{\partial L_j(x)}{\partial x} = \sum_{q=1, q \neq j}^{k} \left[ \prod_{i=1, i \neq j, i \neq q}^{k} \frac{x - x_i}{x_j - x_i} \right] \quad (2.50)
\]

\[
\frac{\partial p_{k-1}(x)}{\partial x} = \sum_{j=1}^{k} \left[ \left( \frac{\partial L_j}{\partial x} \right) G_j \right] \quad (2.51)
\]

We will now note that the above formulation for the function derivative can be written as the summation of cardinal function values multiplied by function values at the roots in Eqn. 2.52.

\[
\frac{\partial p_{k-1}(x)}{\partial x} = \frac{\partial L_1}{\partial x} (x) G_1 + \frac{\partial L_2}{\partial x} (x) G_2 + \ldots + \frac{\partial L_{k-1}}{\partial x} (x) G_{k-1} + \frac{\partial L_k}{\partial x} (x) G_k \quad (2.52)
\]

If we choose values for \( x \) corresponding with the shape function roots of degree \( n - 1 \), Eqn. 2.52 becomes the linear system of equations in Eqn. 2.53. The shape function derivative values are represented by an \( n \times n \) matrix which is only dependent on the choice of the roots. Therefore, the \( n \times n \) coefficient matrix is calculated in advance and stored in order to reduce computation time. Higher-order differentiation is obtained by consecutively applying the shape function derivative to the previously calculated derivatives as demonstrated in Eqn. 2.54.

Similar to Eqn. 2.53, Eqn. 2.46 can be written in matrix form in order to transfer from a polynomial of degree \( k - 1 \) to a polynomial of degree \( m - 1 \) resulting in Eqn. 2.55. The coefficient matrix becomes an \( m \times n \) matrix and stored in advance.
as file input. This operation provides the mechanism for transition between various polynomial degree approximations. However, because there is nearly an infinite number of polynomial transitions that can be made, the coefficient matrices computed in advance need to be chosen as the most useful for the solution procedure.

\[
\begin{bmatrix}
\frac{\partial p}{\partial x} |_1 \\
\frac{\partial p}{\partial x} |_2 \\
\vdots \\
\frac{\partial p}{\partial x} |_{k-1} \\
\frac{\partial p}{\partial x} |_k
\end{bmatrix} =
\begin{bmatrix}
\frac{\partial L_1}{\partial x} (x_1) & \frac{\partial L_2}{\partial x} (x_1) & \ldots & \frac{\partial L_{k-1}}{\partial x} (x_1) & \frac{\partial L_k}{\partial x} (x_1) \\
\frac{\partial L_1}{\partial x} (x_2) & \frac{\partial L_2}{\partial x} (x_2) & \ldots & \frac{\partial L_{k-1}}{\partial x} (x_2) & \frac{\partial L_k}{\partial x} (x_2) \\
\vdots & \vdots & \ldots & \vdots & \vdots \\
\frac{\partial L_1}{\partial x} (x_{k-1}) & \frac{\partial L_2}{\partial x} (x_{k-1}) & \ldots & \frac{\partial L_{k-1}}{\partial x} (x_{k-1}) & \frac{\partial L_k}{\partial x} (x_{k-1}) \\
\frac{\partial L_1}{\partial x} (x_k) & \frac{\partial L_2}{\partial x} (x_k) & \ldots & \frac{\partial L_{k-1}}{\partial x} (x_k) & \frac{\partial L_k}{\partial x} (x_k)
\end{bmatrix}
\begin{bmatrix}
G_1 \\
G_2 \\
\vdots \\
G_{k-1} \\
G_k
\end{bmatrix}
\]  

(2.53)

\[
\frac{\partial^2 p}{\partial x^2} = \sum_{j=1}^{k} \left[ \left( \frac{\partial L_j}{\partial x} \right) \frac{\partial G}{\partial x} \right]_{j}
\]  

(2.54)

\[
\begin{bmatrix}
p_1 \\
p_2 \\
\vdots \\
p_{m-1} \\
p_m
\end{bmatrix} =
\begin{bmatrix}
L_1 (x_1) & L_2 (x_1) & \ldots & L_{k-1} (x_1) & L_k (x_1) \\
L_1 (x_2) & L_2 (x_2) & \ldots & L_{k-1} (x_2) & L_k (x_2) \\
\vdots & \vdots & \ldots & \vdots & \vdots \\
L_1 (x_{k-1}) & L_2 (x_{k-1}) & \ldots & L_{k-1} (x_{k-1}) & L_k (x_{k-1}) \\
L_1 (x_k) & L_2 (x_k) & \ldots & L_{k-1} (x_k) & L_k (x_k)
\end{bmatrix}
\begin{bmatrix}
G_1 \\
G_2 \\
\vdots \\
G_{n-1} \\
G_n
\end{bmatrix}
\]  

(2.55)
2.4 Lobatto Formulation

The choice of the roots is an important consideration as it uniquely defines the shape function. Any number of distributions can be chosen, including uniformly spaced roots. One difficulty in using polynomial refinement is the occurrence of Runge phenomenon which is the result of oscillations at the end points of an interpolation when high-degree polynomials are applied over a region of space [31]. The Runge phenomenon may be diminished by choosing a non-uniform placement for the unknown values or roots. Several choices exist that greatly diminish or eliminate the Runge phenomenon. These require a non-uniform placement for the unknown values. For the current study, the Lobatto formulation is a convenient choice because it includes the endpoints of the interval [13].

The Lobatto points are determined by finding the roots of $P_{k-1}'(x)$, where $P_k(x)$ is the Legendre polynomial of degree $k$ on the interval $[-1, 1]$. The actual region of space solved using Gauss-Lobatto integration will likely not be on the reference interval and a change of coordinates is necessary in order to produce actual distributions. Eqn. 2.46 becomes Eqn. 2.56 where the polynomial derivative is over the basic interval. This derivative is transferred back to the original interval using the coordinate transformation in Eqns. 2.57 & 2.58 where $x_{cv,L}$ and $x_{cv,R}$ are the left and right limits of the domain and $z$ is the location of the $k$ roots on the interval $[-1, 1]$.

\[
\frac{\partial p_{k-1}}{\partial z} = \sum_{j=1}^{k} \left[ \left( \frac{\partial L_j}{\partial z} \right) G_j \right] \quad (2.56)
\]

\[
\frac{\partial p}{\partial x} = \frac{\partial p}{\partial z} \frac{\partial z}{\partial x} \quad (2.57)
\]

\[
\frac{\partial z}{\partial x} = \frac{2}{x_{cv,R} - x_{cv,L}} \quad (2.58)
\]
2.5 Global Domain Decomposition

The discussion so far has been general for a single one dimensional domain. While single-domain systems are useful for solving simple geometry problems, this is often not the case in two and three dimensional flow problems. Kopriva demonstrated that the global domain can be broken into smaller subdomains in order to solve more complicated geometries [13, 15]. Figure 2.1 shows how a single domain formulation can be broken into a multi-domain problem. The method described in the previous sections is now applied to each subdomain individually. The terms cell, subdomain, and local domain are used interchangeably in the current work.

2.6 Boundary Conditions

Two types of boundary conditions are now necessary due to the multi-domain formulation: global and local. The global boundary conditions refer to the conditions at the global domain edges which are simply the physical boundary conditions. The boundary conditions here may be set implicitly or explicitly, but are defined explicitly in the current work.

The other boundary type is the local boundary condition, which is necessary as each subdomain requires an independent solution procedure. Implicit methods such
as the block tri-diagonal solver or Gauss-Seidel iterations can be applied to each cell, but the interaction with the adjacent cell must be accounted. This is handled by extending the cell’s stencil to the first internal node of the adjacent cells on both sides as shown in Figure 2.2. These external nodes serve as the boundary conditions to the local domain. The value $\Delta \vec{Q}$ refers to the change in conservative values and when converged goes to 0 for each time step. Therefore, the local domain utilizes Dirichlet boundary conditions that are incorporated directly into the coefficients for the edge nodes. While the stencil extends to adjacent cell nodes, only the internal and edge nodes require computation.

![Extended cell stencil](image)

Figure 2.2: Extended cell stencil.

Incorporating the Dirichlet local boundary conditions, the left edge coefficients become Eqns. 2.59-2.64.

$$[a_E] = \frac{1}{2\Delta x_R} [R] ([\Lambda_P] - [||\Lambda_P||]) [R^{-1}]$$  \hfill (2.59)

$$[a_P] = \frac{[I]}{\Delta t} - [a_W]$$  \hfill (2.60)

$$\vec{S} = - \left( \frac{\vec{Q}^* - \vec{Q}^n}{\Delta t} \right) - \frac{\partial \vec{F}^*}{\partial x}$$  \hfill (2.61)
The right edge local boundary condition becomes,

$$[a_W] = -\frac{1}{2\Delta x_L} [R] ([A_P] + ||A_P||) [R^{-1}]$$  \hspace{1cm} (2.62)

$$[a_P] = \frac{[I]}{\Delta t} - [a_E]$$  \hspace{1cm} (2.63)

$$\vec{S} = -\left(\frac{\tilde{Q}^* - \tilde{Q}^n}{\Delta t}\right) - \frac{\partial \vec{F}^*}{\partial x}$$  \hspace{1cm} (2.64)

### 2.7 Interface Flux Formulation

Earlier, the Lobatto formulation was mentioned as a convenient distribution for the unknown values without much explanation as to why. This becomes clear when discussing the definitions for the interface flux values. The flux terms at the subdomain interfaces need to be treated differently in order to allow waves to propagate through the interfaces. Furthermore, the normal component of the flux must be consistent across the interface to maintain conservation. A number of flux difference and flux vector splitting methods have been proposed and used over the years [3]. One class of resolving shock waves is flux-vector splitting techniques. These techniques generally use the eigenvalue structure of the problem to determine how to appropriately model the flux at the control volume surface and satisfy conservation laws [32]. Flux vectors are split into two components based on the sign of the eigenvalue at the point in question. Such techniques are Steger-Warming splitting and van Leer flux splitting [3].

Another class of techniques is flux-difference splitting. This class of shock capturing attempts to solve the Riemann problem at the control volume face. Both full and approximate Riemann solvers exist, however approximate solvers are more commonly
used because linear systems of equations are simpler to solve. Probably the most popular among flux-difference splitting techniques is Roe’s approximate Riemann solver [22].

2.7.1 Roe Scheme

The Roe scheme is an approximate Riemann solver which models Euler equations as a Riemann problem. A linear approximation is applied which greatly simplifies the solution procedure. Elements of the flux Jacobian resulting from the linearization are determined using a weighted average of the properties on both sides of the interface [22]. This method is capable of resolving discontinuities and is easily implemented. A 1st-order upwind scheme is developed using the Roe scheme on a cell-centered finite-volume mesh. However, in the present study, the method is applied only at the
subdomain interfaces allowing for higher order solutions to be constructed within the cell. A brief discussion of the Roe scheme is presented below.

The Roe scheme makes a linear approximation of the flux Jacobian associated with nonlinear hyperbolic equations which is accomplished by using a weighted averaging technique.

\[
\beta = \sqrt{\frac{\rho_R}{\rho_L}} \quad (2.65)
\]

\[
\bar{\rho} = \rho_L \beta \quad (2.66)
\]

\[
\bar{u} = \frac{u_L + \beta u_R}{1 + \beta} \quad (2.67)
\]

\[
\bar{h} = \frac{h_L + \beta h_R}{1 + \beta} \quad (2.68)
\]

\[
\bar{c} = \sqrt{(\gamma - 1) \left( \bar{h} - \frac{1}{2} \bar{u}^2 \right)} \quad (2.69)
\]

Earlier, the flux Jacobian was written in terms of only the primitive values \(u\) and \(c\). Now, this definition becomes convenient as the Roe averaged flux Jacobian are written in terms of only \(\bar{u}\) and \(\bar{c}\). The flux at the interface may then be defined as:

\[
\vec{F}_i = \frac{1}{2} \left[ \vec{F}_R + \vec{F}_L - [|\hat{A}|] \left( \vec{Q}_R - \vec{Q}_L \right) \right] \quad (2.70)
\]

where \([|\hat{A}|]\) is the result of \([|\hat{A}|] = [R][|\hat{A}|][R]^{-1}\)

\[
[|\hat{A}|] = \begin{bmatrix}
|\hat{A}_1| & 0 & 0 \\
0 & |\hat{A}_2| & 0 \\
0 & 0 & |\hat{A}_3|
\end{bmatrix} \quad (2.71)
\]
Subscripts L and R refer to the values on the left and right sides of the interface respectively.

One disadvantage of the Roe scheme for the current method is the difficulty in determining the corresponding vector of conservative values from the fluxes. This difficulty arises as a result of the inability to invert the Jacobian matrix at sonic points.

2.7.2 Upwind Characteristics

An alternative flux splitting method was used to pass the flux values across a cell interface. This method involves calculating the characteristic values on the left and right sides of the cell interface and choosing the upwind characteristic values based on the averaged eigenvalues across the interface. Roe averaging and arithmetic averaging on primitive variables perform similarly in this case. Eqns. 2.72-2.76 show the arithmetic averaging for the primitive variables.

\[
\bar{\rho} = \frac{1}{2} (\rho_R + \rho_L) \tag{2.72}
\]

\[
\bar{u} = \frac{1}{2} \left( m_R + m_L \right) \frac{\bar{\rho}}{\bar{\rho}} \tag{2.73}
\]

\[
\bar{E} = \frac{1}{2} (E_R + E_L) \tag{2.74}
\]

\[
\bar{P} = (\gamma - 1) \left( \bar{E} - \frac{1}{2} \bar{\rho} \bar{u}^2 \right) \tag{2.75}
\]

\[
\bar{c} = \sqrt{\frac{\gamma \bar{P}}{\bar{\rho}}} \tag{2.76}
\]

For a simple 1st-order case, cells may be drawn as shown below in Figure 2.4.
We can recast the linearized Euler equations into a decoupled system of equations by multiplying the right eigenvectors in Eqns. 2.77-2.79 producing the decoupled system of equations in Eqn. 2.80. The quantity resulting from the operation in Eqn. 2.79 is the vector of characteristic variables.

\[
\begin{align*}
\frac{\partial \vec{Q}}{\partial t} + [R][\Lambda][R^{-1}] \frac{\partial \vec{Q}}{\partial x} &= 0 \quad (2.77) \\
[R^{-1}] \frac{\partial \vec{Q}}{\partial t} + [\Lambda][R^{-1}] \frac{\partial \vec{Q}}{\partial x} &= 0 \quad (2.78) \\
\vec{W} &= [R^{-1}]\vec{Q} \quad (2.79)
\end{align*}
\]

We will now determine the flux at the cell interface by choosing the upwind characteristic value using Eqns. 2.81 & 2.82 based on the sign of the Eigenvalues and recalculating the flux using Eqn. 2.83 where \( \vec{W}_i \) is the upwinded characteristic values.

\[
\frac{\partial \vec{W}}{\partial t} + [\Lambda] \frac{\partial \vec{W}}{\partial x} = 0 \quad (2.80)
\]

\[
\vec{W}_R = [R^{-1}]\vec{Q}_L \quad (2.81)
\]
\[
\vec{W}_L = [R^{-1}] \vec{Q}_R
\]  

(2.82)

\[
\vec{F}_i = [R][\Lambda] \vec{W}_i
\]  

(2.83)

This process may be succinctly combined into a single equation in Eqn. 2.84 where the primitive values of Eqns. 2.72-2.76 are used to calculate \([\hat{R}]\) and \([\hat{\Lambda}]\).

\[
\vec{F}_i = \frac{1}{2}[\hat{R}] \left( \left( [\hat{\Lambda}] - ||\hat{\Lambda}|| \right) \vec{W}_R + \left( [\hat{\Lambda}] + ||\hat{\Lambda}|| \right) \vec{W}_L \right)
\]  

(2.84)

The primary difference between the Roe scheme and the upwind characteristic methods lies in the fact that the Roe scheme uses the flux values calculated directly on both sides of the interface whereas the present method decouples the flux to apply upwinding directly to the characteristic variable. Here, characteristic variables are conformed as opposed to flux values and the corresponding conservative variables are much more easily determined at sonic points because no inversion of the Jacobian matrix is necessary.

A comparison of the 1st order solution for both the Roe scheme and the present method will be shown later in order to validate the use of upwind characteristic variables.

### 2.7.3 Rusanov Solver

A third method employed in the solver is Rusanov’s scheme [33]. Like the Roe scheme, the interface flux is modified by taking the average of the flux values at the interface and incorporating a diffusive contribution. Here, the diffusive contribution is composed of the maximum possible contribution from the wave speeds (where the Roe scheme used a combination of each wave speed). The Rusanov solver is given in
Eqns. 2.85 & 2.86 for 1D cases.

\[ \vec{F}_i = \frac{1}{2} [(\vec{F}_L + \vec{F}_R) - \lambda(\vec{Q}_R - \vec{Q}_L)] \] (2.85)

\[ \lambda = \frac{1}{2}(|u_L| + |u_R| + c_L + c_R) \] (2.86)

2.8 Adaptive Polynomial Refinement

The variable nature of the current method opens the door to several benefits provided by polynomial interpolation. That is, the ability to transfer between varying degrees of approximation. If a subdomain contains a poor solution, the polynomial order can be increased to enhance the local accuracy. The converse is true as well. If a smaller approximation becomes sufficient in a constant or smooth region, the polynomial degree is reduced locally to save computation time in future time steps. Such a process is known as adaptive polynomial refinement. Several criteria for determining where polynomial refinement is needed were developed and tested for the inviscid Navier-Stokes equations based on the methods of Huang et. al [1, 2].

2.8.1 Q-Refinement Switch

The most direct extension of the methods of Huang et. al [1, 2] is to apply the same method independently to each of the conservation equations and set a tolerance criteria that must be met to trigger an increase in the polynomial approximation. This method requires the conservative variables to be known from the previous time step. These known values are interpolated to two reference polynomial approximations. The area under the 2nd derivative magnitude is compared and if significantly different, refinement is activated. Here Eqns. 2.51 & 2.54 are directly applied to the conservative variables in order to find the first and second derivatives in Eqns. 2.87
& 2.88, where \( i \) refers to the vector quantity and \( j \) refers to the nodal position.

\[
\frac{\partial Q_i}{\partial x} = \sum_{j=1}^{k} \left( \frac{\partial L_j}{\partial x} \right) (Q_i)_j
\]

(2.87)

\[
\frac{\partial^2 Q_i}{\partial x^2} = \sum_{j=1}^{k} \left( \frac{\partial L_j}{\partial x} \right) \left. \frac{\partial Q_i}{\partial x} \right|_j
\]

(2.88)

The 2nd derivative is found due to its sensitivity to change. Higher order derivatives may be used if more sensitivity is necessary. However, the second derivative was found to be sufficiently sensitive for refinement and using higher derivatives would be computationally inefficient. The magnitude of the 2nd derivative is then integrated and compared for both levels of refinement and compared in Eqns. 2.89 & 2.90. If little to no change is found the polynomial approximation remains unchanged.

\[
D_k = \sum_{i=1}^{3} \int_0^{\Delta x} \left| \frac{\partial^2 Q_i}{\partial x^2} \right| dx
\]

(2.89)

\[
\left| 1 - \frac{D_m}{D_k} \right| < \epsilon
\]

(2.90)

The integral above is evaluated using Gaussian quadrature for each conservation law. The value of \( \epsilon \) is chosen as 0.003 and \( \Delta x \) denotes the integration is over the region of the subdomain. The value of \( k \) refers to the lower degree polynomial while \( m \) is the higher order approximation. This process is repeated until refinement produces a satisfactory agreement with the given tolerance or the maximum of the set of available polynomial approximations is reached. Due to increasing sensitivity of this criteria as values of \( D_k \) approach 0, regions with near zero slope may be falsely triggered. This issue is relieved by deactivating refinement if the value of \( D_k \) is below \( 1.0 \epsilon - 6 \).
2.8.2 Pressure Refinement Switch

While the definition for polynomial refinement discussed previously is sufficient for the current application, it is needlessly inefficient due to the process being applied to three sets of equations. Alternatively, the same process can be applied a single variable, namely pressure as a requirement for polynomial refinement. Similar to above, the 2nd derivative of pressure is found using Eqns. 2.91 & 2.92 and a similar comparison is performed using Eqns. 2.93 & 2.90. Refinement is deactivated in regions of constant value.

\[
\frac{\partial P}{\partial x} = \sum_{j=1}^{k} \left( \frac{\partial L_j}{\partial x} \right) Q_j \tag{2.91}
\]

\[
\frac{\partial^2 P}{\partial x^2} = \sum_{j=1}^{k} \left( \frac{\partial L_j}{\partial x} \right) \frac{\partial P}{\partial x} \bigg|_{j} \tag{2.92}
\]

\[
D_k = \int_{0}^{\Delta x} \left| \frac{\partial^2 P}{\partial x^2} \right| dx \tag{2.93}
\]

While this method is a computationally cheaper approach it is less discriminating than the previous definition and requires a lower tolerance of \( \epsilon = 1e - 4 \).
2.8.3 The Refinement Process

![Block diagram showing the refinement process](image)

Figure 2.5: This block diagram shows how polynomial refinement is incorporated into the solution procedure.

Figure 2.5 demonstrates how the refinement process is incorporated with the general solution procedure. Essentially, the method begins by setting the initial conditions on a baseline mesh. The author prefers to initialize the solution on the highest level of refinement to maintain a more precise starting point. Next, the conservative values are run through an initial level of refinement and the polynomial approximations in each cell are adjusted accordingly. This solution is stored as the reference old solution \(Q_{r}^{k-1}\). The solution procedure described in Sections 2.1 - 2.7 is carried out to produce the new solution \(Q^{k}\) on the current level of refinement. \(Q^{k}\) is then run through the refinement check and the new resolution for each cell is stored. If the level or resolution in each cell remains the same then the process ends and the current time step is finalized. \(Q^{k}\) is then stored to \(Q_{r}^{k-1}\) and the process repeats for a new time step. However, if the level of refinement in any cell changes then \(Q^{k}\) is discarded while \(Q^{k-1}\) is redefined by interpolation from \(Q_{r}^{k-1}\) on the new refinement level and
the solution process is repeated. The refinement process generally only requires one or two iterations, but occasionally will become stuck between two levels of resolution. In this instance, the number of polynomial refinement iterations is limited to 5.

2.9  Discontinuity Capturing

Removing Gibbs phenomenon, the appearance of spurious oscillations when a continuous function is used to model a discontinuity or a very steep regime, is no trivial task when using high order polynomial reconstructions [11]. Treating the Gibbs phenomenon is an important consideration in solving the Euler equations as many cases of interest involve steep gradients or discontinuities. Huang et. al (2005) preferred the use of an artificial viscosity term to eliminate Gibbs phenomena over the use of flux limiters [1, 2]. The use of a flux limiter reduces the accuracy to 2nd order and can risk the elimination of physical extrema while diffusion can be used instead to dissipate oscillations.

2.10  Artificial Diffusion

Inclusion of nonphysical dissipation terms is probably one of the oldest and most basic shock capturing methods and has been used in a number of schemes to relieve oscillations produced by shocks [34]. The artificial diffusion term can be selectively activated for each cell depending on the presence of oscillations in the solution or its derivatives [1, 2]. Huang et. al used a simple diffusion term applied only to the internal solution points of each cell. The magnitude of the diffusion is determined by finding the maximum magnitude value and cell spacing within the cell and multiplying by the 2nd derivative in Eqn. 2.94. This definition allows the diffusion value to automatically
scale depending on the local approximation.

\[
ArtificialDiffusion = \frac{1}{2} |u_{\text{max}}| \Delta x_{\text{max}} \frac{\partial^2 u}{\partial x^2}
\] (2.94)

Extension to the Euler equations is not a directly straightforward process. The following are a series of basic approaches proposed as means to eliminate non-physical oscillations in the solution for the Euler equations using numerical dissipation. These definitions are not ideal but investigated for their simplicity and compatibility with the present scheme. Improvements are suggested in Chapter 5.

**a. Diffusion Constant**

Probably the most basic form of the artificial diffusion involves the use of a scalar constant multiplied by the second derivative of \( \vec{Q} \). This method has the advantage of easy implementation and computational efficiency. However, the scalar constant does not have the flexibility to scale with the local solution and is likely to be either over or under dissipative if not carefully chosen for the specific problem.

\[
ArtificialDiffusion = K \frac{\partial^2 \vec{Q}}{\partial x^2}
\] (2.95)

The simple scalar diffusion term is useful for determining the behavior of the high order scheme as diffusion is changed and also provides a basis to determine a general magnitude of diffusion that should be applied. This definition along with all following derivations are only applied to the internal nodes of each cell in order to prevent diffusion from crossing the cell boundary.

**b. Tensor Roe-Based Diffusion**

We will now look to the inherent dissipation in the 1st-order Roe scheme as a starting point for all following diffusion definitions [22]. The basic scheme applied to a cell
drawn around an internal node is given in Eqn. 2.96. The right hand side of this equation is split into two parts (Eqn. 2.97) with the left portion becoming the inviscid flux gradient and the remaining portion identified as the diffusive contribution (Eqn. 2.97).

$$\begin{align*}
\vec{F}_{i+1/2} - \vec{F}_{i-1/2} &= \frac{1}{2\Delta x} \left[ \vec{F}_{i+1} - \vec{F}_{i-1} - \left( \left[ \hat{A}_{i+1/2} \right] \left( \vec{Q}_{i+1} - \vec{Q}_i \right) - \left[ \hat{A}_{i-1/2} \right] \left( \vec{Q}_i - \vec{Q}_{i-1} \right) \right) \right] \\
&= \frac{\partial \vec{F}}{\partial x} - \frac{1}{2\Delta x} \left( \left[ \hat{A}_{i+1/2} \right] \left( \vec{Q}_{i+1} - \vec{Q}_i \right) - \left[ \hat{A}_{i-1/2} \right] \left( \vec{Q}_i - \vec{Q}_{i-1} \right) \right) \quad (2.96)
\end{align*}$$

Because we are looking for an approximate and efficient method for determining the artificial diffusion, we will now take a few liberties in the derivation in order to further simplify the equation. Multiplying \((\vec{Q}_{i+1} - \vec{Q}_i)\) by \(\Delta x_R/\Delta x_R\) and \((\vec{Q}_i - \vec{Q}_{i-1})\) by \(\Delta x_L/\Delta x_L\) which both become first derivatives across each interface reduces 2.97

---

Figure 2.6: A subdomain is broken into smaller internal cells in order to apply the Roe scheme within the subdomain.
\[
\frac{\partial \vec{F}}{\partial x} - \frac{1}{2\Delta x} \left( \Delta x_R[|\vec{A}_{i+1/2}|] \frac{\partial \vec{Q}}{\partial x} |_R - \Delta x_L[|\vec{A}_{i-1/2}|] \frac{\partial \vec{Q}}{\partial x} |_L \right) \quad (2.98)
\]

Next, the Jacobian matrix is assumed at the cell center and the \( \Delta x_R \) and \( \Delta x_L \) values are assumed to be the maximum within the subdomain allowing Eqn. 2.98 to reduce to 2.100. Dividing the first derivatives by \( \Delta x \) produces the second derivative of \( \vec{Q} \).

\[
\frac{\partial \vec{F}}{\partial x} - \frac{1}{2\Delta x} \Delta x_{max}[|\vec{A}|] \left( \frac{\partial \vec{Q}}{\partial x} |_R - \frac{\partial \vec{Q}}{\partial x} |_L \right) \quad (2.99)
\]

Finally, the maximum of each Eigenvalue is used to define the flux Jacobian producing a tensor diffusion constant for the entire subdomain which scales to the local flow conditions. The advantage of this definition lies in the presence of a true second derivative which can be easily matched to the LHS discretization of the delta formulation. This diffusion term is scaled for each conservation law and is grid dependent.

\[
K = \frac{1}{2} \Delta x_{max}[R] \begin{bmatrix}
max(|\hat{\lambda}_1|) & 0 & 0 \\
0 & \max(|\hat{\lambda}_2|) & 0 \\
0 & 0 & \max(|\hat{\lambda}_3|)
\end{bmatrix} [R^{-1}] \quad (2.101)
\]

c. Scalar Roe-Based Diffusion

Returning to Eqn. 2.100, more simplifications can be made to generate a third definition for the artificial diffusion term. Replacing the maximum of each Eigenvalue with the maximum of all Eigenvalues within the cell reduces the \( 3 \times 3 \) matrix term.
to a scalar constant (Eqn. 2.102).

\[
K = \frac{1}{2} [R] \begin{bmatrix}
|\hat{\Lambda}|_{\text{max}} & 0 & 0 \\
0 & |\hat{\Lambda}|_{\text{max}} & 0 \\
0 & 0 & |\hat{\Lambda}|_{\text{max}}
\end{bmatrix} [R^{-1}] = \frac{1}{2} \Delta x_{\text{max}} |\lambda|_{\text{max}} [R][I][R^{-1}] = \frac{1}{2} \Delta x_{\text{max}} |\lambda|_{\text{max}}
\]

(2.102)

The value introduced in Eqn. 2.102 provides another theoretical basis serving as a guideline for an artificial viscosity value. This value also scales the diffusion strength to the local flow conditions and ensures the diffusion is locally at least as dissipative as the 1st-order Roe scheme. The advantage of this derivation lies in the scalar nature of the diffusion coefficient which requires only three multiplications as opposed to the nine required by its tensor counterpart.

### 2.10.1 Applying Artificial Diffusion

Incorporating the diffusion term into both the LHS and RHS, Eqn. 2.35 becomes Eqn. 2.103 where 2nd-order central differencing was applied on numerical dissipation terms.

\[
\frac{\Delta \vec{Q}}{\Delta t} + [A] \frac{\partial \Delta \vec{Q}}{\partial x} - K \frac{\partial^2 \Delta \vec{Q}}{\partial x^2} = -\vec{Q}^* - \vec{Q}^n - \frac{\partial \vec{F}^*}{\partial x} + K \frac{\partial^2 \vec{Q}^*}{\partial x^2}
\]

(2.103)

Eqn. 2.103 may be again broken down in the form of Eqn. 2.37 where the new coefficients are determined in Eqns. 2.104-2.107.

\[
[a_W] = -\frac{1}{2\Delta x_L} [R] ([\Lambda_P] + [|\Lambda_P|]) [R^{-1}] - \frac{K}{\Delta x_L \Delta x_i} [I]
\]

(2.104)

\[
[a_E] = \frac{1}{2\Delta x_R} [R] ([\Lambda_P] - [|\Lambda_P|]) [R^{-1}] - \frac{K}{\Delta x_R \Delta x_i} [I]
\]

(2.105)
\[ [a_P] = \frac{[I]}{\Delta t} - ([a_W] + [a_E]) \quad (2.106) \]

\[ \vec{S} = - \left( \frac{\vec{Q}^* - \vec{Q}^n}{\Delta t} \right) - \frac{\partial \vec{F}^*}{\partial x} + K \frac{\partial^2 \vec{Q}^*}{\partial x^2} \quad (2.107) \]

Second-order time accuracy may also be obtained following the same steps shown in Eqns. 2.42-2.45.

The artificial diffusion term is calculated independently for each cell and applied only to the internal nodes in order to prevent diffusion from crossing the cell boundary. This term modifies the solution by smoothing the reconstruction between the subdomain end points. Very large diffusion constants approaching infinity drive the reconstruction of a scalar function to a linear fit reducing the subdomain solution to 1st-order. However, a lower diffusion constant allows for a smooth reconstruction that suppresses Gibbs oscillations. Figure 2.7 demonstrates the difference between large \( K \) values and well chosen \( K \) values. The behavior of the reconstruction depends on the choice for \( K \).

![Artificial diffusion behavior for very large K values (left) and smaller K values (right) and comparison (center).](image)

The derivations provided above are guidelines for sufficient, yet not completely dominant diffusion constant values. These are theoretical guidelines meant to ensure the diffusion is at least sufficiently dissipative, but may not be the optimum values. It
will be shown later in Chapter 4 that these guidelines are generally overly dissipative and can be adjusted to generate better solutions.

### 2.10.2 Selective Diffusion Activation

A major advantage of the current subdomain spectral method is the independent solution procedure for each cell. This allows diffusion to be activated/deactivated independently in each subdomain for every time step ensuring the solution is smoothed only in the needed regions and that accuracy is not lost in other locations within the domain. The method for triggering the diffusion term can vary. In the present work, we will analyze the local cell solution to determine the presence of oscillatory behavior and propose several methods to trigger diffusion.

#### a. Q Differential Switch

One method for detecting Gibbs oscillations is used by Huang (2005) for scalar hyperbolic equations and adapted for the Euler equations in the present study. This method monitors the change in the gradient between adjacent solution points in the cell. If the gradient is rapidly changing anywhere within the cell, the diffusion term is made active for the current time step.

\[
\frac{|(\frac{\partial Q}{\partial x})_e - (\frac{\partial Q}{\partial x})_w|}{\min \left[ |(\frac{\partial Q}{\partial x})_e|, |(\frac{\partial Q}{\partial x})_w| \right]} > \epsilon
\]  

(2.108)

This criteria is generally only met at discontinuities and when spurious oscillations begin to form as long as \(\epsilon\) is not chosen to be overly sensitive. The parameter \(\epsilon\) determines the sensitivity of the switch and can be tuned to be more or less aggressive. A value of \(\epsilon = 20\) is generally used. Each conservative variable is independently tested against the tolerance and diffusion is activated if any one of them fails to meet the criteria. Furthermore, the switch can become overly sensitive for small gradients in
the denominator and a lower threshold was established to prevent false activation of
the diffusion term. Namely, $\min \left[ \left( \frac{\partial Q_i}{\partial x} \right)_e, \left( \frac{\partial Q_i}{\partial x} \right)_w \right] < 1e - 6$.

**b. Q Integral Switch**

A second method for detecting Gibbs oscillations is applied in Huang (2006) for
scalar hyperbolic equations and again adapted for the Euler equations in the present
study. Here, the formation of non-physical oscillations in a cell solution is detected
by comparing the area under the derivative of the conservative variables between two
polynomial approximations as shown in Eqs. 2.109 & 2.110. The parameters $k_c$ and
$n$ refer to the polynomial degree and derivative order respectively and are determined
empirically. The value for $k_c$ is generally chosen to be 100 and $n = 2$. The parameter
$\epsilon$ is set as 0.003. The sensitivity for the algorithm can be increased by incrementing
the value of $n$ because higher order derivatives are more sensitive to oscillation. As
in the refinement switch, the 2nd derivative was found to be appropriately sensitive
and preferred in order to save computation time.

$$\left| 1 - \frac{D^n_{k_c}}{D_{2k_c}^n} \right| < \epsilon \quad (2.109)$$

$$D^n_{k_c} = \sum_{i=1}^{3} \int_{0}^{\Delta x} \left| \frac{\partial^n Q_i}{\partial x^n} \right| dx \quad (2.110)$$

The above process is similar in nature to the adaptive polynomial process discussed
previously. However, there are a few important distinctions between the current and
former processes. First, the integral comparison is between two significantly different
polynomial orders and the derivative order is allowed to vary. Second, the diffusion
term is only applied once at the beginning of each time step. The result of this search
simply activates/deactivates the diffusion terms provided in Eqn. 2.103.
c. Pressure Differential Switch

The pressure quantity is generally very sensitive to oscillations at discontinuities and is affected by each conservation law as it is directly linked to other thermodynamic properties by the equation of state. It makes sense to only track a single variable as opposed to making the same calculations multiple times. Here, the Q differential switch is modified using pressure as the only criteria. In this case, $\epsilon = 5$ is generally used.

$$\frac{|(\frac{\partial P}{\partial x})_e - (\frac{\partial P}{\partial x})_w|}{\min \left[ \left| (\frac{\partial P}{\partial x})_e \right|, \left| (\frac{\partial P}{\partial x})_w \right| \right]} > \epsilon \quad (2.111)$$

\[
D_{k_e}^{n} = \sum_{i=1}^{3} \int_{0}^{\Delta x} \left| \frac{\partial^n P}{\partial x^n} \right| dx \quad (2.112)
\]

d. Pressure Integral Switch

Using the same arguments as above, the Q integral switch is also modified to track pressure fluctuations. In this case, $\epsilon = 0.001$ is used.

Of the four diffusion switches described above, it was found that the pressure differential switch most reliability applies diffusion in the appropriate locations for all test cases in the most efficient manner.

2.10.3 Low-Pass Spatial Filtering

In addition to diffusion, stability can be fostered by filtering the solution. Low-pass spatial filters have been used in a number of schemes, including compact difference schemes to remove the high-frequency modes causing instability [35]. High-order low-pass spatial filters have been derived in [35] and one-sided high-order filters have been derived in [7] for use near boundaries.

The formulation for an interior point filter is shown in Eqn. 2.113 where $\alpha_f$ is
a free, user-chosen parameter ($0 \leq |\alpha_f| < 0.5$) that controls the degree of filtering applied to the solution where lower values correspond to a more discriminating filter. Values of $\alpha_f = 0$ and $\alpha_f = 0.5$ correspond to a strong explicit filter and a deactivated filter respectively. The coefficients for Eqn. 2.13 are provided in Table 2.1 for up to a 6th-order filter. Coefficients for higher-order filters are provided in [35].

$$\alpha_f \phi_{i-1} + \phi_i + \alpha_f \phi_{i+1} = \sum_{n=0}^{N} \frac{a_n}{2} (\phi_{i+n} + \phi_{i-n})$$ (2.113)

<table>
<thead>
<tr>
<th>Scheme</th>
<th>$a_0$</th>
<th>$a_1$</th>
<th>$a_2$</th>
<th>$a_3$</th>
<th>Order of accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>F2</td>
<td>$\frac{1+2\alpha_f}{2}$</td>
<td>$\frac{1+2\alpha_f}{2}$</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>F4</td>
<td>$\frac{5+6\alpha_f}{8}$</td>
<td>$\frac{1+2\alpha_f}{2}$</td>
<td>$\frac{-1+2\alpha_f}{8}$</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>F6</td>
<td>$\frac{11+10\alpha_f}{16}$</td>
<td>$\frac{15+34\alpha_f}{32}$</td>
<td>$\frac{-3+6\alpha_f}{16}$</td>
<td>$\frac{1-2\alpha_f}{32}$</td>
<td>6</td>
</tr>
</tbody>
</table>

Increasing the filter order increases the stencil size required by the filter which can become a problem at boundaries. For this reason, high-order, one-sided boundary filters were developed in order to apply high-order filters near boundary points [7]. The coefficients for 4th and 6th-order boundary filters are provided in Tables 2.2 & 2.3. Higher-order boundary filters may be obtained from Gaitonde et al [7].

$$\alpha_f \phi_{i-1} + \phi_i + \alpha_f \phi_{i+1} = \sum_{n=1}^{N} a_{n,i} \phi_n$$ (2.114)
Table 2.2: Filter coefficients for one-sided filter at point 2

<table>
<thead>
<tr>
<th>Order of accuracy</th>
<th>$a_{1,2}$</th>
<th>$a_{2,2}$</th>
<th>$a_{3,2}$</th>
<th>$a_{4,2}$</th>
<th>$a_{5,2}$</th>
<th>$a_{6,2}$</th>
<th>$a_{7,2}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>$\frac{1+14\alpha_f}{16}$</td>
<td>$\frac{3+2\alpha_f}{4}$</td>
<td>$\frac{3+2\alpha_f}{8}$</td>
<td>$-\frac{1+2\alpha_f}{4}$</td>
<td>$\frac{1-2\alpha_f}{16}$</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>$\frac{1+62\alpha_f}{64}$</td>
<td>$\frac{29+6\alpha_f}{32}$</td>
<td>$\frac{15+34\alpha_f}{32}$</td>
<td>$-\frac{5+10\alpha_f}{16}$</td>
<td>$\frac{15-30\alpha_f}{64}$</td>
<td>$-\frac{3+6\alpha_f}{32}$</td>
<td>$\frac{1-2\alpha_f}{64}$</td>
</tr>
</tbody>
</table>

Table 2.3: Filter coefficients for one-sided filter at point 3

<table>
<thead>
<tr>
<th>Order of accuracy</th>
<th>$a_{1,3}$</th>
<th>$a_{2,3}$</th>
<th>$a_{3,3}$</th>
<th>$a_{4,3}$</th>
<th>$a_{5,3}$</th>
<th>$a_{6,3}$</th>
<th>$a_{7,3}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>$\frac{-1+2\alpha_f}{64}$</td>
<td>$\frac{3+26\alpha_f}{32}$</td>
<td>$\frac{49+15\alpha_f}{64}$</td>
<td>$\frac{5+6\alpha_f}{16}$</td>
<td>$\frac{-15+30\alpha_f}{64}$</td>
<td>$\frac{3-6\alpha_f}{32}$</td>
<td>$\frac{-1+2\alpha_f}{64}$</td>
</tr>
</tbody>
</table>

Filtering can be applied to any of the primitive or conservative variables during the solution process. The present study applies spatial filtering to the conservative variables after each iteration. While the low-pass filter can be applied after the completion of each time step, instabilities can arise within the iterative solution for each time step and, left unsuppressed by a filter or artificial dissipation, amplify after enough iterations. The application of the filter after each iteration is synonymous with applying the diffusion term for each iteration during a time step.

The current scheme employs the method locally to each cell, filtering only the internal nodes similar to the application of the artificial diffusion term. In order to preserve solution accuracy as much as possible, the highest-order filters are used within each cell which is limited to the local polynomial approximation. Ideally, the filter applied would be at least one order of accuracy higher than the local solution scheme. Figure 2.8 demonstrates the application of a low-pass filter to a 4th degree polynomial stencil. Here, the center point uses a 4th-order internal filter while the
two adjacent points can use either the one-sided 4th-order filter or the internal 2nd-order filter. The 2nd-order filter was found to produce more stable solutions than the 4th-order boundary filter for this case.

It is important to note that the application of the low-pass spatial filter is limited to the 4th-degree polynomial approximation for practical reasons. First, the limited number of points in each cell prevents the application of higher-order filters and second, very high-order filter coefficients (greater than 10th-order) are not readily available or well-known. Furthermore, application of a low-pass filter is limited to low-order solutions for the present scheme due to the impracticalities associated with obtaining filters beyond what is present in the literature.

2.11 Staggered-Grid Formulation

Other formulations of the spectral-difference scheme have preferred a staggered grid formulation which includes both flux points and solution points in each subdomain [13, 9]. The solution points most commonly utilize the Chebyshev-Gauss points while the flux points are the Legendre-Gauss points plus the two end points or the Lobatto-Gauss points (shown in Figure 2.9).

Here, the solution points form a $k - 1$ degree polynomial on $k$ solution points while the flux points form a $k$ degree polynomial on $k + 1$ points. Inviscid flux gradient calculations are performed by first interpolating the known values of the
conservative quantities at the solution points to the flux points, calculating the flux values at the internal and interface points and then determining the flux derivative at the solution points using Eqn. 2.53 where the interpolation matrix is now a $k \times k + 1$ system. This formulation allows for the flux values to be directly conformed at the cell interfaces and the order of accuracy lost from the polynomial approximation is offset by interpolating the derivative values from the flux points to the solution points.

The disadvantage of this formulation comes from the computational cost of the repeated calculations required by the staggered grid system. Consider the 3rd-order stencil shown in Figure 2.9. The interpolation from the solution points to the flux points requires four calculations. The differentiation process requires another three
calculations for a total of 7 calculations for each subdomain. 3rd-order accuracy on
the non-staggered grid would require only the 4 calculations for the differentiation
on a four-point stencil resulting in three fewer calculations for each subdomain. As
the polynomial approximation increases, the number calculations required by the
staggered-grid scheme is roughly doubled and the losses are higher if the viscous terms
are included. The current work focuses on the use of very high-degree polynomial
approximations and therefore utilizes the non-staggered mesh formulation to save
computation time.

2.12 Scheme Overview and Application

2.12.1 General Solution Process

1. Initial conditions are defined for the highest order polynomial approximation on
the respective Gauss-Lobatto points.

2. Polynomial refinement is applied to the initial conditions before time-marching in
order to reduce to the lowest order of accuracy needed to represent the function.

3. Begin the time marching procedure

   a. At the beginning of each step, run each cell through the diffusion switch.

   b. Begin solution iterations

      i. Calculate the flux quantities at the internal Lobatto points and global
         boundaries directly.

      ii. Determine the flux quantities at the subdomain interfaces using one of the
          three solvers presented in Section 2.7.

      iii. Calculate the inviscid flux gradient using Eqn. 2.51.

      iv. Find the inviscid coefficients from Section 2.2.
v. For active diffusion:
   
   - Calculate the conservative variable gradient using the average values at
     the subdomain endpoints and Eqn. 2.51.
   
   - Calculate $\nabla^2 \vec{Q}$ using the average values at the subdomain endpoints
     and Eqn. 2.54.
   
   - Append the viscous contribution to each of the coefficients
   
vi. Solve for $\Delta \vec{Q}$ in each cell and iterate until $\Delta \vec{Q}$ falls below a predefined
    tolerance.

c. Run the new solution through the polynomial refinement procedure and repeat
   step b if refinement is required, otherwise continue to step d.

d. Update $\vec{Q}^{n-1}$ with the solution obtained above

e. Repeat b-d for the next time step

2.12.2 Diffusion Switch

1. Evaluate each subdomain using the desired diffusion switch at the beginning of
   each time step.

2. If the switch is activated, calculate the diffusion term using the desired diffusion
   definition from Section 2.9 and activate the diffusion terms in each coefficient.

3. For each time-step reevaluate the need for diffusion and recalculate the diffusion
   terms.

2.12.3 Polynomial Refinement

1. Calculate gradient of the desired quantity on the Lobatto points using Eqn. 2.51
   for the current polynomial order.

2. Repeat calculation in 1 for the desired nth order derivative.
3. Evaluate the integral in Eqns. 2.89 & 2.93 using Gauss-Quadrature rules.

4. Interpolate quantities from original refinement level to the next polynomial approximation and repeat steps 1-3.

5. Compare the values from Eqns. 2.89 & 2.93 using Eqn. 2.88.

6. Repeat Steps 1-5 if the comparison tolerance is not met, otherwise continue to step 7. Note: higher polynomial approximations are always interpolated from the current approximation.

7. Once the new approximation is established, interpolate $\vec{Q}^{n-1}$ from the reference values of $\vec{Q}_r^{n-1}$ obtained from the final values of the previous time step and approximation. Every new approximation is interpolated from this reference solution.

8. Repeat the solution procedure for the new approximation and repeat steps 1-6. If no change in polynomial approximation occurs in step 6, skip to step 9. Otherwise repeat 1-7 a maximum of 5 times.

9. The final solution has been obtained and the next time step is prepared. Redefine $\vec{Q}_r^{n-1}$ using $\vec{Q}^n$ and repeat steps 1-8 for the next time step.
Chapter 3

Model Problems

3.1 Shock-Tube Problem

A shock-tube problem was used to demonstrate the performance of the Gauss-Lobatto quadrature using a variety of parameters [36]. The problem consists of two chambers initially separated at \( t = 0(\text{s}) \) where the initial conditions are given by,

\[
\begin{align*}
\text{for } x \leq 0 & \quad \begin{cases} 
\rho = 1.0 \\
P = 1.0 \\
u = 0.0 
\end{cases} \\
\text{for } x > 0 & \quad \begin{cases} 
\rho = 0.125 \\
P = 0.1 \\
u = 0 
\end{cases}
\end{align*}
\]  \quad (3.1)

The membrane is then removed and an expansion wave propagates to the left while a contact surface and shock wave propagate to the right. An example of a shock-tube is shown in Figure 3.1. The shock-tube problem is an interesting test case because it has a readily available analytical solution, exhibits several common gas dynamics phenomena, and is a common benchmark for compressible flow solvers [37]. Additionally, the presence of two discontinuities allows one to test the shock capturing capabilities of a new scheme.

The analytical solution to the shock-tube problem is provided in Figure 3.1. Three
phenomena are visible in the density plot. The expansion wave is located in region 1 which is a continuous left-running wave and affects the velocity and pressure as well. The contact discontinuity is located in region 2. The contact discontinuity is a discontinuity seen only in the density and moves with the fluid to the right. Finally, the shock wave is shown in region 3. The shock wave is a right-running discontinuous jump in the pressure that induces a velocity in the flow and a change in density.

Figure 3.2: Analytical shock-tube solution at $t = 0.2$ from initial conditions given in Eqn. 3.1. The density plot (a) shows an expansion wave, contact discontinuity and shock wave in regions 1, 2, and 3 respectively. The plot of velocity is shown in (b) and the pressure is shown in (c)
3.2 Shu-Osher Problem

A shock-entropy wave interaction contains both complex smooth regions and discontinuities allowing a broader test of the method’s performance [18, 38]. Initially, the fluid has a constant pressure, density and velocity on the left and a shock wave propagates into a sinusoidal density profile inducing a number of fine-scale structures behind the main shock. The entropy waves are sensitive to numerical dissipation which provides a good test for the methods presented in this study. The initial conditions are given by,

\[
\begin{align*}
\text{for } x \leq -4 & \quad \begin{cases} 
\rho = 3.857143 \\
P = 10.333333 \\
u = 2.629369
\end{cases} \\
\text{for } x > 0 & \quad \begin{cases} 
\rho = 1 + 0.2 \sin(\pi x) \\
P = 1 \\
u = 0
\end{cases}
\end{align*}
\]

(3.2)

The reference solution is shown in Figure 3.3 where (a), (b), and (c) are again the density, velocity and pressures respectively. This reference was obtained from the author of [18] where a second-order MUSCL scheme with 3,200 cells was used to obtain a hyper-refined solution.

![Reference Shu-Osher solution at t = 1.8 seconds.](image)

Figure 3.3: Reference Shu-Osher solution at t = 1.8 seconds.
Chapter 4

Results

The methodology described in Chapter 2 has been employed in a new code as a 1D solver written in the FORTRAN 90 language [39]. A brief discussion of the algorithm’s makeup and capability are in order to begin this section.

The program was decomposed into a large number of modules and subroutines while the main program follows a basic outline of the solution process. The highly modular design was implemented in order to improve program quality and produce a system that is easily extendible, modified, and debugged. For example, the current version includes subroutines to solve the implicit system of equations using either a block tri-diagonal [31] or a block Gauss-Seidel [32] algorithm. The solution procedures are easily swapped by changing a single parameter. Additional solution procedures may be added by inserting them as subroutines into the solution module. This concept provides similar benefits to many other aspects of the solution procedure where various methods were included in the code and easily switched between. These methods were used to make particular calculations and the performance of each method was compared. A couple examples of this such are switching between the Roe scheme and upwind characteristics definitions for interface fluxes and testing different diffusion definitions.
In order to avoid recompiling the program for each test case, a number of variables were dedicated as namelist inputs. These variables ranged from the polynomial order and number of cells to physical constants and switches between the code’s different capabilities. The inclusion of a namelist allowed for a method to quickly and easily change operating parameters such as the equation solver and boundary conditions and run a new solution using the existing executable file.

A number of data inputs were utilized during the solution process. These included the polynomial interpolation weights described in Eqns. 2.53 & 2.55 which are computed for a general interval of $[-1, 1]$ and transferred to the actual solution interval using the coordinate transformation in Eqn. 2.57. Weights for the integrals and derivatives were also computed in advance and included with the file input. Precomputing these weighting values greatly reduced computation time. A copy of the full program is available from the author upon request.

The rest of this section shows results obtained using the methodology described in Chapter 2 using the code described above on the benchmark cases discussed in Chapter 3. Several aspects of the methodology were tested and demonstrated in the following sections. Recommendations for future work will be discussed in Chapter 5.

### 4.1 Initial Results

The effects of spatial resolution alone were investigated in this article. In order to decouple the effects of time accuracy from that of the spatial resolution, very small time steps were used such that the temporal resolution has a negligible effect on the overall solution accuracy.

Converged solutions are shown in Figure 4.1 to demonstrate the scheme is capable of producing exact solutions given appropriate polynomial/mesh refinement. The Shu-Osher solution was generated using 800 cells and 4th degree polynomials using
the spatial low-pass filter discussed in Section 2.10.3. Adaptive polynomial refinement was not used in this case, but the diffusion switch was used to activate the filter in the appropriate cells each time step. The shock-tube solution also generated using 100 cells, 29th degree polynomials, and the scalar diffusion term.

![Figure 4.1: Converged solutions for (a) the shock-tube problem using 20 cells, 29th degree polynomials and artificial diffusion (b) the Shu-Osher problem using 800 cells, 4th degree polynomials and the low-pass filter.](image)

These refined solutions allow for close agreement in the fine-scale density oscillations of the shock-entropy wave interaction and high-order shock capturing in the shock-tube. The author now steps back to demonstrate and verify selected aspects of the new scheme.

### 4.2 1st-Order Solutions

1st-order solutions were obtained in order to validate the performance of the scheme at the lowest order of accuracy. The upwind characteristic method was compared to the Roe scheme for a basic 1st-order code and then applied to the 1st-order solution using the methods discussed in this work.
4.2.1 Upwind Characteristics vs. Roe Scheme

1st-order solutions were obtained using a traditional 1st order scheme in which the flux values are determined using either the Roe scheme or the upwind characteristic method described in Section 2.7. Both the shock-tube and shock-entropy wave problem were compared using 400 and 1,000 cells in Figure 4.2.

![Figure 4.2](image-url)

(a) Shock-Tube Problem (n = 400)  
(b) Shu-Osher Problem (n = 400)  
(c) Shock-Tube Problem (n = 1,000)  
(d) Shu-Osher Problem (n = 1,000)

Figure 4.2: Roe scheme solution compared to upwind characteristics for the shock tube problem (left) and the shock-entropy wave problem (right) for 400 cells (top) and 1,000 cells (bottom). Every other point is plotted for n = 400 and every four points are plotted for n = 1,000.
The two methods produce identical solutions for both test cases and levels of refinement, leading to the conclusion that choosing the upwind characteristic values is a sufficient method for defining the flux at the subdomain boundaries.

4.2.2 1st-Order Polynomial Solutions

Reducing the number of solution points per cell to the two Lobatto points located on the cell boundaries reduces the polynomial approximation to a linear function. A piecewise-constant approximation results in a 1st-order solution which is identical to standard 1st-order solutions produced by the Roe scheme. Figure 4.3 shows the 1st-order solution using the Roe scheme is equivalent to the linear polynomial approximation for the current method using the same number of cells. As expected, the two solutions agree and the lower limit for the solution procedure is 1st order. Because of extra numerical overhead and overlap of nodes at cell boundaries, the current method is not best suited for low order solutions and all future results will use at least 4th degree polynomials or higher.
Figure 4.3: 1st-order Roe scheme solution compared to 1st-degree polynomial solution using 400 cells (top) and 1000 cells (bottom) for the shock-tube problem (left) and shock-entropy wave problem (right).

4.3 Artificial Diffusion

For high-order approximations of problems with discontinuities or steep gradients, a diffusion term is necessary in order to produce a solution. Growing, spurious oscillations can result in numerical errors that terminate the solution procedure. Because the test cases demonstrated in this paper all have discontinuities or steep gradients,
some form of numerical dissipation is applied to each problem when high-order approximation is used. This non-physical viscosity term needs to be treated carefully in order minimize its effects on the final values. The goal is to determine the best way to apply dissipation to eliminate Gibbs oscillations, while minimizing diffusion.

The methodology for the diffusion term was shown in Section 2.9. The current section presents and discusses the different aspects of the diffusion term’s performance. Behavior of the diffusion term for different polynomial approximations are discussed and several methods for applying the term are demonstrated and compared.

4.3.1 High-Order Diffusion Approximation

The 2nd derivative in Eqn. 2.103 can be defined using a 2nd-order central difference or using the higher-order approximations. The polynomial approximation for higher order derivatives was defined in Eqn. 2.54 as applying the derivative weights of Eqn. 2.53 in a sequential manner. Therefore, a higher-order approximation of the second derivative may be obtained in this manner with existing calculations and applied in the diffusion term. We will briefly discuss and demonstrate the difference in these two approximations here.

Figure 4.4 shows the shock solution for the shock-tube problem using both computations for the diffusion term at uniform, constantly applied, diffusion values small enough to allow some oscillation in final solution. The blue line is the result of the second-order approximation while the red line is the result of the high-order approximation. It is apparent from both of the diffusion constants tested that the high-order definition is slightly more proficient at eliminating the Gibbs’s phenomenon.
Figure 4.4: Comparison between 2nd order central-difference definition for the diffusion term and the polynomial approximation for the 2nd order derivative performance in eliminating Gibbs oscillations. Shock-tube solution at $t = 0.2$ seconds, view limited to the shock. Solutions were obtained using 20 cells and 19th degree polynomials.

Because the polynomial approximation improves with increases in the polynomial degree, the diffusion definition may be expected to improve with increasing polynomial order, while the converse is true with decreasing polynomial order. However, the matrix multiplication involved in calculating the higher-order 2nd derivative becomes more expensive as the polynomial degree is increased. The high-order approximation for the diffusion term does not appear to have a clear advantage over the standard 2nd-order definition which behaves the same regardless of the local polynomial approximation. Therefore, the second-order definition was used for all following calculations.

4.3.2 Constant Scalar Diffusion

Figure 4.5 demonstrates how the shock solution changes as a scalar constant diffusion coefficient is applied throughout the domain and varied. At low diffusion values of $K = 0.0002$ to $K = 0.0003$ oscillations are prevalent. However, these oscillations
decrease to small perturbations behind the shock at the mid-range values of $K = 0.0004$ and $K = 0.0005$. At $K = 0.00075$ the Gibbs phenomenon has been completely removed presenting an oscillation-free solution. Increasing the diffusion to $K = 0.001$ shows the shock resolution begins to be smeared for diffusion coefficients overly large and continues to lose accuracy as the diffusion constant is increased. We can conclude from this that there exists some optimum value for the artificial diffusion constant that eliminates oscillations and is minimally dissipative.
Figure 4.5: Increasing the diffusion constant from \( K = 0.0002 \) to \( K = 0.001 \) reduces and eliminates spurious oscillations. Shock-tube solution at \( t = 0.2 \) seconds, view limited to the shock for demonstration. Solutions were obtained using 20 cells and 19th degree polynomials.
4.3.3 P-Value Adjustment

A basis for setting the level of artificial diffusion was presented previously. However, it is important to revisit and investigate potential improvements to this concept since the derivations of the these terms are non-physical guidelines and likely overly diffuse. Because the diffusion as derived in Section 2.9 is at a minimum larger than it needs to be, an attempt to improve the solution agreement may be made by scaling the diffusion coefficient using Eqns. 4.1 & 4.2 where \(0 < p \leq 0.5\).

Tensor Roe-Based Diffusion

\[
K \approx p[R] \begin{bmatrix}
\max(|\hat{\lambda}_1|) & 0 & 0 \\
0 & \max(|\hat{\lambda}_2|) & 0 \\
0 & 0 & \max(|\hat{\lambda}_3|)
\end{bmatrix} [R^{-1}]\Delta x_{\text{max}}
\]

Scalar Roe-Based Diffusion

\[
K \approx p|\lambda|_{\text{max}}\Delta x_{\text{max}}
\]

The benefits gained by scaling the diffusion term may be observed in Figure 4.6 where polynomial degrees of 9, 19, and 29 are shown using the simplified scalar diffusion term. Scaling the diffusion term greatly improves the solution accuracy, especially as the polynomial degree is increased. The 9th degree polynomial solution becomes oscillatory for small decreases in the diffusion term. However, the agreement for the 19th degree solution is significantly improved without the appearance of oscillations as \(p\) is reduced to \(p = 0.05\). In order to demonstrate the effect of scaling the diffusion term, the 99th degree polynomial solution using \(p = 0.5\) is compared to the 19th degree solution using \(p = 0.1\). It may be observed that the scaled 19th degree solution is nearly as resolved as the unscaled 99th degree solution at a fraction of the computational cost. The effect of p-adjustment on the 99th degree solution is shown in Figure 4.8 to allow the very high-order approximation to fit the shock
almost exactly.

Figure 4.6: Solutions at the shock wave for (a) 9th, (b) 19th, and (c) 29th degree polynomial approximations using several p-values. Solutions obtained using 20 cells and simplified scalar diffusion with the diffusion switch. The view is limited to the shock.

Figure 4.7: 19th degree polynomial solution with a p-value of 0.1 compared to the 99th degree polynomial solution with a p-value of 0.5. Solution obtained using 20 cells and simplified scalar diffusion with the diffusion switch.
A drastic improvement in the shock-entropy wave problem was observed by scaling the diffusion term to be less aggressive using the simplified tensor diffusion in Figure 4.9. The solutions were produced using the nominal value of \( p = 0.5 \), the lowest value of \( p \) producing a stable solution, and an intermediate value for \( p \) that is well-conditioned to the given problem.

Little advantage was found in polynomial refinement for \( p = 0.5 \) due to the overbearing nature of the base diffusion term. However, reducing the diffusion to the minimum values demonstrates a drastic improvement in the solution agreement to the reference values. The lower \( p \)-values allow the higher degree polynomials to more accurately fit the fine-scale density fluctuations of the reference solution as the polynomial degree is increased using only 20 cells.
Figure 4.9: Shock-entropy wave solutions for (a) 9th, (b) 19th, and (c) 29th degree polynomial approximations using several p-values. Solutions were obtained using 20 cells and simplified tensor diffusion. The diffusion switch was active.

In each case, scaling the diffusion to the lowest stable value results in spurious oscillations. This is most noticeably observed in Figure 4.9c where a single oscillation is observed leading the pressure wave. Resulting oscillations in the other cases are present, but less significant.

Figure 4.10 provides a closer look at the density fluctuations for the different
polynomial approximations. The 9th degree solution shows a very poor result for 
$p = 0.5$, but the fine scale structures become more obvious as $p$ is reduced to 0.1. 
In this case, the poorly resolved polynomial/cell approximation results in physical 
oscillations that are slightly shifted. The 19th degree polynomial solution shows a 
poor solution using $p = 0.5$ but much better agreement with the density fluctuations 
for lower $p$ values. In this case, the oscillations are no longer shifted as a result of the 
improved accuracy. Finally, the 29th degree solution shows a significant improvement 
in this region over the base diffusion definition and the effects of polynomial refinement 
are more apparent.

![9th degree polynomial](image)
![19th degree polynomial](image)
![29th degree polynomial](image)

Figure 4.10: Shock-entropy wave solutions for (a) 9th, (b) 19th, and (c) 29th degree 
polynomial approximations using several $p$-values. Solutions were obtained using 20 
cells and simplified tensor diffusion. The diffusion switch was active. The view is 
limited to the fine-scale density fluctuations.

Each polynomial degree used in this work was tested at several levels of grid 
resolution for different $p$ values and safe scaling factors were determined. These are 
compiled in Table 4.1 and used as guideline values for all remaining solutions.

<table>
<thead>
<tr>
<th>Degree</th>
<th>4th</th>
<th>9th</th>
<th>19th</th>
<th>29th</th>
<th>39th</th>
<th>49th</th>
<th>99th</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p = $</td>
<td>0.5</td>
<td>0.25</td>
<td>0.13</td>
<td>0.13</td>
<td>0.12</td>
<td>0.12</td>
<td>0.1</td>
</tr>
</tbody>
</table>
4.3.4 Selective Diffusion

In previous sections, solutions were generated by consistently applying the diffusion term to the entire domain throughout the solution procedure with a few examples using solutions obtained with the diffusion switch described in Section 2.10.2. The diffusion switch will now be demonstrated. The following examples utilize the pressure diffusion switch using the p-value guidelines established in Table 4.1.

The first example in Figure 4.11 shows the diffusion switch applied to a uniform 19th degree shock-tube solution on 20 cells at different times. The diffusion term is active leading the shock wave and expansion waves in the first frame. As the solution continues, the diffusion term remains active leading the shock and is occasionally active on the shock depending on the shock’s location within a cell. The diffusion term leading the expansion wave is no longer necessary once this region has been sufficiently smoothed. The 3rd and 4th frames show the diffusion term begins to activate at the tail of the expansion wave and occasionally near the contact surface as oscillations begin to appear due to the discontinuity of density in this region.

Figure 4.12 is a similar solution using 49th degree polynomials. The first frame reveals the diffusion term to be active in the subdomains containing and leading the shock wave, but the diffusion term is not active leading the expansion wave in the first frame of this example. This is likely a result of the increased resolution eliminating the need for the diffusion term in this region. As time advances, the diffusion term becomes active around the shock wave and again in the region trailing the expansion and contact discontinuity. This trend continues as the solution advances in time.

There appears to be little difference in the behavior of the diffusion term between the two polynomial approximations with the exception that the diffusion term appears to be more aggressive at the shock wave for the higher-order case. This is to be expected as the higher-order approximation is more prone to oscillation due to the steeper slope at the shock.
Figure 4.11: Shock-tube solution using 20 cells and 19th degree polynomials. Yellow bars indicate the diffusion term is active for the current time step, m indicates the local polynomial approximation.
Figure 4.12: Shock-tube solution using 20 cells and 49th degree polynomials. Yellow bars indicate the diffusion term is active for the current time step, m indicates the local polynomial approximation.
The next examples contain the time solutions for the Shu-Osher problem using 29th and 49th degree polynomials in Figures 4.13 & 4.14 respectively. For the 29th degree solution, the diffusion term is active in a large number of cells surrounding the initial perturbation. As this shock moves into the density fluctuations the diffusion term remains largely active surrounding the shock and in the fine-scale structures trailing the shock. The diffusion term is occasionally activated in the coarser density fluctuations as numerical instabilities arise.

A more prevalent difference is observed in the shock-entropy wave problem when the diffusion switch is applied to a higher-degree polynomial solution. The results in Figure 4.14 reveal the diffusion term to be active in only a few cells at a time at any point in the solution procedure. These are generally active leading the shock and occasionally within one of the density oscillations. The better resolution provided by the higher degree polynomials allows for the solution points to fit the fine-scale oscillations in a more exact manner. The resulting solution is more well-behaved leading to less of a dependence on artificial diffusion to maintain stability.

The accuracy of Shu-Osher problem solutions depend on the behavior of the shock as it propagates downstream. Even though the diffusion term is only active at the head of the shockwave a majority of the time, its effects remain at every point the shock has passed through. This is apparent in the reduced magnitude of the fine-scale structures of the final solution where the diffusion term rarely activates yet have failed to recover to the exact analytical values.
Figure 4.13: Shock-entropy wave solution using 20 cells and 29th degree polynomials. Yellow bars indicate the diffusion term is active for the current time step, m indicates the local polynomial approximation.
Figure 4.14: Shock-entropy wave solution using 20 cells and 49th degree polynomials. Yellow bars indicate the diffusion term is active for the current time step, m indicates the local polynomial approximation.
By minimally applying the diffusion term in specific locations, the effect of diffusion was greatly reduced. This is demonstrated in Figures 4.15 & 4.16 where the solutions for the shock-tube and shock-entropy wave are presented using the diffusion switch and constantly applying the diffusion term. Figure 4.15 shows the shock-tube solution using 20 cells and 9th degree polynomials. It may be observed that using the diffusion switch greatly improved the resolution at the head of the expansion wave and had a significant effect at the contact discontinuity and shock wave. The velocity term is most affected at the shock by the presence of the viscous term. Figure 4.15(b) presents the shock and more clearly portrays the smeared solution resulting from the over-used diffusion term.

Figure 4.15: Comparison of performance between selective diffusion and constantly applied diffusion terms for (a) the shock-tube problem and (b) magnified view of the shock. Solution was obtained using 20 cells and 9th degree polynomials.

The shock-entropy wave problem shown in Figure 4.16 is also significantly affected by the lack of a selective diffusion switch. Both the fine-scale and coarser fluctuations are smoothed by the diffusion term with the greatest amount of resolution lost in the fine-scale oscillations as the peak of each oscillation is cut shorter than the equivalent solution using the diffusion switch.
Figure 4.16: Comparison of performance between selective diffusion and constantly applied diffusion terms for (a) the shock-entropy wave problem and (b) magnified view of the fine-scale density oscillations. Solutions were obtained using 20 cells and 19th degree polynomials.

4.4 Polynomial Refinement

The polynomial weights are easily computed for each level of approximation and stored before the solution procedure is run as stated previously. An infinite number of polynomial approximations and interpolations can be precomputed and used in the simulation procedure. One advantage of the current scheme is the ease at which these approximations are obtained and transferred between. Here, we will investigate the application of polynomial refinement.

4.4.1 Uniform Refinement

Figure 4.17 presents the results of polynomial refinement at the shock using 20 cells and an increasing degree polynomial approximation with a scalar diffusion term of $K = 0.0002$. The lowest order solution shown produces a very good agreement with the shock, however, large non-physical oscillations occur behind the wave while
smaller oscillations lead the wave. Increasing the polynomial degree using the same
diffusion value not only improves the shock fit, but reduces the spurious oscillations.
The 39th degree solution produces an oscillation-free solution with a very good shock-fit while the 49th degree solution improves it further.

It is apparent that polynomial refinement not only serves as a mechanism to
improve solution accuracy, but dependence on artificial dissipation reduces with in-
creasing polynomial degree. While the 49th degree solution produces an improved shock fit, an even better agreement might be produced by reducing K. This obser-
vation helps to explain the need for decreasing scaling factors presented in Table 4.1. Furthermore, it demonstrates the benefit of using diffusion definitions (b) and (c) which are proportional to $\Delta x_{max}$. As the polynomial approximation is increased $\Delta x_{max}$ decreases, reducing the diffusion applied within the subdomain.
Figure 4.17: Reduction of oscillations with polynomial refinement using a constant diffusion value of $K = 0.0002$. Shock-tube solution at $t = 0.2$ seconds, view limited to the shock for demonstration. Solutions were obtained using 20 cells.

Uniform polynomial refinement is demonstrated in Figures 4.18 & 4.19 for the test cases using a coarse mesh of 20 cells. Solutions approach the reference values for both cases as the polynomial approximation is increased. Unfortunately, convergence at the contact discontinuity appears to be limited as the polynomial approximation increases and this limitation is likely related to dissipation in conservation of mass.
4.4.2 Adaptive Polynomial Refinement

Adaptive polynomial refinement has been applied to the scheme and demonstrated in this section. Any order of accuracy can be used in this process, but it is more
efficient to use a smaller selection to reduce the number of comparisons. For the present study refinement was limited to polynomial approximations of 4th, 9th, 19th, 29th, 39th, and 49th degrees. Solutions were captured at several points in time in order to demonstrate the variation of refinement with time.

The first example shows the shock-tube solution using adaptive polynomial refinement with a maximum of 29th degree polynomials. The initial refinement based on the initial conditions reduces the solution to 4th-order everywhere except near the discontinuity where the approximation uses the highest order available in order to produce the most accurate solution. As time advances, the local approximations increase as the shock and expansion waves move into new subdomains and the procedure attempts to produce exact solutions based on the available polynomial refinement. Eventually, local solutions reduce the approximation as the head of each wave leaves the subdomain and lower-order becomes a sufficient approximation.

Figure 4.21 provides a similar solution with the maximum approximation reaching 49th degree polynomials. Again, the solution begins with the approximation at the lowest order everywhere except at the discontinuity where it is automatically elevated to the highest order in an attempt to produce more accurate solutions. However, 29th degree polynomials provide enough accuracy to satisfy the refinement criteria at the head of the expansion wave in the first frame. As time advances, more cells are again raised to the highest order of accuracy, while a few return to lower order approximations as the waves leave each cell. The fifth frame provides a good example of the solution returning to low order in the perturbed flow region.
Figure 4.20: Shock-tube solution using adaptive polynomial refinement on 20 cells. The approximation is limited to 29th degree polynomials. Yellow bars indicate the diffusion term is active for the current time step, $m$ indicates the local polynomial approximation.
Figure 4.21: Shock-tube solution using adaptive polynomial refinement on 20 cells. The approximation is limited to 49th degree polynomials. Yellow bars indicate the diffusion term is active for the current time step, m indicates the local polynomial approximation.
Figure 4.22: Comparison of performance between uniform and adaptive polynomial refinement for (a) the shock-tube problem and (b) magnified view of the shock. Solutions were obtained using 20 cells, adaptive polynomial refinement and 29th degree polynomials as the maximum approximation.

Figure 4.23: Comparison of performance between selective diffusion and constantly applied diffusion terms for (a) the shock-tube problem and (b) magnified view of the shock. Solutions were obtained using 20 cells, adaptive polynomial refinement and 49th degree polynomials as maximum refinement.
Figure 4.24: Shock-tube solution using adaptive polynomial refinement on 100 cells. The approximation is limited to 29th degree polynomials. Yellow bars indicate the diffusion term is active for the current time step, m indicates the local polynomial approximation.
Figures 4.25 & 4.26 demonstrate adaptive polynomial refinement applied to the Shu-Osher problem for upper polynomial limits of 29th and 49th degree respectively. The 29th degree solution shows the local approximation is immediately increased to the maximum near the initial discontinuity and approximations jump to the maximum as the shock passes into each subdomain. However, approximations rarely reduce in order in the perturbed region as the maximum approximation is insufficient to recover the accuracy lost due to the diffusion applied to the head of the shock wave. The approximation does decrease as the low frequency oscillations on the left leave each subdomain.
Figure 4.25: Shu-Osher solution using adaptive polynomial refinement on 20 cells. The approximation is limited to 29th degree polynomials. Yellow bars indicate the diffusion term is active for the current time step, m indicates the local polynomial approximation.
Figure 4.26: Shu-Osher solution using adaptive polynomial refinement on 20 cells. The approximation is limited to 49th degree polynomials. Yellow bars indicate the diffusion term is active for the current time step, \( m \) indicates the local polynomial approximation.
The 29th and 49th degree solutions for the Shu-Osher problem are compared in Figures 4.27 & 4.28 respectively. Both cases resulted in very similar solutions highlighting the point that adaptive polynomial refinement can be used to vary the order of accuracy within the domain without significantly affecting solution accuracy.

Figure 4.27: Comparison of performance between uniform and adaptive polynomial refinement final solutions (a) the Shu-Osher problem and (b) magnified view of the density fluctuations. Solutions were obtained using 20 cells, adaptive polynomial refinement and 29th degree polynomials as maximum refinement.
Figure 4.28: Comparison of performance between uniform and adaptive polynomial refinement final solutions (a) the shock-tube problem and (b) magnified view of the shock. Solutions were obtained using 20 cells, adaptive polynomial refinement and 49th degree polynomials as maximum refinement.

The Shu-Osher problem solution using adaptive polynomial refinement with maximum resolution of 29th degree polynomials on 100 cells is presented in Figure 4.29. As before, the local approximation increases to the maximum in an attempt to maximize solution accuracy as each subdomain is perturbed. Following the shock wave, a number of the cells in the fine-scale structures reduce to low-order. Obviously, these regions have not matched the reference solution, yet they have met the refinement criteria. Several cells reduced approximation order in this region for the 20 cell solution as well, yet the results were not significantly affected. This observation suggests higher approximations in these cells would have little impact on the local solution. The lack of change in the solution with refinement is likely a consequence of the effects smoothing due to diffusion at the head of the shock-wave. Consequently, the solution accuracy is largely dominated by the application of the diffusion term at the head of the wave and not significantly by polynomial approximations in the region following the wave.
Figure 4.29: Shu-Osher solution using adaptive polynomial refinement on 100 cells. The approximation is limited to 29th degree polynomials. Yellow bars indicate the diffusion term is active for the current time step, m indicates the local polynomial approximation.
4.5 Performance Considerations

The focus of this study was to establish the numerical scheme, not to optimize computation time and therefore, many of the algorithms developed are less than ideal. A discussion of performance considerations for the different techniques used will now be presented for further developments. Because the main program has not been thoroughly optimized, comparisons with other schemes are not included.

Adaptive refinement produced solutions 25-35% faster using 29th degree polynomials as a maximum resolution. Greater efficiency was expected, however, not realized due to the number of polynomial refinement iterations required. Typically, only a single refinement iteration is necessary until a large-scale change occurs within a subdomain where several time step might require the maximum number of refinement iterations until the procedure stabilizes where the entire time step is recalculted multiple times. The efficiency of this procedure could be greatly improved over the present state since the process is not yet mature. Additionally, a sufficient refinement iteration number has not yet been determined and reducing the maximum number of iterations could greatly reduce computation time without adversely affecting the final solution.

The difference in computation time between the scalar and tensor versions of the artificial diffusion term were insignificant compared to other factors such as the choice of $p$. Overly-diffuse $p$-values required more subiterations each time step to converge on the temporal solution. The $p$-value guidelines established in Table 4.1 not only produced oscillation-free solutions, but also corresponded to some of the quicker computations. Therefore, the choice of $K$ was determined by its performance to the model problem. The shock-tube problem was found to be better suited to the scalar definition while the tensor terms performed better with the Shu-Osher problem. The addition of the diffusion switch further decreased simulation time by limiting the number of subdomains containing the diffusion term.
The pressure refinement switch was found to be faster than the Q refinement switch, but was not aggressive enough to create solutions comparable to those produced by uniform refinement. The differential diffusion switches were significantly faster than their integral counterparts due to fewer calculations. Furthermore, the differential pressure diffusion switch performed comparably to the differential Q diffusion switch, but required even fewer calculations and was preferred in the present study.
Chapter 5

Future Work

From the selected test cases presented in the current study it is apparent the solution accuracy is most sensitive to the application of artificial diffusion. While the present work attempts to define basic numerical dissipation terms for simple application and efficient computation, other methods exist that may be better suited for the model problems.

Artificial viscosity methods first proposed by VonNeumann and Richtmyer and improved by Landshoff were later addressed by a number of authors and remain a subject of improvement to this day [40, 41]. Caramana et. al. worked to further address the issues associated with applying such definitions to staggered grid solutions [34]. Fiorina and Lele have recently applied nonlinear diffusivity terms to compact difference schemes in order to eliminate non-physical oscillations in high-order applications [42]. Here, Eqn. 2.1 can be rewritten as Eqn. 5.1 where the term on the right-hand-side is the non-linear dissipation term for each conservation law. The coefficients $\mu$ and $\chi_\rho$ are based on velocity and entropy gradients in Eqns. 5.2 & 5.3.
respectively.

\[
\frac{\partial \tilde{Q}}{\partial t} + \frac{\partial \tilde{F}}{\partial x} = \begin{bmatrix}
\frac{\partial}{\partial x} \left( \chi \rho \frac{\partial \rho}{\partial x} \right) \\
\frac{\partial}{\partial x} \left( \mu \frac{\partial u}{\partial x} \right) \\
\frac{\partial}{\partial x} \left( \mu u \frac{\partial u}{\partial x} \right)
\end{bmatrix}
\]  \hspace{1cm} (5.1)

\[
\mu = C_\mu \rho (\Delta x)^{r+1} \left| \frac{\partial^r u}{\partial x^r} \right|
\]  \hspace{1cm} (5.2)

\[
\chi_\rho = C_\rho \frac{a_0}{c_p} (\Delta x)^{r+1} \left| \frac{\partial^r s}{\partial x^r} \right|
\]  \hspace{1cm} (5.3)

The values of \( C_\mu \) and \( C_\rho \) are user-defined constants while \( r \) corresponds to the order of the derivative approximation which is generally chosen as 4. The operator \( |f| \) refers to a filter applied to the absolute value of the enclosed quantity. Fiorina et al. preferred a truncated Gaussian filter for this operation although other filters can be applied. Eqn. 5.2 serves as a mechanism to detect and damp shocks while Eqn. 5.3 detects and damps regions with contact discontinuities.

Once the best method for removing Gibbs oscillations has been determined, further extensions of the existing work should be pursued. The scalar version of the present study was extended to two-dimensions and high-order approximation was used in time [1, 2]. The current high-resolution method can similarly be applied to two and three-dimensional problems. However, extension of adaptive polynomial refinement to multi-dimensions will not be straightforward and the mortar methods of Kopriva should be considered in this case [13, 15]. Also, extension to the full compressible Navier-Stokes equations should be pursued where physical diffusion terms can be treated in a similar manner as the artificial terms.

Examples of polynomial refinement up to 99th order were shown to increase solution accuracy, but even ultra-high order approximations were not capable of producing results in perfect agreement with the reference solutions on a coarse mesh. Therefore,
some element of adaptive mesh refinement should be incorporated in order to allow the method to automatically approach exact solutions. A form of hp-refinement could easily be realized by splitting a subdomain when the local polynomial approximation reaches the upper limit.

More robust techniques for determining adaptive polynomial refinement should be considered or the current techniques should be better tuned. The methods utilized in this study were capable of detecting where a higher-degree polynomial would produce a better solution, but the switches were not capable of reliably reducing the polynomial approximation in the settled regions following shocks and other perturbations.
Chapter 6

Conclusion

An implicit, variable order scheme has been developed for the Euler equations that is capable of fitting very high order polynomial approximations across discontinuities without resorting to one-sided stencils or reducing to low order. The scheme is capable of producing high order solutions and is fairly simple compared to other state-of-the-art schemes.

Spurious oscillations resulting from high-order approximation at discontinuities were relieved through the use of an artificial dissipation which serves as a smoothing mechanism within each cell. A few diffusion terms were developed and tested in this work and the benefit of each was found to be problem dependent. This diffusion term only needs to be applied in locations where oscillations arise. Several methods for triggering diffusion were presented and a switch based on the second derivative of pressure was found to perform well for every case and significantly reduced the effect of dissipation.

Polynomial refinement was demonstrated for the test cases. Even with a very coarse mesh, increasing polynomial refinement was capable of approaching the reference values. The variable order capability of the scheme was applied through adaptive polynomial refinement where each subdomain solution was allowed to increase
its polynomial approximation based on the local solution. The adaptive polynomial refinement procedure allowed for the solution process to automatically increase accuracy in regions lacking sufficient resolution without the need for mesh refinement. Furthermore, adaptive refinement created nearly identical solutions to uniform refinement without requiring ultra-high order calculations in every subdomain greatly reducing computation time.

The primary limiting factor in the present study appears to be related to the limitations enforced by the dissipation term. This is observed through the failure of the shock-tube solutions to produce a sharp contact discontinuity fit and in the entropy waves of the Shu-Osher problem where dissipation leading the shock tends to damp the oscillations limiting the magnitude of these fine-scale structures. Because the inclusion of the artificial dissipation term fundamentally changes the local equation solved and smooths nonlinearities, the exact solution cannot be perfectly obtained with the current methods. However, approximations can be improved locally by incorporating mesh refinement and local refinement of the diffusion term to include nonlinearities in the flow structure. Future progress should focus on adapting the work of more refined artificial dissipation terms for this scheme in order to improve solution accuracy and remove the limitations imposed by the overly dissipative methods currently used. The nonlinear dissipation terms proposed by Fiorina & Lele [42] could be adapted to the current scheme to each solution point.

The methods of this paper show promise of maturing towards a robust variable-order scheme capable of solving a variety of fluid dynamics problems by incorporating polynomial and mesh refinement and some element or artificial dissipation or spatial filtering.


