A Comparison of Artificial Viscosity Sensors for the Discontinuous Galerkin Method

by

Joshua Favors

A thesis submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Master of Science

Auburn, Alabama
December 14, 2013

Keywords: Discontinuous Galerkin Method, discontinuity sensor, entropy residual, modal decay

Approved by

Andrew Shelton, Assistant Professor of Aerospace Engineering
Roy J. Hartfield, Jr., Chair, Woltosz Professor of Aerospace Engineering
Brian Thurow, Reed Associate Professor of Aerospace Engineering

Abstract

The following work focuses on the development of an explicit Runge-Kutta, discontinuous Galerkin finite element method for the nonlinear, hyperbolic Euler equations. The discontinuous Galerkin method is a control-volume formulation, such as in the finite-volume method, but achieves high-order accuracy through the use of high-order polynomial approximations to the computed solution in each discretized element.

Contrary to the Navier-Stokes equations, in which viscosity is present, the Euler equations do not contain any natural dissipation terms. Numerical methods of second-order or higher are susceptible to oscillations when discontinuities, such as shocks or large gradient features form in the solution. The lack of a natural dissipation mechanism requires the use of a stabilizing method to ensure physical solutions can be obtained.

In order to preserve real physical solutions, the addition of an artificial viscosity was applied in the elements where discontinuities or large/sharp gradients of the flow variables occurred. The addition of artificial viscosity was accomplished through the use of a sensor that detected where oscillations in the approximated solution occur due to the Gibbs-phenomenon. The present work incorporates two different artificial viscosity sensors. The first sensor is based on the work by Klockner, Warburton, and Hesthaven [1] which looks at the modal decay rate of the higher-order solution modes. The second sensor was based on the work presented by Zingan, Guermond, and Popov [2] which uses the entropy production residual to determine where artificial viscosity should be applied.

In comparing the two artificial viscosity sensors, several benchmark problems were modeled: the 2-D isentropic vortex, the 2-D Kelvin-Helmholtz shear layer instability, the 1-D Sod's shock tube, the 1-D Planar Shu-Osher problem, and the 2-D shock-vortex interaction. The results include error analysis, computational cost, robustness, and user-parameters.

Acknowledgments

I have always heard that behind every great man is a strong woman. It must stand to reason that behind every crazy man is an even stronger and more patient woman. Kelly, thank you for taking my last name and enduring what I am sure has felt like one never ending semester for the past few years. You stood beside me as I chased a dream and for that I will always appreciate what we have accomplished together. I also would like to thank my daughter, Hannah, who has become the greatest inspiration in my life. You have brought a joy into our lives that we have never experienced before.

I also owe my deepest thanks to my parents who always supported me throughout my life. Without your help and confidence in me, the road I have traveled would have been impassible. Every accomplishment I have made in life has been due to the qualities instilled through you both. I also must thank my extended family for the life lessons and motivational support you have provided throughout the years.

The idea of graduate school was always a distant hope, but I must thank you Dr. Shelton for making it a reality. I appreciate you showing faith in my studies and approaching me about the possibility of graduate school as I was completing my undergraduate degree. It has been an honor to study and learn under your guidance.

timshel

Table of Contents

List of Figures

List of Tables

List of Abbreviations

b_j^k	basis function
CFD	Computational Fluid Dynamics
CFL	Courant-Fredrichs-Lewy
FD	Finite Difference
FOU	First- Order Upwind
FV	Finite Volume
$\hat{i}$, $\hat{j}$, $\hat{k}$	Cartesian coordinate directions
LGL	Legendre-Gauss-Lobatto
$\mathcal{M}^k$	transformation map
N	approximating polynomial order
NI, NJ, NK	number of elements that span each spatial direction $\hat{i}$, $\hat{j}$, and $\hat{k}$
$\mathbb{P}$	function space
$\tilde{u}_m$	modal coefficient
$\hat{u}_n$	nodal coefficient
RK	Runge-Kutta
s	entropy/smoothness coefficient
SOU	Second-Order Upwind

TVD	total variation diminishing
V	Vandermonde matrix
Ω	physical domain
Ω_h	computational domain

Chapter 1

Introduction

1.1 Motivation

The vast majority of commercial computational fluid dynamics (CFD) software used in the engineering field today are second-order numerical methods. This is due to the great amount of research performed over the last few decades to making the finite difference (FD) and finite volume (FV) methods highly robust and well understood. As more intensive flow fields are being analyzed, the need for higher-order methods has gained popularity. The interest in higher-order methods results from the superior accuracy obtained in a shorter simulation run-time when compared to a lower-order method achieving the same accuracy through grid and temporal refinements. The higher-order methods provide higher accuracy per degree of freedom when compared to classical lower-order methods [3].

The extension to higher-order methods comes at a loss of numerical dissipation that is present in lower-order numerical approximations. The inherent dissipation in lower-order methods is due to the even derivative terms in the approximating schemes truncation error. Since the truncation error in higher-order methods contains higher-order even derivatives, the dissipation inherent in the numerical scheme decreases as the order of the method increases. Flow features that were once smoothed by the inherent numerical dissipation in low-order schemes become more susceptible to producing spurious oscillations in higher-order methods. These spurious oscillations that occur near discontinuities and sharp gradients are known as the Gibb's phenomenon.

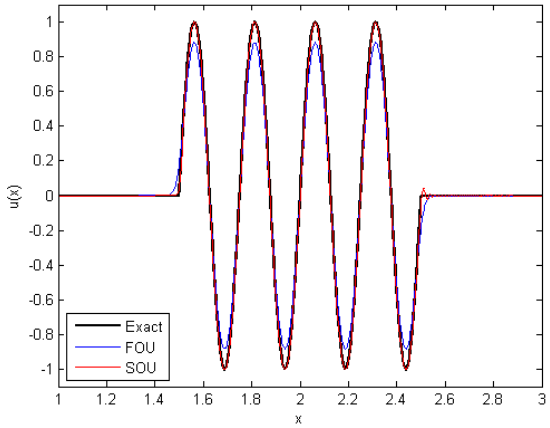
To help grasp the idea of the Gibb's phenomenon, Figure 1.1 presents the numerical solution of the hyperbolic linear convection equation for a sine wave packet (Figure 1.1a) and a jump discontinuity (Figure 1.1b). The numerical solution is obtained using two finite

difference approximations known as the first-order and the second-order upwind methods. The one-dimensional transport equation is given by

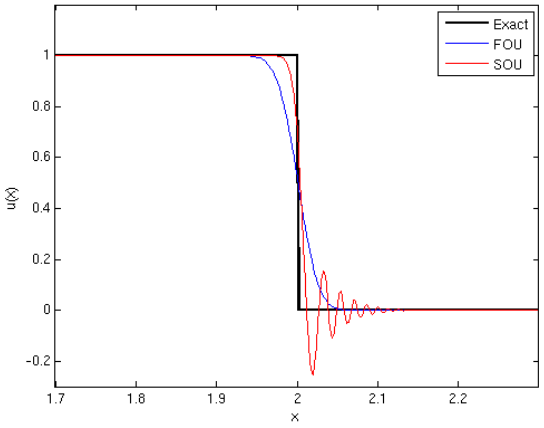
$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} = 0 \tag{1.1}$$

where a is the convective velocity which transports a flow variable, u. Two forms of errors present in the numerical approximation are the dissipation and dispersion errors. If the leading term in the truncation error contains an even derivative, then the error in the numerical method will be dominated by dissipative errors which tend to smooth high gradients in the flow variable. Conversely, if the leading derivative of the truncation error is odd, the approximation will feature dispersive errors which result in spurious oscillations around the large gradient features. The first-order upwind (FOU) approximation, with a leading second-order differential in the truncation error, is dominated by dissipation errors. Figure 1.1a shows the effect of the dissipative error as the amplitude of the wave has decreased compared to the exact solution. Figure 1.1b shows how the once sharp shock is now smeared, or widened, due to the dissipative error. The second-order upwind(SOU) approximation contains less inherent dissipation when compared to the FOU method, but instead is dominated by the dispersive error. The SOU method given in Figure 1.1a presents a good approximation in the amplitude of the wave, but minor oscillations occur where the wave meets the constant solution. Figure 1.1b truly captivates the effects of the dispersive error. The SOU method, while maintaining a closer approximation prior to the jump discontinuity, is riddled with high frequency oscillations immediately following the discontinuity.

Due to the linear nature of the transport equation, as long as the Courant-Fredrichs-Lewy (CFL) condition is met the approximation remains bounded and will not cause an unstable approximation to develop. In contrast, oscillations occurring in the highly nonlinear Euler equations of fluid motion can quickly become unstable. The resulting instabilities



(a) Wave Packet for Transport Equation



(b) Jump Discontinuity for Transport Equation

Figure 1.1: Dissipation and Dispersion in the Transport Equation

require a method to keep the unphysical oscillations from destroying the numerical approximation. Another reason of concern occurs in Figure 1.1b where the oscillations cause the transport quantity to take on negative values. This becomes an issue if, for example, the Euler equations are being modeled and density takes on a negative value. Negative values of density or pressure changes the nature of the partial differential equations locally which results in the hyperbolic nature of the equation being lost [4]. The approximation cannot be continued when such an unphysical solution is encountered.

Despite these challenges, the RMS error for the above two cases yields an important result. Table 1.1 shows that although the SOU method may not be visually appealing, the method is still more accurate than the FOU method. For the smooth wave packet test case, the solution is an order of magnitude more accurate. The FOU method is overly dissipative due to the implicit artificial viscosity that is inherent in the numerical method, while the SOU method develops spurious oscillations due to the lack of implicit artificial viscosity.

Case	Wave Packet	Jump Discontinuity
FOU	5.074E-2	3.943E-2
SOU	4.710E-3	3.078E-2

Table 1.1: RMS Error for Transport Equation Cases

Using the previous results, one approach in dealing with the oscillations is to add explicit artificial viscosity in order to stabilize the approximation while still maintaining higher-order accuracy. The explicit artificial viscosity, represented by a dissipative term, is added to the governing equation and proves to be a robust method for higher-order methods. The artificial viscosity will dampen the oscillations such that the discontinuity or sharp gradient can be properly resolved in the numerical method. The addition of the artificial viscosity controls the spurious oscillations and will prevent the numerical approximation from becoming unstable or taking on unphysical values. Unlike the implicit artificial viscosity that is applied without bias throughout the entire discretized domain, the explicit artificial viscosity should be added only in the regions where discontinuous or sharp gradients threaten the stability of the numerical method in order to ensure optimal accuracy. In order to apply artificial viscosity to the elements containing the spurious oscillations, a sensor determines where in the flow domain dissipation is required. Robustness, accuracy, and computational expense become important characteristics when determining which sensor to use.

1.2 Background

The Euler equations of fluid motion, when coupled with an appropriate equation of state, describe the conservation laws for an inviscid compressible flow. The equations are a simplification of the Navier-Stokes equations when viscosity and thermal conduction terms are neglected. The nonlinear hyperbolic Euler equations are useful for approximating flows where the effects of viscosity are negligible, as in high-speed flows and regions outside of the boundary layer. When approximating the conservative form of the Euler equations, flow discontinuities such as shock waves and contact discontinuities may develop in the solution.

Unlike the full Navier-Stokes equations, the Euler equations do not contain any natural diffusion operators. Artificial viscosity is the inclusion of dissipating terms in a numerical method, whether caused by the numerical scheme itself (implicit artificial viscosity) or by a purposely added term for numerical stability (explicit artificial viscosity). Artificial viscosity

exhibits a diffusive behavior and tends to reduce all gradients in the flow, whether physically correct or as a result of the numerical scheme [5]. Since the previously discussed Gibb's phenomenon lead to dispersive errors resulting in spurious oscillations, artificial viscosity can be used to selectively dampen the nonphysical approximations.

The DG method is a higher-order method that is closely related to the FV method. Whereas the FV method uses a piecewise linear approximation across each discretized cell, the DG method instead approximates the solution using a polynomial representation inside each discretized element. The DG method is capable of *hp*-adaptivity: increased accuracy of the numerical method can be accomplished by increasing the number of approximating elements, *h*-adaptivity, or by increasing the order of the approximating polynomial, *p*-adaptivity.

The addition of artificial viscosity is a robust method to contain the spurious oscillation that occur in higher-order schemes such as the DG method. Artificial viscosity diffuses a discontinuity or sharp gradient to the extent that the numerical method may approximate the solution without the resulting Gibb's phenomenon causing instabilities. While numerical methods approximate shocks as discontinuities, shocks occur in nature over a length scale on the order of the mean-free-path of the medium in which it occurs. Discretizing a flowfield to accurately capture a shock wave would require computing times that become unrealistic. The role of artificial viscosity is to smear the sharp profile over such a length that the numerical method can properly approximate the discretized flowfield condition.

Other methods that have been used to stabilize numerical schemes include reducing the order of the approximation down to first-order in the region of sharp gradients via the use of limiters or non-oscillatory polynomial reconstructions (ENO,WENO). Reducing the order of the approximating scheme down to first-order introduces the large amount of implicit artificial viscosity into the solution. This approach defeats the idea of using a higher-order method as the solution is reduced to first-order in the vicinity of Gibb's phenomenon. The order of the applied artificial viscosity scales like $\mathcal{O}(h)$ for first-order methods. In contrast,

the needed explicit artificial viscosity added to a high-order method to properly resolve a shock profile is of $\mathcal{O}(h/p)$ [6]. Unless mesh refinement is coupled with the first-order reduction in the vicinity of troubled cells, the accuracy of the solution may become severely degraded. Limiters, while the preferred method in the FV setting, have been found to only work in the DG scheme for structured, one-dimensional discretizations. The limiter methods developed for DG schemes are usually dependent upon grid structure, polynomial order, and the choice of quadrature sets [7]. This dependency reduces the overall robustness of the method and does not allow a general approach to solving a wide range of flow situations. Limiters also reduce the approximation to linear or constant in a wide region around the trouble elements [8]. If high-order approximations are sought then mesh refinement must then be carried out in the regions where limiting is occurring. Weighted essentially non-oscillatory (WENO) approaches have garnered interest due to their ability to retain higher-order solutions near oscillatory regions. This is accomplished by using additional degrees of freedom to resolve the sharp profiles which results in preserving the nonlinear stability [8]. Unfortunately the computational expense becomes increasingly excessive for higher-order methods. In addition, the methods have only proven compatible with one-dimensional structured grids.

In this work, the implementation of the artificial viscosity was constructed using a sensor to detect where sharp gradients or discontinuities were occurring in the approximated solution. Two separate sensors were analyzed to determine accuracy, robustness, and computational expense. The first sensor analyzed uses the modal expansion of the approximate solution inside each discretized element. The modal decay of the expansion is defined as the decreasing magnitude of the modal coefficients as the mode number increases. This method is based on the resulting numerical behavior of the modal coefficients of the solution. The sensor approximates the modal decay of the solution using an exponential equation, $|\hat{u}_n| \sim n^{-s}$. The resulting estimated decay exponent, s, is then used to determine if artificial viscosity should be applied to the approximated solutions in each element. The second method uses the residual of an associated entropy equation to determine where artificial viscosity should

be applied. The residual of the entropy equation takes a value less than zero when under-resolved features, such as shocks or steep gradients in the flow variables, are encountered. For smooth regions, the residual is nonnegative. The artificial viscosity sensor is then built on the maximum magnitude of the residual in each element. Contrary to the modal sensor, the entropy sensor is based on the physics of the flowfield by applying artificial viscosity when the Second Law of Thermodynamics is being violated.

Several test cases will be presented in one and two-dimensions. The 2-D isentropic vortex case presents a smooth solution where, given adequate resolution, no activation of artificial viscosity should occur. The 2-D Kelvin-Helmholtz test case presents a developing shear layer originating from a horizontal jet. An instability in the flow is created by small vertical velocity perturbations that are added onto the horizontal jet. The shock-free flowfield is still prone to numerical instabilities due to the sharp flow gradient present in the shear layer. The ability of each sensor to detect and diffuse these sharp regions is examined. The classic 1-D Sod shock tube is presented due to the ability to compare the numerical results with a known exact solution. The resulting flowfield is shown to develop a shock wave, a contact discontinuity, and a rarefaction wave. The shock-vortex interaction and Shu-Osher 2-D test cases present flowfields that incorporate a shock wave interacting with small-scale features. These test cases give insight into the ability of each scheme to diffuse the shock wave while not overly dampening the small-scale structures that occur in the same proximity.

Chapter 2

Discontinuous Galerkin Method

2.1 DG Method

The following analysis of the DG method follows the work of Hesthaven and Warburton in [4]. Consider a physical domain, Ω, that is discretized using $k = 1, \ldots, K$ non-overlapping, boundary conforming elements, D^k. For the structured meshes used in this work, the total number of elements is given by $K = (NI \cdot NJ \cdot NK)$, where NI, NJ, and NK are the number of elements that span each spatial direction, $\hat{i}$, $\hat{j}$, and $\hat{k}$. A one-dimensional schematic is shown in Figure 2.1. The physical domain is approximated by the discretized computational domain, Ω_h.

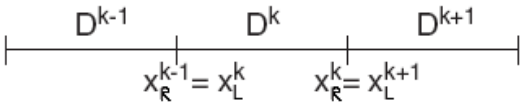


Figure 2.1: One-Dimensional Computational Domain

To assist in the explanation of the DG method, a general one-dimensional scalar conservation law of the following form is specified.

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = 0, \qquad x \in \Omega \tag{2.1}$$

The global solution, $u(x,t)$, is approximated by the direct sum of each element's piecewise continuous, N-th order polynomial approximations, $u_h(x,t)$, and is represented as

$$u(x,t) \simeq u_h(x,t) = \bigoplus_{k=1}^{K} u_h^k(x,t) \tag{2.2}$$

Let b_j^k represent a high-order local basis function and $\tilde{u}_j^k$ represents the modal coefficients, then the approximate local solution can be given by

$$u_h^k(x,t) = \sum_{j=0}^{N} \tilde{u}_j^k(t) b_j^k(x) \tag{2.3}$$

The local residual of the approximate solution can be formed on each element by

$$\mathcal{R}_h^k(x,t) = \frac{\partial u_h^k}{\partial t} + \frac{\partial f(u_h^k)}{\partial x}, \qquad x \in D^k \tag{2.4}$$

The residual is then required to vanish locally in a Galerkin sense by performing a weighted average. The Galerkin method specifies that the weighting function is chosen from the same basis as the state. The foundation of determining the local solution is then given by

$$\int_{D^k} \mathcal{R}_h^k b_i^k \mathrm{d}x = 0 \qquad i = 0, \ldots, N \tag{2.5}$$

By requiring the residual to vanish on each element, the elemental local solutions are disconnected from the local solutions on adjacent elements. This results in double-valued solutions at each element interface as shown in Figure 2.2.

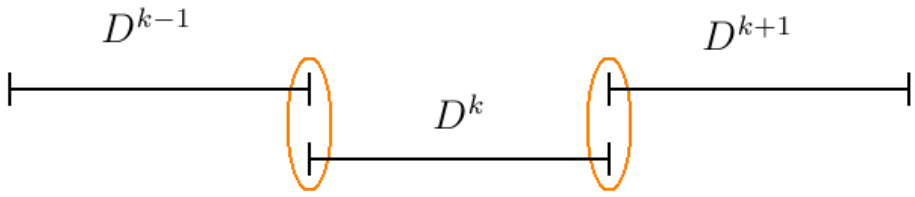


Figure 2.2: Sketch of Discontinuous Solutions at Element Interfaces

In order to obtain a global solution, Equation 2.5 is first integrated by parts and then the divergence theorem is applied resulting in

$$\int_{D^k} \mathcal{R}_h^k b_i^k \, \mathrm{d}x = \int_{D^k} b_i^k \frac{\partial f(u_h^k)}{\partial t} \, \mathrm{d}x + \int_{D^k} b_i^k \frac{\partial u_h^k}{\partial x} \, \mathrm{d}x = 0 \tag{2.6}$$

$$\int_{D^k} b_i^k \frac{\partial u_h^k}{\partial t}\ \mathrm{d}x + \int_{D^k} \frac{\partial}{\partial x}\left(b_i^k f(u_h^k)\right)\ \mathrm{d}x - \int_{D^k} \frac{\partial b_i^k}{\partial x} f(u_h^k)\ \mathrm{d}x = 0 \tag{2.7}$$

$$\int_{D^k} b_i^k \frac{\partial u_h^k}{\partial t}\ \mathrm{d}x + \oint_{\partial D^k} b_i^k f(u_h^k) n_x^k\ \mathrm{d}s - \int_{D^k} \frac{\partial b_i^k}{\partial x} f(u_h^k)\ \mathrm{d}x = 0 \tag{2.8}$$

where ∂D^k denotes the element edge.

The term $f(u_h^k)n_x^k$ presented in Equation 2.8 represents the non-unique flux that occurs at the element interfaces. This term can be replaced with a physics-based numerical flux function, f^*, that approximates the physical flux through the solution of the Riemann problem. Many numerical flux functions have been developed by the FV community for use on fluid flow problems. The numerical flux function is defined in terms of interior (-) and exterior (+) interface states and is given by

$$x \in \partial D^k, \qquad f_h^k n_x^k \rightarrow f^*(u_h^{k,-}, u_h^{k,+}; n_x^{k,-}) \tag{2.9}$$

The minimum requirements imposed for consistency are then given by

$$\begin{gathered} f^*(u_h^{k,-}, u_h^{k,+}; n_x^{k,-}) = -f^*(u_h^{k,+}, u_h^{k,-}; n_x^{k,+}) \\ f^*(u, u; n_x) \equiv f(u) n_x \end{gathered} \tag{2.10}$$

The first equation states that any flux leaving the right interface of an element, is the negative of the flux leaving the left interface of the adjacent element. This ensures a conservation of the flux between two adjacent cells. The second condition ensures that the numerical flux of two identical states is simply equal to the flux. One convenient aspect of the DG method is that the boundary conditions are simply applied through the $u_h^{k,+}$ term in the numerical flux function. The stability of the DG method is enforced through the local flux choice [4].

With the numerical flux function inserted, the approximating semi-discrete DG scheme takes the form

$$\int_{D^k} b_i^k \frac{\partial u_h^k}{\partial t} \mathrm{d}x + \oint_{\partial D^k} b_i^k f^*(u_h^{k,-}, u_h^{k,+}; n_x^{k,-}) \mathrm{d}s - \int_{D^k} \frac{\partial b_i^k}{\partial x} f_h^k \mathrm{d}x = 0 \qquad (2.11)$$
$$i = 0, \ldots, N, \qquad k = 1, \ldots, K$$

The solution of Equation 2.11 are piecewise smooth inside each element, but at the element interfaces the solution is discontinuous. The addition of the numerical flux function allows the discontinuous elements to communicate and the global solution can be obtained from each individual element solution.

2.1.1 Choice of Basis Function

Consider a standard element such that $x \to \xi \in (-1, 1)$, and expand the state using a degree N polynomial function space such that, $u_h(\xi) \in \mathbb{P}_N$. A transformation map, $\mathcal{M}^k$, is used to get from the standard element to each physical element, D^k, by

$$x \in D^k : \qquad x = \mathcal{M}^k(\xi) \qquad (2.12)$$

and is shown in Figure 2.3.

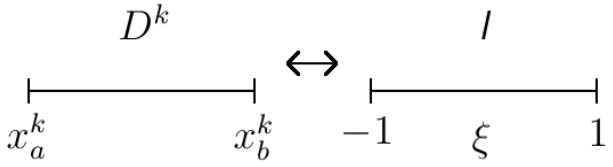


Figure 2.3: 1-D Transformation from Physical to Standard Element

Using a linear interpolation, the transformation map is given by

$$\mathcal{M}^k(\xi) = x_a^k + \frac{1+\xi}{2}(x_b^k - x_a^k) \qquad (2.13)$$

The choice of the local basis is important because it is the local basis which gives the DG method it's accuracy. At each element, the local solution can be represented as a N^{th}-order polynomial by either a modal (2.14) or nodal basis (2.15).

$$u_h(\xi) = \sum_{m=0}^{N} \tilde{u}_m \pi_m(\xi) \tag{2.14}$$

$$u_h(\xi) = \sum_{n=0}^{N} \hat{u}_n \ell_n(\xi) \tag{2.15}$$

where π_m is a local polynomial basis, and ℓ_n is a Lagrange interpolating polynomial. The weighting functions are also selected from either the modal class, $b_i \to \pi_i$, or the nodal class, $b_i \to \ell_i$, where $\pi_m, \ell_n \in \mathbb{P}_N$.

Modal Basis

The modal coefficients, $\tilde{u}_m$, can be found using an L^2 projection,

$$\begin{aligned}
\int_{-1}^{1} \pi_i(\xi) u(\xi) \mathrm{d}\xi &= \int_{-1}^{1} \pi_i(\xi) u_h(\xi) \mathrm{d}\xi \\
&= \int_{-1}^{1} \pi_i(\xi) \left(\sum_{m=0}^{N} \tilde{u}_m \pi_m(\xi \right) \mathrm{d}\xi \\
&= \sum_{m=0}^{N} \left(\int_{-1}^{1} \pi_i(\xi) \pi_m(\xi) \mathrm{d}\xi \right) \tilde{u}_m \\
&= \sum_{m=0}^{N} M_{im} \tilde{u}_m
\end{aligned} \tag{2.16}$$

where M_{im} is the mass matrix. To recover the modal coefficients, the inverse of the mass matrix must be computed. An orthonormal basis is used to insure that the mass matrix is well-conditioned in order to preserve accuracy. The three term recurrence relation is used

to find an orthonormal basis and is given by [9]

$$\begin{aligned} \sqrt{\beta_{m+1}}\pi_{m+1}(\xi) &= (\xi - \alpha_m)\pi_m(\xi) - \sqrt{\beta_{m-1}}\pi_{m-1}(\xi) \\ \pi_{-1} &\doteq 0, \qquad \pi_0 \doteq 1/\sqrt{\beta_0} \end{aligned} \tag{2.17}$$

for $m = 0, \ldots, N-1$. By choosing the coefficients (α_m, β_m), different types of polynomials can be obtained from Equation 2.17 such as the Legendre, Chebychev, or Hermite polynomials. Following the work of Hesthaven [4], the Legendre polynomials were chosen to form a hierarchical modal basis. The coefficients that result in the Legendre polynomials are

$$\alpha_m = 0, \qquad \beta_m = \begin{cases} 2 & \text{if } \ m = 0 \\ 1/(4 - m^{-2}) & \text{if } \ m \geq 1 \end{cases} \tag{2.18}$$

An important property, resulting from the use of Legendre polynomials, is given by

$$\int_{-1}^{1} \pi_i(\xi)\pi_j(\xi)d\xi = \delta_{ij} \doteq \begin{cases} 1 & \text{if } \ i = j \\ 0 & \text{if } \ i \neq j \end{cases} \tag{2.19}$$

Nodal Basis

The nodal basis, given in Equation 2.15, uses the Lagrange interpolating polynomials. The resulting polynomial takes the form

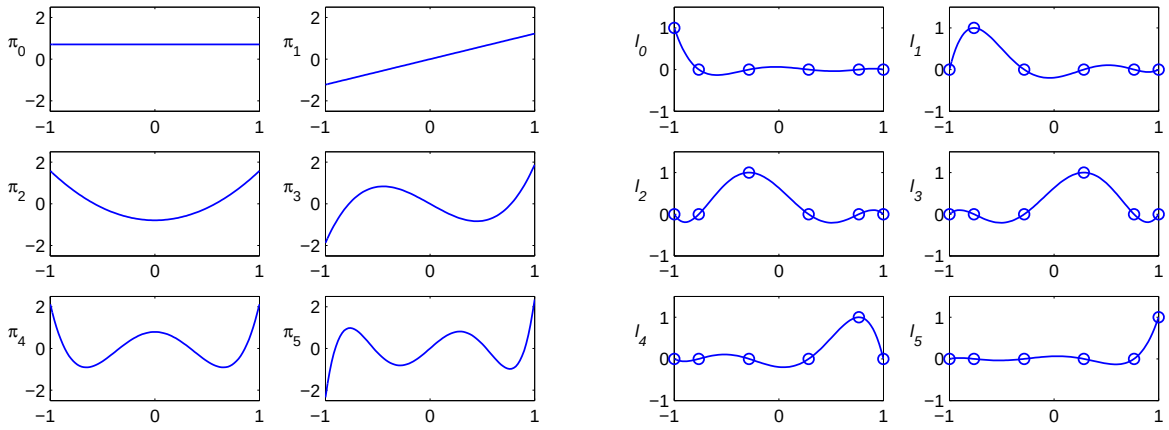
$$\ell_n(\xi) = \prod_{\substack{i=0 \\ i \neq n}}^{N} \frac{\xi - \xi_i}{\xi_n - \xi_i} \tag{2.20}$$

for the nodal locations given by $\xi_o, \ldots, \xi_N$. The important property resulting from using the Lagrange polynomials is given by

$$\ell_j(\xi_i) = \delta_{ij} \doteq \begin{cases} 1 & \text{if } \ i = j \\ 0 & \text{if } \ i \neq j \end{cases} \tag{2.21}$$

Modal/Nodal Relationship

While the modal and nodal representations are both mathematically equivalent, they are computationally different. Figure 2.4 shows the representations of the modal and nodal basis for a fifth-order polynomial. The modal basis, using Legendre polynomials, is a hierarchical basis. This implies that an N^{th}-order approximation will use $N+1$ curves of increasing polynomial order. Figure 2.4a shows a 5^{th}-order modal approximation that is composed of a constant line and then the curves of polynomial orders $1, 2, \ldots, 5$. When computing a solution in an element, the modal representation will use the combination of the $N+1$ curves to obtain a solution on the element. The nodal representation will find a solution by generating $N+1$ curves of each polynomial order N. The nodal representation then computes an approximation for only the node whose solution is being required and zeros at the other node locations.



(a) A modal basis with $\pi_m(\xi) \in \mathbb{P}_5$ (b) A nodal basis with $\ell_n(\xi) \in \mathbb{P}_5$

Figure 2.4: Representation of Modal and Nodal Basis for $N = 5$

A connection between the modes, $\tilde{u}$, and the nodal values, $\hat{u}$, can be made by defining the Vandermonde matrix (V). The Vandermonde matrix is formed by evaluating the Legendre polynomials at the same location that define the Lagrange polynomials. The Vandermonde

matrix is then evaluated as

$$V_{ij} \doteq \pi_j(\xi_i) \tag{2.22}$$

Substituting the Vandermonde matrix into the modal and nodal representations yields

$$u_h(\xi_i) = \sum_{m=0}^{N} \tilde{u}_m \pi_m(\xi_i) = \sum_{n=0}^{N} \hat{u}_n \ell_n(\xi_i) \iff \sum_{m=0}^{N} V_{im} \tilde{u}_m = \sum_{n=0}^{N} \delta_{in} \hat{u}_n \tag{2.23}$$

Rewriting these in matrix-vector form results in

$$\vec{u}_h \doteq u_h(\vec{\xi}) = \hat{u} = V\tilde{u} \quad \iff \quad \tilde{u} = V^{-1}\hat{u} \tag{2.24}$$

In order to successfully transition between the modal and nodal values the Vandermonde matrix must be well-conditioned. The conditioning of the Vandermonde matrix is dependent on the choice of the nodal interpolation points. The Lebesque constant, given by

$$\Lambda = \max_r \sum_{i=1}^{N_p} |\ell_i(\xi)| \tag{2.25}$$

indicates how far away from the interpolation may from the best possible polynomial representation[4]. Minimizing the Lebesque constant will result in the best possible interpolation points so that the Vandermonde matrix is well-conditioned. The resulting points are the Legendre-Gauss-Lobatto (LGL) quadrature points [10].

2.1.2 Numerical Quadrature

Following the work by Gautschi [9], the integration of a function, f, may be approximated via summation by

$$\int_{-1}^{1} f(\xi) \mathrm{d}\xi = \sum_{j=1}^{N_p} \omega_j f(\xi_j) \tag{2.26}$$

where ω_j are the quadrature weights and ξ_j are the quadrature nodes. A key property of polynomials is that Gaussian quadrature is exact.

The Gauss rule, which does not account for the interface nodes since $f \in \mathbb{P}_{2N_p-1}$ (with $-1 < \xi_1, \xi_{N_p} < 1$), is given by

$$\int_{-1}^{1} f(\xi)\mathrm{d}\xi \simeq \sum_{i=1}^{N_p} \omega_i f(\xi_i) \tag{2.27}$$

The nodes and weights are found by using the polynomial recursion formula. Begin by letting $\vec{\pi}(\xi) \doteq [\pi_0(\xi), \pi_1(\xi), \ldots, \pi_{p-1}(\xi)]^T$ and define a Jacobi matrix

$$J_p \doteq \begin{bmatrix} \alpha_0 & \sqrt{\beta_1} & & 0 \\ \sqrt{\beta_1} & \alpha_1 & \sqrt{\beta_2} & \\ & & \ddots & \\ 0 & & \sqrt{\beta_{p-1}} & \alpha_{p-1} \end{bmatrix} \tag{2.28}$$

The first p terms of the recursion in matrix-vector form is

$$\xi\vec{\pi}(\xi) = J_p\vec{\pi}(\xi) + \sqrt{\beta_p}\pi_p(\xi)\vec{e}_p \tag{2.29}$$

Finding the eigenvalues, λ_i, and the corresponding eigenvectors, $\vec{v}_{(i)}$, of the Jacobi matrix J_p, then the nodes and weights of the p-point Gauss formula are

$$\xi_i = \lambda_i, \quad \omega_i = \beta_0(v_{(i)1})^2, \quad i = 1, \ldots, p \tag{2.30}$$

with $-1 < \xi_1$ and $\xi_{N_p} < 1$.

The Gauss-Lobatto rule, which includes the interface nodes since $f \in \mathbb{P}_{2N_p-3}$ (with $\xi_0 = 1, \xi_{N_p+1} = 1$), is given by

$$\int_{-1}^{1} f(\xi)\mathrm{d}\xi = \omega_0 f(\xi_0) + \sum_{i=1}^{N_p} \omega_i f(\xi_i) + \omega_{N_p+1} f(\xi_{N_p+1}) \tag{2.31}$$

The polynomial recursion formula is also used to find the nodes and weights for the Gauss-Lobatto rule. A Jacobi-Lobatto matrix is given by

$$J_{p+2}^{L} \doteq \begin{bmatrix} J_{p+1} & \sqrt{\beta_{p+1}^{*}}\vec{e}_{p+1} \\ \sqrt{\beta_{p+1}^{*}}\vec{e}_{p+1}^{T} & \alpha_{p+1}^{*} \end{bmatrix} \tag{2.32}$$

where for Legendre polynomials the coefficients are given as

$$\alpha_{p+1}^{*} = 0, \quad \beta_{p+1}^{*} = 1/2 + 1/(4p+2) \tag{2.33}$$

Finding the eigenvalues, λ_i^L, and the corresponding eigenvectors, $\vec{v}_{(i)}^L$, of the Jacobi-Lobatto matrix J_{p+2}^L, then the nodes and weights of the $p+2$-point Gauss-Lobatto formula are

$$\xi_i = \lambda_i^L, \quad \omega_i = \beta_0 (v_{(i)1}^L)^2, \quad i = 0, \ldots, p+1 \tag{2.34}$$

with $\xi_0 = -1$ and $\xi_{N_p+1} = 1$.

2.1.3 Operator Form

Restating the model problem and DG discretization:

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = 0 \tag{2.35}$$

$$\int_{D^k} b_i^k \frac{\partial u_h^k}{\partial t} \mathrm{d}x - \int_{D^k} \frac{\partial b_i^k}{\partial x} f_h^k \mathrm{d}x + \oint_{\partial D^k} b_i^k f^*(u_h^{k,-}, u_h^{k,+}; n_x^{k,-}) \mathrm{d}s = 0 \tag{2.36}$$

In order to evaluate the integrals on the physical elements, a transformation for the differentials is given by

$$x \in D^k : \quad \mathrm{d}x = \mathcal{J}^k d\xi, \quad \mathcal{J}^k = \frac{x_b^k - x_a^k}{2} \tag{2.37}$$

where $\mathcal{J}^k$ is the Jacobian of the transformation. The state and flux decomposition is found by using the coordinate transformation so that

$$u_h^k(x) = u_h(\mathcal{M}^k(\xi)) = \sum_{j=0}^{N} \tilde{u}_j^k(t) b_j(\xi) \tag{2.38}$$

$$f_h^k(x) = \sum_{j=0}^{N} \tilde{f}_j^k(t) b_j^k(\xi) \tag{2.39}$$

where $\mathcal{M}$ is the transformation map defined in Equation 2.12.

The time derivative present in Equation 2.36 are now evaluated using the modal expansion given in Equation 2.3:

$$\begin{aligned}
\int_{D^k} b_i^k \frac{\partial u_h^k}{\partial t} \mathrm{d}x &= \int_{D^k} b_i^k(x) \frac{\partial}{\partial t} \left(\sum_{j=0}^{N} \tilde{u}_j^k(t) b_j^k(x) \right) \mathrm{d}x \\
&= \sum_{j=0}^{N} \left(\int_{D^k} b_i^k(x) b_j^k(x) \mathrm{d}x \right) \frac{\mathrm{d}\tilde{u}_j^k}{\mathrm{d}t} \\
&= \sum_{j=0}^{N} \left(\int_{-1}^{1} b_i(\xi) b_j(\xi) \mathcal{J}^k d\xi \right) \frac{\mathrm{d}\tilde{u}_j^k}{\mathrm{d}t} \\
&= \mathcal{J}^k \sum_{j=0}^{N} M_{ij} \frac{\mathrm{d}\tilde{u}_j^k}{\mathrm{d}t}
\end{aligned} \tag{2.40}$$

where M_{ij} is the mass matrix defined by

$$M_{ij} \doteq \int_{-1}^{1} b_i^k(\xi) b_j^k(\xi) \mathrm{d}\xi \tag{2.41}$$

The spatial derivative term goes through the same process as the time derivative term, and is given by

$$\begin{aligned}\int_{D^k} \frac{\mathrm{d}b_i^k}{\mathrm{d}x} f_h^k \mathrm{d}x &= \int_{D^k} b_i^k(x) \frac{\mathrm{d}b_i^k(x)}{\mathrm{d}x} \left(\sum_{j=0}^{N} \tilde{f}_j^k b_j^k(x) \right) \mathrm{d}x \\ &= \sum_{j=0}^{N} \left(\int_{D^k} \frac{\mathrm{d}b_i^k(x)}{\mathrm{d}x} b_j^k(x) \mathrm{d}x \right) \tilde{f}_j^k \\ &= \sum_{j=0}^{N} \left(\int_{-1}^{1} \frac{\mathrm{d}b_i(\xi)}{\mathrm{d}\xi} b_j(\xi) \mathrm{d}\xi \right) \tilde{f}_j^k \\ &= \sum_{j=0}^{N} (S^T)_{ij} \tilde{f}_j^k \end{aligned} \tag{2.42}$$

where S_{ij} is the stiffness matrix and is defined by

$$S_{ij}^T \doteq \int_{-1}^{1} b_i(\xi) \frac{\mathrm{d}b_j(\xi)}{\mathrm{d}\xi} \mathrm{d}\xi \tag{2.43}$$

Noticing the closed interval, the numerical flux, f^*, does not exist over the whole element, D^k, but only on the interface ∂D^k. The numerical flux is expanded out regardless as

$$f^*(s) = f^*(u_h^{k,-}, u_h^{k,+}; n_x^{k,-}) \Big|_s = \sum_{j=0}^{N} \widetilde{(f_s^*)}_j^k b_j(s) \tag{2.44}$$

where the variable, s, simply denotes the restriction $x \in \partial D^k$. The remaining formulation is given by

$$\begin{aligned}
&\oint_{\partial D^k} b_i^k f^*(u_h^{k,-}, u_h^{k,+}; n_x^{k,-}) \mathrm{d}s \\
&\qquad = [b_i^k f^*(u_h^{k,-}, u_h^{k,+}; n_x^{k,-})]_{x_a^k} + [b_i^k f^*(u_h^{k,-}, u_h^{k,+}; n_x^{k,-})]_{x_b^k} \\
&\qquad = b_i^k(x_a^k) \sum_{m=0}^{N} \widetilde{(f_a^*)}_m^k b_m^k(x_a^k) + b_i^k(x_b^k) \sum_{n=0}^{N} \widetilde{(f_b^*)}_n^k b_n^k(x_b^k) \\
&\qquad = \sum_{m=0}^{N} \left(b_i^k(x_a^k) b_m^k(x_a^k) \right) \widetilde{(f_a^*)}_m^k + \sum_{n=0}^{N} \left(b_i^k(x_b^k) b_n^k(x_b^k) \right) \widetilde{(f_b^*)}_n^k \\
&\qquad = \sum_{m=0}^{N} \left(b_i(-1) b_m(-1) \right) \widetilde{(f_a^*)}_m^k + \sum_{n=0}^{N} \left(b_i(1) b_n(1) \right) \widetilde{(f_b^*)}_n^k \\
&\qquad = \sum_{m=0}^{N} (L_a)_{im} \widetilde{(f_a^*)}_m^k + \sum_{n=0}^{N} (L_b)_{in} \widetilde{(f_b^*)}_n^k
\end{aligned} \tag{2.45}$$

The new variables $(L_a)_{ij}$ and $(L_b)_{ij}$ are the lifting matrices. Since the solution is approximated by a N^{th}-order polynomial, the data obtained at the boundaries must be extended throughout the rest of the domain. These matrices "lift" the lower dimensional boundary data to the higher dimensional element data. The lifting matrices are given by

$$(L_a)_{ij} \doteq b_i(-1) b_m(-1), \qquad (L_b)_{ij} \doteq b_i(1) b_n(1) \tag{2.46}$$

where $(L_a)_{ij}$ corresponds to the left interface of the standard element and $(L_a)_{ij}$ corresponds to the right interface of the standard element With the basis expansion complete, Equation 2.36 can now be expressed as

$$\mathcal{J}^k \sum_{j=0}^{N} M_{ij} \frac{\mathrm{d}\tilde{u}_j^k}{\mathrm{d}t} - \sum_{p=0}^{N} (S^T)_{ip} \tilde{f}_p^k + \sum_{m=0}^{N} (L_a)_{im} \widetilde{(f_a^*)}_m^k + \sum_{n=0}^{N} (L_b)_{in} \widetilde{(f_b^*)}_n^k = 0 \tag{2.47}$$

The DG discretization is further simplified into the matrix vector form by using the mass, stiffness, and lifting matrices:

$$\mathcal{J}^k M \frac{\mathrm{d}\tilde{u}^k}{\mathrm{d}t} - S^T \tilde{f}^k + L_a \widetilde{(f_a^*)}^k + L_b \widetilde{(f_b^*)}^k = 0 \tag{2.48}$$

Equation 2.48 represents a ODE in time after the spatial discretization has been carried out. This final form is the method implemented in the current computational work.

2.1.4 Element-wise Operations

The current work employed the nodal basis for the computational implementation of the DG method. At element interfaces, the nodal approach is more efficient due to the presence of nodal values at the interface locations. The modal approach would require extrapolation in order to obtain the interface values needed for the numerical flux function. The nodal approach, which outputs point-wise physical values, is more convenient to deal with when the flux is a nonlinear function. The advantage of the modal approach lies in it having the property of hierarchical construction. This would be a benefit if resolution adaptivity where to be implemented.

Using the same methods that allow the representation of the function u_h by,

$$u_h(\xi_i) = \sum_{j=0}^{N} \tilde{u}_j \pi_j(\xi) = \sum_{i=0}^{N} \hat{u}_i \ell_i(\xi) \tag{2.49}$$

then the same method is applicable to the basis functions resulting in the expression

$$\pi_j(\xi) = \sum_{i=0}^{N} \widehat{(\pi_j)}_i \ell_i(\xi) \tag{2.50}$$

The nodal coefficient is then simply the pointwise value of the function:

$$\widehat{(\pi_j)}_i = \pi_j(\xi_i) \doteq V_{ij} \tag{2.51}$$

where V_{ij} is again the Vandermonde matrix. The relationship between the nodal and the modal basis functions are given by

$$\pi_j(\xi) = \sum_{i=0}^{N} V_{ij}\ell_i(\xi) = \sum_{i=0}^{N}(V^T)_{ji}\ell_i(\xi) \Longleftrightarrow \ell_i(\xi) = \sum_{j=0}^{N}(V^{-T})_{ij}\pi_j(\xi) \tag{2.52}$$

To formulate the DG method using the nodal expression relies on the relationship

$$\ell_i(\xi) = \sum_{j=0}^{N}(V^{-T})_{ij}\pi_j(\xi) \tag{2.53}$$

$$\begin{aligned}
M_{ij} &\doteq \int_{-1}^{1} \ell_i(\xi)\ell_j(\xi)\mathrm{d}\xi \\
&= \int_{-1}^{1} \sum_{m=0}^{N}(V^{-T})_{im}\pi_m(\xi) \sum_{n=0}^{N}(V^{-T})_{jn}\pi_n(\xi)\mathrm{d}\xi \\
&= \sum_{m=0}^{N}\sum_{n=0}^{N}(V^{-T})_{im}\left(\int_{-1}^{1}\pi_m(\xi)\pi_n(\xi)\mathrm{d}\xi\right)(V^{-1})_{nj} \\
&= \sum_{m=0}^{N}\sum_{n=0}^{N}(V^{-T})_{im}\delta_{mn}(V^{-1})_{nj} \\
&= \sum_{m=0}^{N}\sum_{n=0}^{N}(V^{-T})_{im}(V^{-1})_{mj} \qquad \Longleftrightarrow \qquad M = (VV^T)^{-1}
\end{aligned} \tag{2.54}$$

In order to express the stiffness matrix in a nodal form, first a differentiation matrix that transforms a point value to the derivative at that point is defined by

$$\begin{aligned}
D_{ij} &\doteq \ell_i'(\xi_j) \\
&= \sum_{m=0}^{N}(V^{-T})_{jm}\pi_m'(\xi_i) \\
&= \sum_{m=0}^{N}(V^{-T})_{jm}(V_\xi)_{im} \\
&= \sum_{m=0}^{N}(V_\xi)_{im}(V^{-1})_{mj} \qquad \Longleftrightarrow \qquad D = (V_\xi V^{-1})
\end{aligned} \tag{2.55}$$

The stiffness matrix in a nodal basis is given by

$$S_{ij} \doteq \int_{-1}^{1} \ell_i(\xi)\ell'_j(\xi)\mathrm{d}\xi \qquad \Longleftrightarrow \qquad (S^T)_{ij} \doteq \int_{-1}^{1} \ell'_i(\xi)\ell_j(\xi)\mathrm{d}\xi \tag{2.56}$$

Using the mass matrix with the differentiation matrix defined in Equation 2.55 the stiffness matrix can be formulated by

$$\begin{aligned}
\sum_{n=0}^{N} M_{ij}D_{nj} &= \sum_{n=0}^{N} \left(\int_{-1}^{1} \ell_i(\xi)\ell_n(\xi)\mathrm{d}\xi \right) \ell'_j(\xi_n) \\
&= \int_{-1}^{1} \ell_i(\xi) \left(\sum_{n=0}^{N} \ell'_j(\xi_n)\ell_n(\xi) \right) \mathrm{d}\xi \\
&= \int_{-1}^{1} \ell_i(\xi) \left(\sum_{n=0}^{N} \widehat{\left(\ell'_j\right)}_n \ell_n(\xi) \right) \mathrm{d}\xi \\
&= \int_{-1}^{1} \ell_i(\xi)\ell'_j(\xi)\mathrm{d}\xi \\
&= S_{ij} \qquad \Longleftrightarrow \qquad S = (MD)
\end{aligned} \tag{2.57}$$

Since the transpose of the stiffness matrix is used in the DG implementation, the formula for the transpose of the stiffness matrix is given by

$$\begin{aligned}
S = (MD) \qquad \Longleftrightarrow \qquad S^T &= (D^T M^T) \\
&= (V_\xi V^{-1})^T (V^{-T} V^{-1})^T \\
&= (V_\xi V^{-1})^T (V^{-T} V^{-1}) \\
&= (V_\xi V^{-1})^T (V V^T)^{-1}
\end{aligned} \tag{2.58}$$

In order to incorporate the solution of the numerical flux function at the interface throughout the element, the lifting matrices are formulated for the two element interfaces given by "a" and "b". The "a" interface corresponds to the standard element left interface location of -1 and the "b" interface corresponds to the standard element right interface location of 1. The

formulation of the "a" and "b" interfaces follow the same procedure by

$$\begin{aligned}(L_a)_{ij} &\doteq \ell_i(-1)\ell_j(-1) & (L_b)_{ij} &\doteq \ell_i(1)\ell_j(1)\\ &= \ell_i(\xi_0)\ell_j(\xi_0) & &= \ell_i(\xi_N)\ell_j(\xi_N)\\ &= \delta_{0i}\delta_{0j} & &= \delta_{Ni}\delta_{Nj}\end{aligned} \tag{2.59}$$

Defining the numerical flux function using the interface locations of "a" and "b", their formulation is given by

$$\begin{aligned}(f_a^*)^k &= \sum_{j=0}^{N} \widehat{(f_a^*)}_j^k \ell_j(-1) & (f_b^*)^k &= \sum_{j=0}^{N} \widehat{(f_b^*)}_j^k \ell_j(1)\\ &= \sum_{j=0}^{N} \widehat{(f_a^*)}_j^k \ell_j(\xi_0) & &= \sum_{j=0}^{N} \widehat{(f_b^*)}_j^k \ell_j(\xi_N)\\ &= \sum_{j=0}^{N} \widehat{(f_a^*)}_j^k \delta_{0j} & &= \sum_{j=0}^{N} \widehat{(f_b^*)}_j^k \delta_{Nj}\\ &= \widehat{(f_a^*)}_0^k & &= \widehat{(f_b^*)}_N^k\end{aligned} \tag{2.60}$$

Combining the results for the lifting matrix and the flux function at the "a" interface results in

$$\sum_{i=0}^{N}(L_a)_{ij}\widehat{(f_a^*)}_j^k = \sum_{i=0}^{N}\delta_{0i}\delta_{0j}\delta_{0j}\widehat{(f_a^*)}_j^k = \sum_{i=0}^{N}\delta_{i0}\widehat{(f_a^*)}_0^k \tag{2.61}$$

$$L_a\widehat{(f_a^*)}^k = \begin{bmatrix}1 & 0 & \dots & 0\\ 0 & 0 & \dots & 0\\ \vdots & & \ddots & \vdots\\ 0 & & \dots & 0\end{bmatrix}\begin{Bmatrix}\widehat{(f_a^*)}_0^k\\ 0\\ \vdots\\ 0\end{Bmatrix} = \widehat{(f_a^*)}_0^k\begin{Bmatrix}1\\ 0\\ \vdots\\ 0\end{Bmatrix} = \widehat{(f_a^*)}_0^k\vec{e}_0$$

Following the same method as above, the "b" interface becomes

$$\sum_{i=0}^{N}(L_b)_{ij}\widehat{(f_b^*)}_j^k = \sum_{i=0}^{N}\delta_{Ni}\delta_{Nj}\delta_{Nj}\widehat{(f_b^*)}_j^k = \sum_{i=0}^{N}\delta_{iN}\widehat{(f_b^*)}_N^k \tag{2.62}$$

$$L_b\widehat{(f_b^*)}^k = \begin{bmatrix} 0 & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & & \dots & 1 \end{bmatrix} \begin{Bmatrix} 0 \\ \vdots \\ 0 \\ \widehat{(f_b^*)}_N^k \end{Bmatrix} = \widehat{(f_b^*)}_N^k \begin{Bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{Bmatrix} = \widehat{(f_b^*)}_N^k \vec{e}_N$$

With the results from the above formulations, the local form of the 1-D DG statement in using a nodal basis becomes

$$\mathcal{J}^k M \frac{\mathrm{d}\hat{u}^k}{\mathrm{d}t} - S^T \hat{f}^k + L_a (f_a^*)^k + L_b (f_b^*)^k = 0 \tag{2.63}$$

and resummarizing the components as

$$\begin{aligned} \text{Mass Matrix} \qquad & M = (VV^T)^{-1} \\ \text{Stiffness Martrix} \qquad & S^T = (V_\xi V^{-1})^T (VV^T)^{-1} \\ \text{Lifting Martrices - (“a”)} \qquad & L_a = \vec{e}_0 \\ \text{Lifting Martrices - (“b”)} \qquad & L_b = \vec{e}_N \\ \text{Vandermonde Martrix} \qquad & V_{ij} \doteq \pi_j(\xi_i) \\ \text{Derivative Vandermonde Martrix} \qquad & (V_\xi)_{ij} \doteq \pi_j'(\xi_i) \end{aligned} \tag{2.64}$$

Chapter 3

Numerical Implementation

3.1 Conservation Laws

The DG code developed in this work was used to approximate the Euler equations. The Euler equations are the non-linear, hyperbolic equations that describe the conservation of mass, momentum, and energy using the continuum hypothesis for fluid flows. The Euler equations are a simplification of the Navier-Stokes equation in which viscosity and thermal conduction terms are neglected. The assumption of inviscid flow is a standard simplification for high-speed flows in which the effect of viscosity becomes increasingly negligible with increasing Reynolds number. The challenge of numerically approximating the Euler equations is the occurrence of localized discontinuities, such as shocks and contact discontinuities, which may occur in the approximated solution.

The Euler equations, which express the conservation of mass (Eq. 3.1), momentum (Eq. 3.2 & 3.3), and energy (Eq. 3.4), are given in two dimensions as

$$\frac{\partial \rho}{\partial t} + \frac{\partial (\rho u)}{\partial x} + \frac{\partial (\rho v)}{\partial y} = 0 \tag{3.1}$$

$$\frac{\partial (\rho u)}{\partial t} + \frac{\partial (\rho u^2 + p)}{\partial x} + \frac{\partial (\rho uv)}{\partial y} = 0 \tag{3.2}$$

$$\frac{\partial (\rho v)}{\partial t} + \frac{\partial (\rho uv)}{\partial x} + \frac{\partial (\rho v^2 + p)}{\partial y} = 0 \tag{3.3}$$

$$\frac{\partial (\rho e_0)}{\partial t} + \frac{\partial u(\rho e_0 + p)}{\partial x} + \frac{\partial v(\rho e_0 + p)}{\partial y} = 0 \tag{3.4}$$

where the conserved variables are density, ρ, momentum, ρu, and the total energy of the gas, e_0. Rearranging the equations in flux vector (strong conservation) form results in

$$\frac{\partial U}{\partial t} + \frac{\partial}{\partial x}F(U) + \frac{\partial}{\partial y}G(U) = 0 \tag{3.5}$$

The state vector, U, and the inviscid flux vectors, F and G, are expressed by

$$U = \begin{Bmatrix} \rho \\ \rho u \\ \rho v \\ \rho e_0 \end{Bmatrix} \qquad F = \begin{Bmatrix} \rho u \\ \rho u^2 + p \\ \rho v u \\ (\rho e_0 + p)u \end{Bmatrix} \qquad G = \begin{Bmatrix} \rho v \\ \rho u v \\ \rho v^2 + p \\ (\rho e_0 + p)v \end{Bmatrix}$$

Noting that the ratio of specific heats is defined by $\gamma = c_p/c_v$, the system of equations can be closed with the ideal gas equation of state

$$p = (\gamma - 1)\left(\rho e_0 - \rho\frac{u^2 + v^2}{2}\right) \tag{3.6}$$

3.2 Shock-Capturing via Artificial Viscosity

In the approach of shock-capturing methods, shock waves are computed as part of the solution using the conservative form of the Euler equations. The conservative form of the Euler equations are required so that the solution will satisfy the Rankine-Hugoniot equations [5].

When shock-capturing methods are used, high frequency solutions will form near shock waves and other high gradient features. The high frequency solutions are caused by the Gibbs phenomena, and a method must be used to keep the solution from taking on unreal values. In this work, artificial viscosity is used to smooth out the high frequency areas in the solution. The resulting piecewise discontinuous artificial viscosity that is applied to the needed elements is accomplished with the addition of an unphysical diffusion term to the

Euler equations such that

$$\frac{\partial U}{\partial t} + \frac{\partial}{\partial x}F(U) + \frac{\partial}{\partial y}G(U) = \nabla \cdot (\nu \nabla U) \tag{3.7}$$

The resulting diffusive scheme is numerically consistent with damping methods provided by some schemes. Expanding out the viscosity term on the right hand side yields

$$\frac{\partial U}{\partial t} + \frac{\partial}{\partial x}F(U) + \frac{\partial}{\partial y}G(U) = \frac{\partial}{\partial x}\left(\nu \frac{\partial U}{\partial x}\right) + \frac{\partial}{\partial y}\left(\nu \frac{\partial U}{\partial y}\right) \tag{3.8}$$

where the artificial viscosity is given by ν.

Due to the unbiased nature that artificial viscosity possesses, an appropriate method is required to ensure that the diffusion is applied to the unphysical high-frequency solutions instead of the physically correct features present in the flowfield. With the possibility of artificial viscosity dampening relevant information, the method in which diffusion is applied becomes vital to ensuring that the artificial viscosity is applied only in areas where it is required. Although the Gibb's phenomenon tends to produce oscillations around sharp gradients, the accuracy of the approximation is still preserved. The key to ensuring a robust method while still maintaining accuracy is to apply the smallest amount of artificial viscosity such that the solution remains stable.

With the addition of the artificial viscosity components given in Equation 3.8, the numerical flux function used for the inviscid flux will be avoided for these terms. When the numerical flux function encounters the artificial viscosity terms in Equation 3.8, simple averages of the adjacent states are used in place of the numerical flux function and is shown by

$$\begin{aligned} \mathcal{U} &= \frac{U^L + U^R}{2} \\ \mathcal{F}_{vis} &= \frac{F_{artvis}(U^L) + F_{artvis}(U^R)}{2} \end{aligned} \tag{3.9}$$

The present work uses a simple nonlinear artificial flux, $\nabla \cdot (\nu \nabla U)$, applied to each state. It is worth noting that other methods can be used to apply the artificial viscosity to the conservation equations. Zingan [11] uses the following viscous flux to augment the inviscid flux, and is given by

$$\mathbf{g}(\mathbf{c}) = \begin{Bmatrix} -\nu \nabla \rho \\ -\mu \nabla_s \mathbf{u} \\ -\mu \nabla_s \mathbf{u} \cdot \mathbf{u} - \kappa \nabla T \end{Bmatrix} \tag{3.10}$$

where $\nabla_s \mathbf{u}$ is a symmetric tensor of rank 2 given by

$$(\nabla_s \mathbf{u})_{ij} = \frac{1}{2} \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \tag{3.11}$$

The new artificial viscosity terms are ν a diffusion viscosity, μ a dynamic viscosity, and κ a thermal viscosity. The temperature, T, is found using the standard Equation of State for a perfect gas.

3.3 Normalization of Flow Variables

The flow variables present in the Euler equations are normalized in this work for convenience. The normalization assumes a uniform gas constant, $R = R_\infty$, and specific heat ratio, $\gamma = \gamma_\infty$. The choice of normalization used in this work is

$$\begin{aligned}
x^\dagger &= \frac{x}{L}, & t^\dagger &= \frac{t a_\infty}{L}, & u^\dagger &= \frac{u}{a_\infty} \\
\rho^\dagger &= \frac{\rho}{\rho_\infty}, & p^\dagger &= \frac{p}{\rho_\infty a_\infty^2} = \frac{p}{\gamma p_\infty}, & T^\dagger &= \frac{T R_\infty}{a_\infty^2} = \frac{T}{\gamma T_\infty} \\
e^\dagger &= \frac{e}{a_\infty^2}, & c_p^\dagger &= \frac{c_p}{R_\infty} = \frac{\gamma}{\gamma - 1}, & c_v^\dagger &= \frac{c_v}{R_\infty} = \frac{1}{\gamma - 1}
\end{aligned} \tag{3.12}$$

By normalizing the flow variables in this way, the ideal gas equation and the speed of sound in an ideal gas remain the same and are given by

$$p^\dagger = (\gamma - 1)\rho^\dagger e^\dagger, \qquad a^\dagger = \sqrt{\frac{\gamma p^\dagger}{\rho^\dagger}}, \qquad e^\dagger = c_v^\dagger T^\dagger \tag{3.13}$$

Any reference to these flow variables will assume the normalized form and the normalization notation $(.)^\dagger$ will be abandoned throughout the following.

3.4 Time Evolution

Determining the timestep when solving the Euler equations, the Courant-Friedrichs-Lewy (CFL) criteria is usually employed. Due to the effects of artificial viscosity, a more restrictive formulation is employed to ensure stability. Using the same method as [4], the time-step is determined by

$$\Delta t \sim \frac{1}{\lambda_{max}\frac{N^2}{h} + \|\nu\|_{L^\infty}\frac{N^4}{h^2}} \tag{3.14}$$

where λ_{max} refers to the largest global characteristic velocity, ν is the artificial viscosity, and h is the smallest local cell size. The determination of the timestep using Equation 3.14 was found to be overly restrictive for some of the test cases due to the sensor never applying the maximum artificial viscosity throughout the solution. In order to ensure continuity in the sensor comparison, Equation 3.14 was used for every method regardless.

The simple forward Euler time stepping is only first-order accurate in time and is given by

$$u^{n+1} = u^n + \Delta t\ \mathcal{R}(u^n) \tag{3.15}$$

where $\mathcal{R}$ is given by

$$\frac{\mathrm{d}u}{\mathrm{d}t} = \mathcal{R}(u) \tag{3.16}$$

Since the DG method focuses on obtaining higher-order spatial accuracy, the time integration should also employ a method that obtains higher-order accuracy. Because discontinuities

are being modeled with the Euler equations in this work, it is important that the temporal scheme does not also admit spurious oscillations into the solution. The current work employs a three-stage, third-order Runge-Kutta (RK) method that maintains the TVD property [12]. The TVD property ensures monotonic solutions, thus eliminating the spurious oscillations caused by the time evolution. The 3^{rd}-order TVD-RK method is given by

$$\begin{aligned} u^{(1)} &= u^n + \Delta t\ \mathcal{R}(u^{(n)}) \\ u^{(2)} &= \left(\frac{3}{4}\right) u^n + \left(\frac{1}{4}\right) u^{(1)} + \left(\frac{1}{4}\right) \Delta t\ \mathcal{R}(u^{(1)}) \\ u^{n+1} &= \left(\frac{1}{3}\right) u^n + \left(\frac{2}{3}\right) u^{(2)} + \left(\frac{2}{3}\right) \Delta t\ \mathcal{R}(u^{(2)}) \end{aligned} \tag{3.17}$$

3.5 Numerical Flux Function : AUSM$^+$

The upwind numerical inviscid flux function, AUSM$^+$ (Advection Upstream Splitting Method +), developed by Liou in [13] was used in the present work. The method is based on the idea of the inviscid flux consisting of two physically distinct parts: the convective fluxes and the pressure fluxes. The convective fluxes are associated with the advection speed (flow speed) while the pressure fluxes are associated with the acoustic speed. The convective and pressure fluxes are formulated using the eigenvalues of the flux Jacobian matrices. The resulting AUSM$^+$ scheme has proven to be highly robust and accurate for varying types of fluid dynamics problems. The decreased computational cost, along with the approved accuracy, when compared to other flux-vector or flux-difference splitting methods was the driving force to implement the method in the present work.

The Euler equations given by

$$\frac{\partial U}{\partial t} + \frac{\partial}{\partial x} F(U) + \frac{\partial}{\partial y} G(U) = 0 \tag{3.18}$$

consist of the state vector of conservative variables, U, along with the components of the inviscid flux terms, $F(U)$ and $G(U)$. The numerical fluxes through each element edge can

be written as combinations of convective and pressure terms

$$\hat{\mathbf{f}} = n_x F(U) + n_y G(U) = V_n \begin{Bmatrix} \rho \\ \rho u \\ \rho v \\ \rho h_0 \end{Bmatrix} + p \begin{Bmatrix} 0 \\ n_x \\ n_y \\ 0 \end{Bmatrix} = \hat{\mathbf{f}}^c + \hat{\mathbf{p}} \begin{Bmatrix} 0 \\ n_x \\ n_y \\ 0 \end{Bmatrix} \tag{3.19}$$

where $V_n = un_x + vn_y$ is the convective velocity normal to the corresponding element interfaces and h_0 is the total enthalpy given by $h_0 = e_0 + p/\rho$. Examining Equation 3.19, the numerical flux, $\hat{\mathbf{f}}$, has been decomposed to the sum of the numerical convective flux, $\hat{\mathbf{f}}^{\mathbf{c}}$, and the numerical pressure flux, $\hat{\mathbf{p}}$.

The formulation begins with the computation of the average element interface speed of sound, with the left and right side of the interface denoted by L and R respectively.

$$a_{L,R} = \frac{a_L + a_R}{2} \tag{3.20}$$

The interface speed of sound is found by

$$a_{face} = \min\left(\frac{a_L^2}{c_L^*}, \frac{a_R^2}{c_R^*}\right) \tag{3.21}$$

where the variable c^* is computed using the corresponding left and right states, $c^* = \max\left(a, |(n_x u + n_y u)|\right)$. The speed of sound, a, for the left and right faces is also found by using the corresponding sides by

$$a = \sqrt{\frac{2(\gamma - 1)}{(\gamma + 1)h_0}} \tag{3.22}$$

The Mach number in the left and right elements are then found by

$$M_L = \frac{V_{n,L}}{a_{L,R}}, \qquad M_R = \frac{V_{n,R}}{a_{L,R}} \tag{3.23}$$

where the value $a_{L,R}$ is the average of the speed of sound. The Mach number and pressure splittings can then be found by

$$\begin{aligned}
\mathscr{M}^{\pm}(M) &= \begin{cases} \frac{1}{2}(M \pm |M|, & \text{if } |M| \geq 1, \\ \pm\frac{1}{4}(M \pm 1)^2 \pm \beta(M^2-1)^2, & \text{otherwise} \end{cases} \\
\mathscr{P}^{\pm}(M) &= \begin{cases} \frac{1}{2}(1 \pm \text{sign}(M)), & \text{if } |M| \geq 1, \\ \frac{1}{4}(M \pm 1)^2(2 \mp M) \pm \alpha M(M^2-1)^2, & \text{otherwise} \end{cases}
\end{aligned} \tag{3.24}$$

where α and β are two constants with bounds such that ($-3/4 \leq \alpha \leq 3/16$) and ($-1/16 \leq \beta \leq 1/2$). All of the test cases presented use the values $(\alpha, \beta) = (3/16, 1/8)$. Liou [13] stated that these values show an improvement over the original AUSM method and are comparable to the Roe flux splitting scheme. The interface values are then found by

$$\begin{aligned}
M_{face} &= \mathscr{M}^{+} + \mathscr{M}^{-} \\
p_{face} &= (\mathscr{P}^{+} p_L) + (\mathscr{P}^{-} p_R)
\end{aligned} \tag{3.25}$$

The final numerical flux is then computed by

$$\begin{aligned}
\hat{\mathbf{f}} = a_{face} \Bigg(& \frac{1}{2}(M_{face} + |M_{face}|) \begin{Bmatrix} \rho \\ \rho u \\ \rho v \\ \rho h_0 \end{Bmatrix}_L + \frac{1}{2}(M_{face} + |M_{face}|) \begin{Bmatrix} \rho \\ \rho u \\ \rho v \\ \rho h_0 \end{Bmatrix}_R \Bigg) \\
& \ldots + p_{face} \begin{Bmatrix} 0 \\ n_x \\ n_y \\ 0 \end{Bmatrix}
\end{aligned} \tag{3.26}$$

3.6 Artificial Viscosity Sensors

3.6.1 Modal Decay Sensor

The following formulation follows the work presented by Klockner, Warburton, and Hesthaven in [1]. The idea behind their work builds upon a numerical analysis of the modal coefficients of the approximate solution in each element. Since this work focuses on the approximate solution of the Euler equations, the nodal values of density are converted to their modal coefficient form, $\{\hat{q}_n\}_{n=0}^{N_p-1}$, through the Vandermonde matrix relationship given in Equation 2.24. This requires a matrix multiplication of order $[N_p, N_p]$ to convert the nodal values into the respective modal coefficients. The conversion of the coefficients is carried out at each Runge-Kutta iteration. It is worth noting that other flow variables, such as Mach number, entropy, or the residual of the conservation equations may be used as inputs into the modal decay sensor. Given the different test cases analyzed, the density was found to be the most reliable flow variable to base the sensor on.

The method focuses on the modal decay of the approximate solution in each element. Modal decay is the shrinking of modal coefficient magnitudes, $|\hat{q}_n|$, as the modal number, n, increases and is given by

$$|\hat{q}_n| \sim cn^{-s} \tag{3.27}$$

and taking the logarithm of the relationship yields

$$\log|\hat{q}_n| \sim \log(c) - s\log(n) \tag{3.28}$$

The coefficients, s and $\log(c)$, can then be found by solving the overdetermined system of equations of the form $Ax = b$ where

$$A = \begin{Bmatrix} 1 & -\log(1) \\ \vdots & \vdots \\ 1 & -\log(N) \\ 1 & -\log(N+1) \end{Bmatrix}, \qquad x = \begin{Bmatrix} \log(c) \\ s \end{Bmatrix}, \tag{3.29}$$

The coefficients then satisfy

$$\min \left(\sum_{n=1}^{N_p-1} |\log|\hat{q}_n| - (\log(c) - s\log(n))|^2 \right) \tag{3.30}$$

Two methods are applied to the initial modal coefficients prior to the solution of Equation 3.27. The methods, skyline pessimization and baseline decay, counteract certain occurrences that may be encountered by using the raw modal coefficients acquired by the numerical approximation. The baseline decay procedure is first applied to the initial modal coefficients, $\hat{q}_n$, resulting in the new coefficients, $\tilde{q}_n$. The skyline pessimization procedure is then applied to $\tilde{q}_n$ with the final resulting modal coefficients, $\bar{q}_n$, becoming the inputs into Equation 3.27.

In the instance that the modal coefficients contain noise that is small compared to the magnitude of the modal coefficient, the method of baseline decay is applied to the modal coefficients. Baseline decay adjusts for this by adding a "sense of scale" by distributing energy according to a "perfect modal decay", which is given by

$$|\hat{b}_n| \sim \frac{1}{\sqrt{\sum_{i=1}^{N_p-1} \frac{1}{i^{2N}}}} \frac{1}{n^N} \tag{3.31}$$

where N is the polynomial degree of the modal expansion. The new input to the skyline pessimization simply controls coefficients that are relatively small compared to the element-wise norm of the approximation and takes the form

$$|\tilde{q}_n|^2 \doteq |\hat{q}_n|^2 + \| q_n \|^2_{L^2(D_k)} |\hat{b}_n|^2 \qquad \text{for } n \in \{1, \ldots, N_p - 1\} \tag{3.32}$$

The approximation of the modal decay, represented by Equation 3.27, is only capable of generating monotone modal decay approximations. Skyline pessimization is used to ensure that the modal decay is indeed exhibiting a monotone behavior. The method is formulated by considering two modes, n and m, where $m > n$. If $|\tilde{q}_m| \gg |\tilde{q}_n|$, then the small coefficient $|\tilde{q}_n|$ was likely spurious and can be replaced. The method ensures monotone decay by raising each modal coefficient up to the largest higher-numbered modal coefficient. The resulting procedure for skyline pessimization is then given by

$$\bar{q}_n \doteq \max_{i \in \{\min(n, N_p-2), \ldots, N_p-1\}} |\tilde{q}_i| \qquad \text{for } n \in \{1, 2, \ldots, N_p - 1\} \tag{3.33}$$

In the case where only the first few modes are used a final correction must be made to counter odd-even effects of the modal data. This is accomplished by ensuring that the last modal coefficient is larger than the second-to-last modal coefficient.

Now that the modal decay can be approximated by Equation 3.27, the artificial viscosity can be computed using the estimated decay exponent, s, in an element-wise manner. The coefficient, s, was scaled above such that $s = 1$ corresponds to a discontinuous solution, $s = 2$ corresponds to a C^0 solution, $s = 3$ corresponds to a C^1 solution, and so forth. The artificial viscosity is then determined using the formulation

$$\nu(s) = \nu_0 \begin{cases} 1 & s \in (-\infty, 1), \\ \frac{1}{2}(1 + \sin(-(s-2)\frac{\pi}{2})) & s \in [1, 3], \\ 0 & s \in (3, \infty). \end{cases} \tag{3.34}$$

The work by Klockner[1] references Barter and Darmofal [14] as also defining the term, ν_0 as

$$\nu_0 = \frac{\sigma^2}{2t} = \lambda \frac{h}{N} \tag{3.35}$$

where h is the local element size and N the approximating polynomial. A reference time scale is given by $t = (N/2)\Delta t$, and the final two variables to determine the maximum artificial viscosity are defined as $\sigma = h/N$ and $\Delta t \approx h/(\lambda N^2)$. The present work deviates slightly in the computation of the maximum viscosity. Previous work had shown a tendency for the maximum artificial viscosity computed by Equation 3.35 to be overly dissipative. The maximum artificial viscosity will be determined using the entropy residual form of the ν_0 presented in the next section. Using the same maximum artificial viscosity enables a clear comparison to be drawn between the two methods.

3.6.2 Entropy Residual Sensor

The entropy residual sensor is taken from the work of Guermond, Zinghan, and Pasquetti [2, 15, 16]. The premise behind the entropy production residual is based on the Second Law of Thermodynamics which states that the entropy of a system must be nondecreasing with time. The method monitors $\mathcal{R}$ for negative values which would indicate a violation of physics. In the computational approximation, this condition would be violated when a feature of the flow is under-resolved. In the case of a shock wave, the properties of the flow are undergoing significant changes across an area on the order of mean-free paths. This is orders of magnitude smaller than the flowfield domain discretization. The resulting under-resolution results in a negative entropy production across the shock wave. The result is a discontinuity sensor that detects discontinuities and sharp gradients based on the physical characteristics of the flow.

The entropy residual sensor is incorporated as a state variable along with the Euler equation state vector for the system of conservation laws. The addition of the entropy residual to the state vector, U, and the inviscid flux vectors, $F(U)$ and $G(U)$, is expressed

as

$$U = \begin{Bmatrix} \rho \\ \rho u \\ \rho v \\ \rho e_0 \\ \rho s \end{Bmatrix} \qquad F(U) = \begin{Bmatrix} \rho u \\ \rho u^2 + p \\ \rho v u \\ (\rho e_0 + p)u \\ u\rho s \end{Bmatrix} \qquad G(U) = \begin{Bmatrix} \rho v \\ \rho u v \\ \rho v^2 + p \\ (\rho e_0 + p)v \\ v\rho s \end{Bmatrix} \tag{3.36}$$

where the entropy, s, is given by

$$s = \frac{1}{\gamma - 1} \ln \left(\frac{p}{\rho^\gamma} \right) \tag{3.37}$$

The residual of the entropy production equation is then given in two-dimensions by

$$\mathcal{R} = \frac{\partial}{\partial t}(\rho s) + \frac{\partial}{\partial x}(u\rho s) + \frac{\partial}{\partial y}(v\rho s) \geq 0 \tag{3.38}$$

Using a nonlinear transformation of the conservative variables, the time derivative in Equation 3.38 can be expressed in terms of the spatial derivatives. The transformation begins by using the chain rule on the time derviative such that

$$\begin{aligned} \frac{\partial(\rho s)}{\partial t} &= \frac{\partial(\rho s)}{\partial U}\frac{\partial U}{\partial t} \\ &= \frac{\partial(\rho s)}{\partial V}\left(\frac{\partial U}{\partial V}\right)^{-1}\frac{\partial U}{\partial t} \\ &= LM^{-1}\frac{\partial U}{\partial t} \\ &= W\frac{\partial U}{\partial t} \\ &= -W\frac{\partial}{\partial x}F(U) - W\frac{\partial}{\partial y}G(U) \end{aligned} \tag{3.39}$$

The variables present in Equation 3.39 are the vector of conservative variables, U, along with the vector of primitive variables, V, given by

$$U = \begin{Bmatrix} \rho \\ \rho u \\ \rho v \\ \rho e_0 \end{Bmatrix}, \qquad V = \begin{Bmatrix} \rho \\ u \\ v \\ p \end{Bmatrix} \tag{3.40}$$

the transformation matrix, M, given by

$$M = \frac{\partial U}{\partial V} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ u & \rho & 0 & 0 \\ v & 0 & \rho & 0 \\ k & \rho u & \rho v & 1/(\gamma - 1), \end{bmatrix}, \tag{3.41}$$

along with the inverse

$$M^{-1} = \frac{\partial V}{\partial U} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -u/\rho & 1/\rho & 0 & 0 \\ -v/\rho & 0 & 1/\rho & 0 \\ k \cdot (\gamma - 1) & -u \cdot (\gamma - 1) & -v \cdot (\gamma - 1) & (\gamma - 1), \end{bmatrix} \tag{3.42}$$

and the resulting entropy variables [17], W, given by

$$W = \left[s + \frac{\rho k}{p} - \frac{\gamma}{\gamma - 1}, \quad -\frac{\rho u}{p}, \quad -\frac{\rho v}{p}, \quad \frac{\rho}{p} \right] \tag{3.43}$$

where k is the kinetic energy given by $k = {}^1/_2(u^2 + v^2)$. The entropy residual is then formulated using only the spatial derivative terms by substituting the transformation from

Equation 3.39 into the initial Equation 3.38. This results in

$$\mathcal{R} = \frac{\partial}{\partial x}(u\rho s) + \frac{\partial}{\partial y}(v\rho s) - W\frac{\partial}{\partial x}F(U) - W\frac{\partial}{\partial y}G(U) \geq 0 \tag{3.44}$$

where $F(U)$ and $G(U)$ are the original Euler inviscid fluxes. After the entropy residual is determined, the state vector value ρs is then recomputed from the updated state and the next Runge-Kutta step is performed.

Once the residual, $\mathcal{R}$, of the transformed entropy production equation is determined using Equation 3.44, the artificial viscosity can then be determined. The maximum artificial viscosity is determined by setting a maximum artificial viscosity limit, ν_0, found by

$$\nu_0 = c_{max}\frac{h\lambda}{N^2} \tag{3.45}$$

where c_{max} is a constant user-defined parameter dependent on the test case. The value of ν_0 computed in Equation 3.45 was also used as the maximum artificial viscosity for the modal decay sensor. The additional factor of N in the denominator, when compared to Equation 3.35, not only seemed to benefit the modal sensor, but allowed an even comparison between the two artificial viscosity methods.

The entropy sensor proceeds by defining a viscosity term, ν_s, in each element that is determined by using the residual values, $\mathcal{R}$, that have been developed at each nodal location. The formulation of ν_s for each element is given by

$$\nu_s = c_s\frac{h^2}{N^2}\max\left(0, \frac{-\mathcal{R}}{|\Delta\rho s|_{ref}}\right) \tag{3.46}$$

where $|\Delta\rho s|_{ref}$ is a normalizing coefficient ensuring that ν_s has the proper viscosity units and c_s is a user-defined, problem specific constant. The final piecewise constant value of

artificial viscosity in each element is then found through

$$\nu_{reg} = \min\left(\nu_s, \nu_0\right) \tag{3.47}$$

The entropy sensor, while based in the physics of the flow problem, has the unfortunate property of containing the two user-defined parameters, c_{max} and c_s, in order to scale the artificial viscosity. This is opposed to the modal sensor that is a parameter free method since ν_0 is determined through known flow variables. The parameters are only dependent on the temporal and spatial integration methods and are independent of the timestep and mesh size [16]. Guermond has suggested a simple method of finding the parameters by using a coarse grid that should enable a quick determination of the values. Through his research, Guermond suggests the parameters typically fall in the range of $c_{max} \in [0.15, 0.5]$ and $c_s \in [0.1, 1]$ while also neglecting the normalizing term $|\Delta\rho s|_{ref}$.

In the present work a slightly different approach was taken in regards to the user-defined parameters. Through experiments the constant, c_{max}, was found to vary between 1 and 4 depending on the type of test case. The present work used $c_{max} = 1$ as a basis for all test cases, but results were included for $c_{max} = 4$ for the shock-vortex interaction as a comparison. It was found that the c_s term could be set to one such that the scaling factor could be incorporated into the normalizing coefficient $|\Delta\rho s|_{ref}$.

3.7 Computational Efficiency

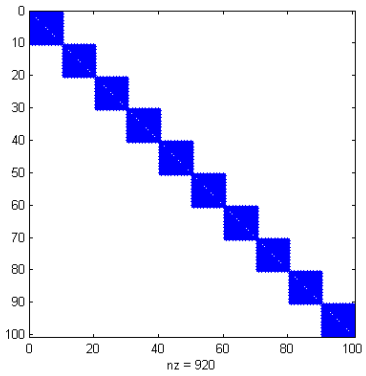
The interest in higher-order methods stems from the ability to achieve higher accuracy at the cost of increased work for each degree of freedom. If a given accuracy is required for a solution, a higher-order method can achieve the accuracy more efficiently while using less degrees of freedom than a lower-order method. This is due to the increased work required per degree of freedom when using a higher-order method. A lower-order method must increase

the number of approximating discretized elements to achieve the same order of accuracy. Increasing the number of elements, results in an increase in degrees of freedom.

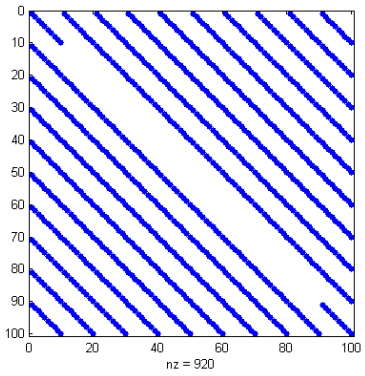
3.7.1 Sparse-Matrix Multiplication

Sparse-matrix multiplication was used in the present work to improve computational efficiency. The stiffness matrices, S_x and S_y, are of the order $[(N+1)^2, (N+1)^2]$ with $(N+1)(N+1)^2$ nonzero entries. The sparse layout of the stiffness matrices are shown in Figure 3.1. The lifting matrices, L_x and L_y, are of the order $[(N+1)^2, (N+1), 2]$ with $(N+1)(N+1)$ nonzero entries for each of the third dimensions. The sparsity of the lifting matrices can be seen in Figure 3.2.

The sparse matrix multiplication routine was used in the main DG scheme when computing the local solution in each cell with the stiffness matrix. The routine was also employed when the results of the numerical flux function at the element interfaces is "lifted" through the local solution in each element. The wallclock time savings that the sparse matrix multiplication returned scaled directly with the choice of the approximating polynomial order used in the DG method. To gauge the effectiveness of the sparse matrix multiplication routine, the standard isentropic vortex test case, presented later, showed a 25% reduction in wall-clock time for a 5^{th}-order polynomial, a 34% reduction for a 7^{th}-order polynomial, and a 45% reduction for a 9^{th}-order polynomial.

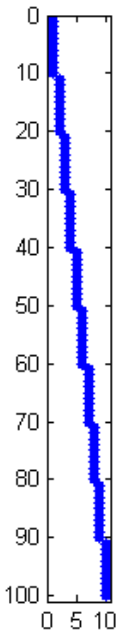


(a) Sparse X-Stiffness Matrix

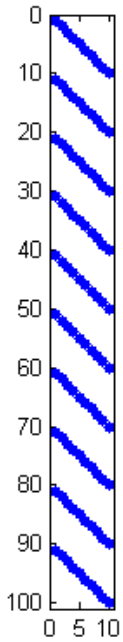


(b) Sparse Y-Stiffness Matrix

Figure 3.1: Sparse Stiffness Matrices for N = 9



(a) Sparse X-Lifting Matrix



(b) Sparse Y-Lifting Matrix

Figure 3.2: Sparse Lifting Matrices for N = 9

3.7.2 Open-MP

The discontinuous nature of the DG method promotes a computational environment that easily incorporates a parallelization of the computational scheme. The local solutions computed in each element are independent of adjacent elements allowing the solutions to

be computed independently and thus each element may be handled by a single processor. After the local solutions are computed, the global solution is computed with each element's edge data by use of a numerical flux function. The numerical flux function can then be computed in a parallel setting because each element interface is independently solved using the local element solutions that were previously solved. The present numerical code, being developed in Fortran 90, utilized the Open-MP software to parallelize the code. The Open-MP functionality was simply adapted by adding command lines into the base code.

To highlight the time-savings, the run-times are presented in Figure 3.3. The values are given as the percentage of time normalized by the serial case (1 processor used). Using the results, it was found that a better efficiency of running two cases using three or four processors each became more efficient than using all eight processors on a single case run.

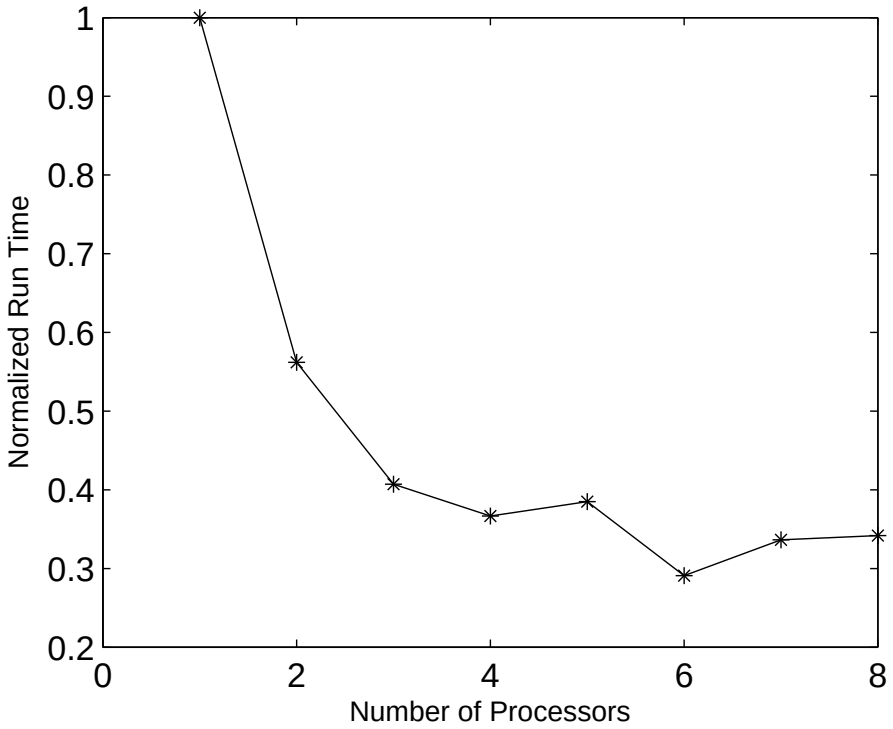


Figure 3.3: Decrease in Wallclock Time Due to Parallelization

Chapter 4

Test Cases

4.1 Isentropic Vortex and Kelvin Helmholtz Shear Layer

4.1.1 Isentropic Vortex

The isentropic vortex is a two-dimensional test case where velocity perturbations are added into a uniform flowfield. The initial vortex is convected through the flowfield by the mean velocity field without any change to the vortex's size or strength. Due to the absence of any diffusive mechanisms present in the test case, the isentropic vortex presents a good measurement of the amount of inherent artificial viscosity that occurs in the numerical discretization method being used. The freestream values are determined by

$$u_\infty = M_\infty \cos(\alpha), \quad v_\infty = M_\infty \sin(\alpha), \quad p_\infty = 1/\gamma, \quad T_\infty = 1/\gamma \tag{4.1}$$

where α is the flow angle relative to the x-axis. The test case was performed using $M_\infty = 0.5$ and $\alpha = 0.0$. The above relations result in a density of $\rho_\infty = 1$ and an acoustic speed $a_\infty = 1$. The vortex is formed by the addition of perturbations defined by

$$\Delta u = -\Delta y f, \quad \Delta v = \Delta x f, \quad \Delta T = -\frac{\gamma - 1}{2\gamma} f^2 \tag{4.2}$$

The vortex potential, f is defined by

$$f = \frac{B}{2\pi} \exp A(1 - r^2), \quad r^2 = \Delta x^2 + \Delta y^2 \tag{4.3}$$

The parameters A and B are referred to as the vortex strength and gradient, respectively, and the vortex Mach number can be determined from those parameters by $M_v = B/2\pi$. The parameters used here are $A = 0.5$ and $B = 5.0$. The pressure may then be computed using the isentropic relation

$$p = p_\infty \left(\frac{T}{T_\infty}\right)^{\frac{\gamma}{\gamma-1}} = \frac{1}{\gamma}\left(\gamma T\right)^{\frac{\gamma}{\gamma-1}} \tag{4.4}$$

The domain is given by $x, y \in [-5, 5]$ x $[-5, 5]$ with doubly-periodic boundary conditions in the x and y directions. The final time was computed such that the vortex would return to the initial starting location after traversing the domain. This corresponded to a $t_{final} = 20.0$.

4.1.2 Kelvin-Helmholtz Shear Layer

The Kelvin-Helmholtz shear layer is a two-dimensional test case with periodic boundary conditions in both the spatial directions. The domain has dimensions in x,y of $[0, 2\pi]$ x $[0, 2\pi]$. Initial ambient density and pressure values are $\rho = 1$ and $p = 1/\gamma$. An instability appears in the flowfield due to a free shear layer that is caused by a velocity gradient between two flows. The flow domain consists of an horizontal inner jet moving through an ambient region. The initial horizontal and vertical velocity conditions are given by

$$\begin{aligned} s &= \begin{cases} M\tanh\left(\frac{3\pi-2y}{2\epsilon}\right) & \text{for } y > \pi \\ M\tanh\left(\frac{2y-\pi}{2\epsilon}\right) & \text{otherwise} \end{cases} \\ u &= \frac{1}{2}(s+1)M \\ v &= M\delta\left(\sin\left(\frac{x}{3}\right) - \sin\left(\frac{2x}{3}\right) + \sin\left(\frac{3x}{3}\right)\right) \end{aligned} \tag{4.5}$$

The flow parameters used in this work are $\epsilon = \pi/15$, $\delta = 0.1/4.9$, and a jet Mach number of $M = 0.25$. In order to initialize the instabilities, vertical velocity perturbations are seeded into the initial flowfield. The perturbations are given by

$$v = M\delta\sin(x) \tag{4.6}$$

The Kelvin-Helmholtz test case presents an example of a shock free flow that will still develop numerical instabilities due to the sharp gradients that occur as a result of the shear layer. As the solution progresses, the shear layer becomes thinner and thinner against a fixed resolution. This enables a comparison of the two sensors as to the onset and application of the artificial viscosity. In order to examine the effects of each sensor over long time integrations, a final time of $t = 300$ was used. The artificial viscosity parameter, $|\Delta\rho s|_{ref}$, was found experimentally to be 0.01.

4.2 Sod Shock Tube

The Sod shock tube is a one-dimensional test case in which a gas at two different states is initially held separate from each other. At $t = 0$, the initially stagnant gases are allowed to interact resulting in an unsteady flow. The flowfield exhibits a shock wave that moves into the low-pressure state, a rarefaction wave that expands into the high-pressure state, and a contact discontinuity that separates the two regions. The exact solution for the test case is found iteratively using the method explained by Toro in [18]. The initial conditions for the Sod shock tube problem, where $x \in [-1, 1]$, are

$$\begin{cases} \rho = 1.0, & u = 0.0, & p = 1.0, & \text{if } x < 0 \\ \rho = 0.125, & u = 0.0, & p = 0.1, & \text{if } x > 0 \end{cases} \tag{4.7}$$

The computation is carried out to a final time of 0.4. The entropy sensor parameter, $|\Delta\rho s|_{ref}$, is taken as 0.095 which corresponded to the middle value in between the minimum and maximum initial values of ρs.

The test case is well suited for examining unsteady compressible computational fluid dynamics solvers. The shock tube case enables a comparison of each sensor's ability to separate between the contact discontinuity and shock wave. The presence of the expansion region also allows a comparison of each artificial viscosity method and their ability to not unduly diffuse the important flow features. With the resulting flowfield exhibiting many flow

features, each sensor is tested for the ability to separate and properly apply diffusion where needed.

4.3 Shu-Osher Planar Problem

The one-dimensional Shu-Osher planar wave test case follows the work by Shu and Osher in [19]. The present test case exhibits a Mach 3 shock wave propagating across the domain from left to right. The upstream and downstream boundary conditions are both fixed by the initial conditions. Small scale features originate as the shock encounters a sinusoidal density field. As the shock passes through the density field, high frequency oscillations occur in the immediate vicinity behind the shock wave until oscillations with the same frequency as the initial sinusoidal profile occur. The small scale features that occur behind the shock provide insight of each artificial viscosity sensor ability to not unduly diffuse these structures. The Shu-Osher test case presents the ability of the artificial viscosity sensors to apply diffusion in the region of the shock wave while not over diffusing the resulting small-scale features that result around the shock wave.

The domain is given as $x \in [0, 10]$. The initial condition are given as

$$\begin{cases} \rho = 3.857143, & u = 2.629367, & p = 10.333333, & \text{if } x < 1 \\ \rho = 1 + 0.2\sin(5x), & u = 0.0, & p = 1.0, & \text{if } x > 1 \end{cases} \tag{4.8}$$

The simulation is carried out to a simulation time of t = 1.8. The entropy sensor parameter, $|\Delta\rho s|_{ref}$, is taken as 0.5 and corresponds to the middle value of the minimum and maximum ρs initial values.

4.4 Shock-Vortex Interaction

The shock-vortex interaction test case follows Case G presented in the work by Inoue and Hattori in [20]. The test case is a crude model for sound generation in compressible turbulent flows. The interaction shows an acoustic wave that is composed of four alternating compression and rarefaction regions [20]. The test case occupies the domain $x, y \in [-10, 20]x[-15, 15]$ with a stationary vortex located at $(x, y) = (5, 0)$ and a moving shock starting at $(x, y) = (-5, 0)$ and moving downstream to the right. The Mach numbers of the shock and vortex are $M_s = 1.29$ and $M_v = 0.39$. The flow parameters used for the vortex in the present case, which were defined in the isentropic vortex case, are $A = 0.5$ and $B = 2.45044$. The perturbations that define the vortex are added to the initial conditions downstream of the shock wave. The flow parameters downstream of the normal shock wave are given by

$$\begin{aligned} p_L &= p_R \left(1 + \frac{2\gamma}{\gamma+1}(M_s^2 - 1)\right) \\ \rho_L &= \rho_R \left(1 + \frac{\gamma+1}{\gamma-1}\frac{p_L}{p_R}\right)\left(\frac{\gamma+1}{\gamma-1} + \frac{p_L}{p_R}\right)^{-1} \\ u_L &= V_s \left(1 - \frac{\rho_R}{\rho_L}\right) \end{aligned} \tag{4.9}$$

with the upstream conditions given by

$$p_R = 1, \qquad \rho_R = \frac{1}{\gamma}, \qquad u_R = 0 \tag{4.10}$$

The shock is moving into the stagnant region with a speed

$$V_s = M_s \sqrt{\frac{\gamma p_R}{\rho_R}} \tag{4.11}$$

The upstream and downstream boundary conditions at $x = -10, 20$ are fixed while the upper and lower boundaries at $y = \pm 15$ are periodic. The simulation is run out until a final

time of $t_{final} = 16.20$. The entropy sensor parameter, $|\Delta\rho s|_{ref}$, was determined through experimentation and is taken as 0.5. While the choice used here was less dissipative and allowed oscillations to form on the element interface locations it did not overly diffuse a resulting double shock inside the vortex and was run at this value to show this.

Chapter 5

Results

5.1 Isentropic Vortex and Kelvin Helmholtz Shear Layer

5.1.1 Isentropic Vortex

The test case for the isentropic vortex was performed for polynomial approximation orders of 3 through 9 with 100, 225, 400, 625, and 900 total number of elements. The degrees of freedom for each set of discretized parameters are given, for quadrilateral elements, by $DoF = (N+1)^2(NI \cdot NJ)$ and is presented in Table 5.1. Because the isentropic vortex case is simply a convected vortex with no dissipation present, using a final time of t = 20 convects the vortex back to the initial position when periodic boundary conditions are employed. Using the initial conditions, the L^2-errors are calculated for the resulting flowfield density values and are presented in Table 5.2. The bold face values are used to highlight the grid parameters that exhibit the same magnitude errors that are presented in Table 5.2. The computational wall-clock times are given in Table 5.3 for each run case. All of the test cases were run using 7 processors with Open-MP.

Analyzing the bold faced value at $N = 6$ and $N = 9$, the 9^{th}-order approximation maintains an L^2-error with the same order of magnitude as the 6^{th}-order approximation. The 9^{th}-order approximation does this with 67.3% fewer degrees of freedom and 72.5% less time than the 6^{th}-order approximation. Looking at the bold face cases of $N = 7$ and $N = 8$, the order of the error is still of the same magnitude, but the DG method does enjoy a slightly lower value. Comparing these two cases, the 8^{th}-order approximation results in a 28.8% decrease in degrees of freedom and a 16.8% decrease in runtime over the 7^{th}-order approximation. As can be determined from the above two analysis, using a higher-order

polynomial approximation will result in a decreasing amount of runtime to determine the solution. This is achieved due to the decreasing number of needed degrees of freedom to compute the same order of magnitude for the error values.

The exponential convergence of the DG method is seen in Figure 5.1 where the L^2-errors are plotted for the computed density field. Figure 5.1 presents the convergence rate for the L^2-errors of the computed density field. As the polynomial order is increased, the DG method enjoys an exponential convergence rate for the smoothly varying solution field.

	Degrees of Freedom for Quadrilateral Elements						
K	N=3	N=4	N=5	N=6	N=7	N=8	N=9
100	1600	2500	3600	4900	6400	8100	**10000**
225	3600	5625	8100	11025	14400	**18225**	22500
400	6400	10000	14400	19600	**25600**	32400	40000
625	10000	15625	22500	**30625**	40000	50625	62500
900	14400	22500	32400	44100	57600	72900	90000

Table 5.1: Degrees of Freedom for Grid Discretization

	L^2 Error for Density Values						
K	N=3	N=4	N=5	N=6	N=7	N=8	N=9
100	2.32E-01	4.84E-02	1.21E-02	3.07E-03	8.64E-04	2.21E-04	**4.69E-05**
225	5.91E-02	1.08E-02	2.16E-03	3.38E-04	8.16E-05	**1.06E-05**	2.01E-06
400	2.40E-02	3.99E-03	5.59E-04	8.2E-05	**1.37E-05**	1.30E-06	3.18E-07
625	1.22E-02	1.77E-03	1.83E-04	**2.39E-05**	2.96E-06	3.65E-07	1.83E-07
900	7.60E-03	9.25E-04	7.82E-05	8.91E-06	8.44E-07	8.83E-08	8.99E-09

Table 5.2: Density L^2 Error for Number of Elements, K, and Polynomial Order, N

	Wall-Clock Times (seconds)						
K	N=3	N=4	N=5	N=6	N=7	N=8	N=9
100	1.007	2.470	4.114	8.419	11.867	24.064	**39.592**
225	2.767	8.527	13.438	28.005	42.946	**82.471**	128.859
400	7.781	20.520	37.131	70.809	**99.169**	214.139	334.289
625	14.386	34.029	70.325	**143.711**	207.170	419.264	678.581
900	24.90	73.87	127.36	249.07	460.86	594.30	1138.67

Table 5.3: Wall-Clock for Number of Elements, K, and Polynomial Order, N

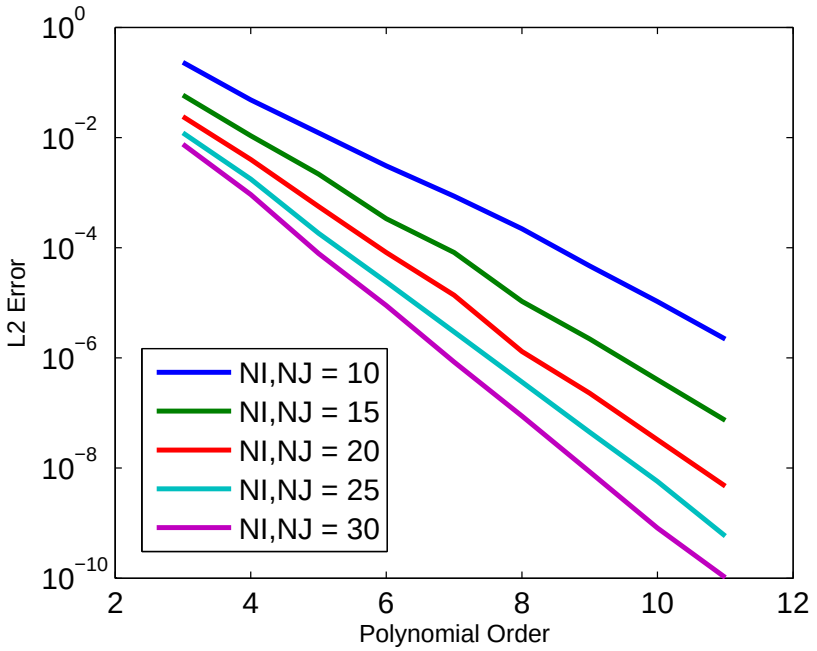


Figure 5.1: L^2 Density Errors for Approximating Polynomial Order

5.1.2 Kelvin-Helmholtz Shear Layer

The results presented for the Kelvin-Helmholtz test case were run using 27 elements in the i-direction (NI=27) and 18 elements in the j-direction (NJ = 18). The modal sensor took a total wall-clock time of 23938.839 seconds compared to 23346.527 seconds for the entropy

sensor. This equated to the modal sensor having a 2.5% longer wall-clock computation time compared to the entropy sensor, or a 9.87 minute savings from a 6.65 hour run. The modal sensor applied artificial viscosity to 53.2% fewer total elements over the simulation run than the entropy sensor, but did apply 32.5% more total artificial viscosity compared to the entropy sensor. The reasons for this can be seen through select timesteps of the simulation procedure.

Figure 5.2 shows the energy concentration at t = 61.004. A qualitative inspection shows that the solution utilizing the modal sensor exhibits more diffusive effects compared to the entropy sensor. The modal sensor shows a higher effect of diffusion at the edges of the shear layer and in the developing vortices. Contrary to what may be expected, Figure 5.3 shows the magnitude and element locations where artificial viscosity was being applied in the flow domain. The modal sensor applied a total amount of artificial viscosity equaled to 0.1106358 across the flowfield while the entropy sensor applied a total value of 0.1510923. This corresponds to a 36.57% increase of the total sum of artificial viscosity across the flowfield for the entropy sensor compared to the modal sensor at the current timestep.

Figure 5.4 presents the energy concentration at the final simulation time of $t = 300$. The energy concentration for the modal sensor shows a great deal of diffusive errors in comparison to the entropy sensor. Using the vortex in Figure 5.4 that resides in the region $(x, y) \approx (9, 3)$ as a comparison, the modal sensor, Figure 5.4a, has lost the sharp gradients surrounding the vortex and the features have been lost into one blurred or smeared structure. The solution computed with the entropy sensor, given in Figure 5.4b, still possesses sharp features of the shear layer. The total artificial viscosity applied at the final time step for the entropy sensor is 63.3% less compared to the modal sensor.

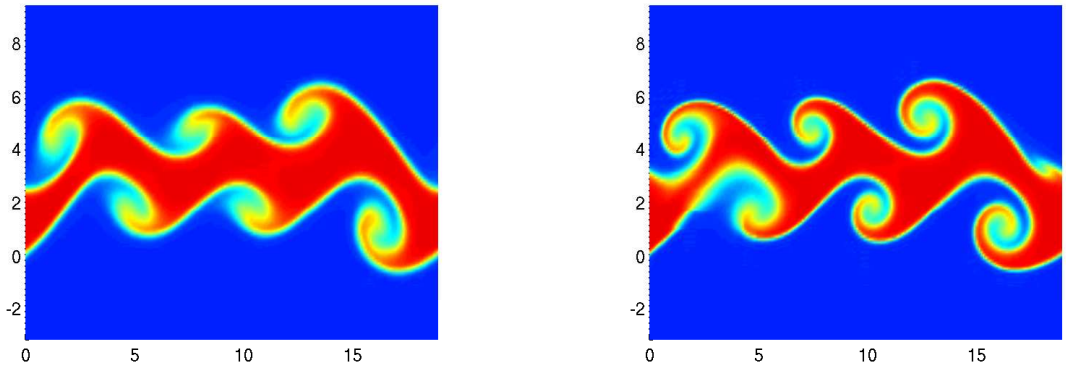
Figure 5.6 presents the total number of elements activated with artificial viscosity (Figure 5.6a) and the total amount of artificial viscosity applied throughout the flow domain (Figure 5.6b) for selected timesteps throughout the simulation. The modal sensor activated fewer elements per timestep with artificial viscosity for almost the entirety of the simulation.

Conversely, with the exception of the beginning of the simulation corresponding to the initial vortex formulation, the modal sensor applied more total artificial viscosity per timestep compared to the entropy sensor. Over the course of the entire simulation, the entropy sensor applied artificial viscosity to 53.18% more elements, while also applying 32.46% less total artificial viscosity. The entropy sensor applied artificial viscosity with a much smaller value, orders of magnitude, in majority of the elements than the modal sensor. To emphasize this point, returning to Figure 5.5 (t=300), the modal sensor is applying artificial viscosity to 331 total elements as opposed to the entropy sensor which is activating all 486 elements in the flow domain. Throughout the entire simulation the modal sensor never applied the maximum value of artificial viscosity to an element, whereas the entropy sensor applied the maximum amount of artificial viscosity, ν_0, to 158 elements throughout the simulation. Out of the 158 elements activated with ν_0, 150 of the elements were activated between $t = 20$ and $t = 70$ which corresponds to the initial vortex formations.

Figure 5.7 presents the percentage of the maximum artificial viscosity that was applied throughout the simulation. Due to the presence of sharp gradients as opposed to discontinuities such as shock waves, both sensors apply smaller fractions of the maximum artificial viscosity than the other test cases to be presented. The results further support the that the entropy sensor is applying a smaller amount of artificial viscosity throughout the simulation. The entropy sensor is sensitive to the resolution of the discretized domain, with small values of the entropy residual leading to small values of artificial viscosity being introduced into the approximation. It is also evident that the entropy sensor has significantly less elements that are not being activated compared to the modal sensor.

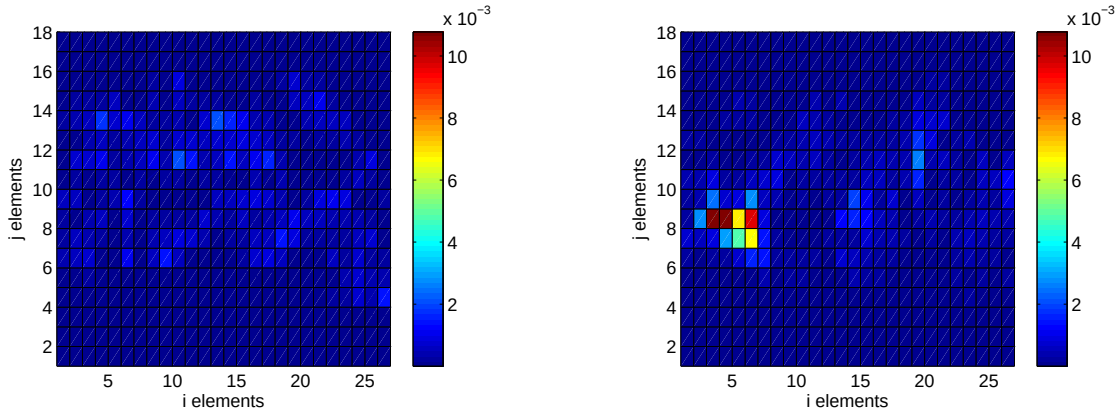
The modal sensor applied artificial viscosity throughout the flow simulation by applying a high, but not maximum, amount of artificial viscosity in the select regions once the spurious oscillations become large enough to warrant diffusion. The entropy sensor applied the maximum amount of artificial viscosity in the beginning of the flow simulation as the vortices started forming. As the solution progressed in time, the entropy sensor applied a

relatively small amount of diffusion in the elements once any sign of oscillations occurred. The threshold for activating artificial viscosity proved to be much lower for the entropy sensor when compared to the modal sensor.



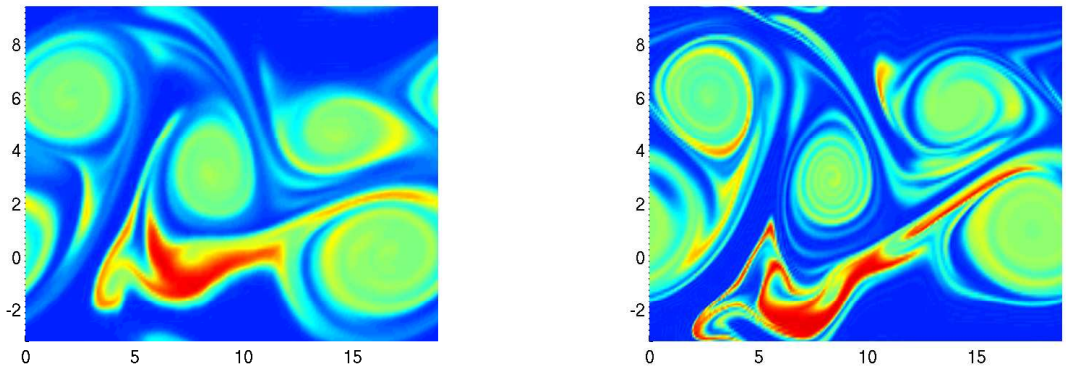
(a) Modal Decay Sensor (b) Entropy Residual Sensor

Figure 5.2: Kelvin-Helmholtz Energy Concentration at t = 61.0043



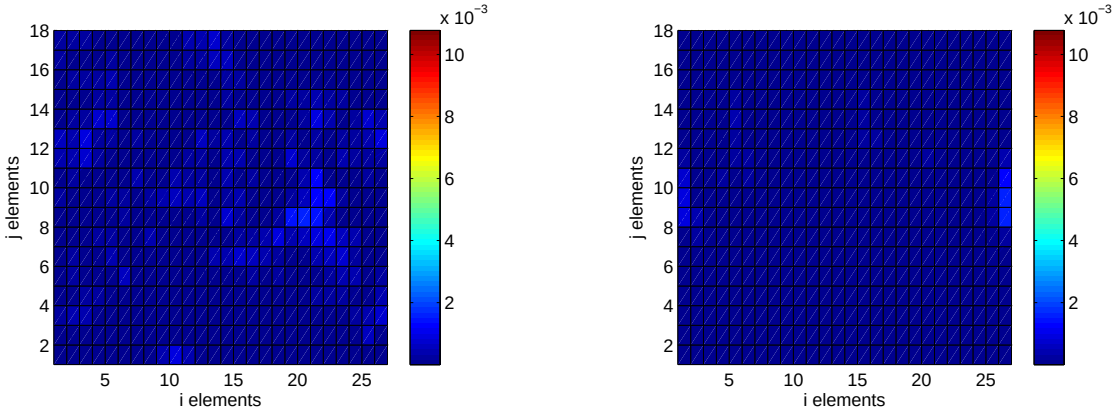
(a) Modal Decay Sensor (b) Entropy Residual Sensor

Figure 5.3: Kelvin-Helmholtz Artificial Viscosity at t = 61.0043



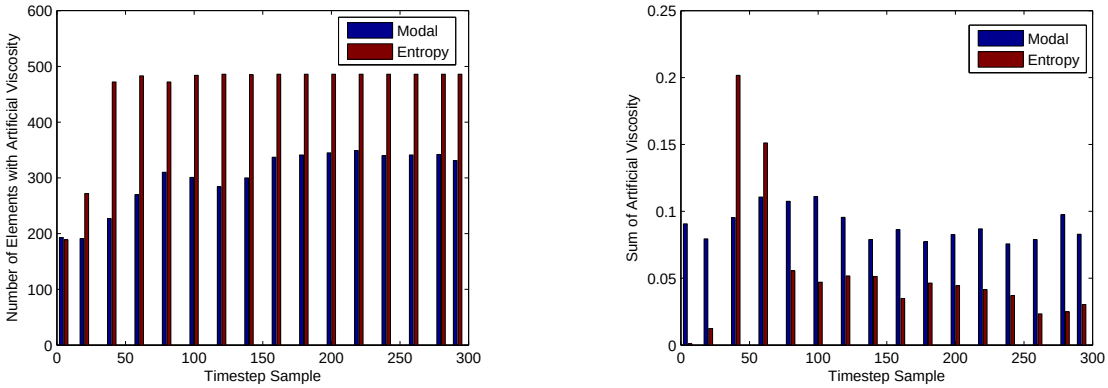
(a) Modal Decay Sensor (b) Entropy Residual Sensor

Figure 5.4: Kelvin-Helmholtz Energy Concentration at t = 300



(a) Modal Decay Sensor (b) Entropy Residual Sensor

Figure 5.5: Kelvin-Helmholtz Artificial Viscosity at t = 300



(a) Number of Elements with Artificial Viscosity (b) Total Sum of Artificial Viscosity

Figure 5.6: Kelvin-Helmholtz Stats for Artificial Viscosity Sensors

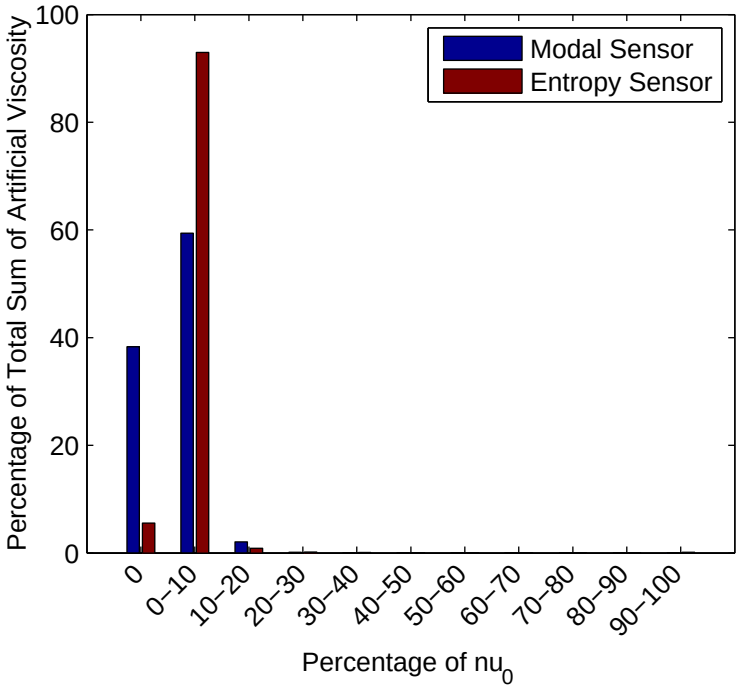


Figure 5.7: Percentage Distribution of Applied Artificial Viscosity for Sod Shock Tube

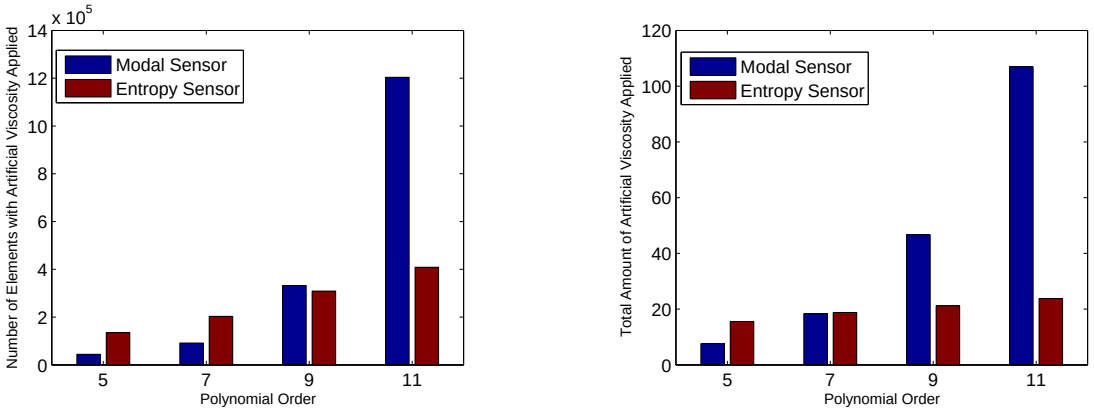
5.2 Sod Shock Tube

The Sod shock tube test case was run using four polynomial approximation orders of 5, 7, 9 and 11 for a flow domain consisting of 100 elements. The resulting L^2-errors for the density values are presented in Table 5.4. The modal sensor resulted in a higher accuracy solution for the given test cases. The results indicate that as the polynomial order increased, the percent difference of the L^2-errors between the two sensors decreased. The explanation behind this correlation can be deduced from the results given in Figure 5.8. As the order of the polynomial approximation increased, the number of elements activated with artificial viscosity and the total sum of the applied artificial viscosity for the modal sensor increased at a much higher rate than the entropy sensor. The modal sensor appears to be better suited for lower polynomial orders, whereas when the polynomial order increases, the entropy sensor appears to be the better choice. This sensor arrangement is also supported by

Table 5.5 which presents the percent difference of the entropy sensor when compared to the modal sensor. As the polynomial approximation increases, the entropy sensor increasingly becomes the more efficient scheme to stabilize the DG approximation. The resulting time savings of the entropy sensor are expected due to the required matrix multiplication that the modal sensor must perform. The order of the matrix, $[(N+1)^2, (N+1)^2]$, increases with an increasing order of the polynomial approximation, N. Therefore, in actuality, it is the modal sensor becoming less efficient as the resulting larger matrix multiplication becomes increasingly more computationally expensive to perform.

Sensor	N=5	N=7	N=9	N=11
Modal	9.658E-04	7.954E-04	7.268E-04	5.899E-04
Entropy	1.168E-03	8.699E-04	7.696E-04	6.280E-04

Table 5.4: Sod Shock Tube L^2-Error for Density Values



(a) Total Number of Elements Activated (b) Total Amount of Artificial Viscosity Applied

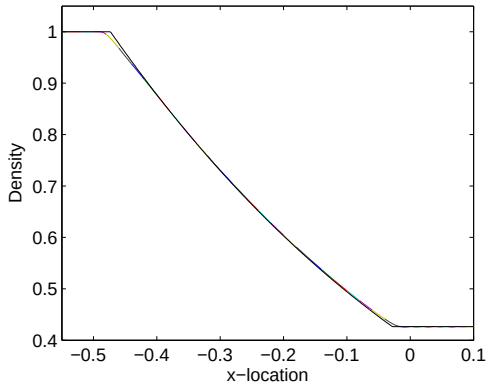
Figure 5.8: Artificial Viscosity Statistics per Order of the Approximating Polynomial for Sod Shock Tube

N=5	N=7	N=9	N=11
-1.27%	-1.93%	-5.75%	-9.62%

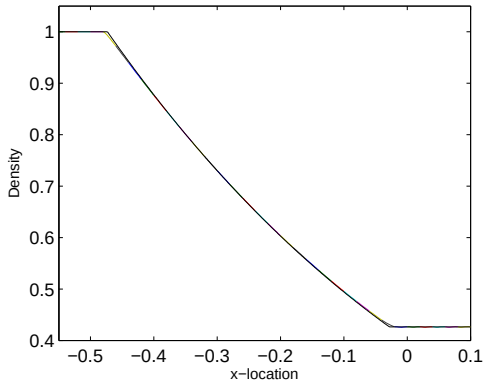
Table 5.5: Wallclock Percent Decrease of Entropy Sensor from Modal Sensor for Sod Shock Tube

5.2.1 Case for N = 11, NI = 100

Focusing on a comparison of the test case corresponding to $N = 11$ and $NI = 100$, the entropy sensor had a 9.62% decrease in wallclock time, a 66.07% decrease in the number of activated elements, and a 77.76% decrease in total amount of artificial viscosity applied throughout the simulation. The expansion wave, contact discontinuity, and shock wave features are magnified in Figures 5.9, 5.10, and 5.11, respectively, so that any diffusion effects may be highlighted. The exact solution in each figure is given by the solid black line, whereas the multicolored lines are each elemental approximated local solution. The modal and entropy sensors both exhibit diffusive effects near the onset of the expansion region (Fig.5.9). The solution resulting from the modal sensor has lost the initial sharpness and is occurring further upstream compared to the exact solution. The solution resulting from the entropy sensor shows a sharper profile at the start of the expansion and a closer approximation, compared to the modal sensor, of the exact solution. The modal sensor applied less diffusive effects in the region of the contact discontinuity compared to the entropy sensor (Fig. 5.10). This implies that the entropy sensor applied a larger total amount of viscosity at the contact discontinuity compared to the modal sensor. The results for both sensors appear almost exact for the approximated shock wave (Fig. 5.11). The only noticeable difference is a single spurious nodal coefficient for the entropy sensor at the beginning of the element upstream of the shock wave. This spurious point present in the entropy based approximation has been smeared out by the modal sensor due to the increased artificial viscosity.

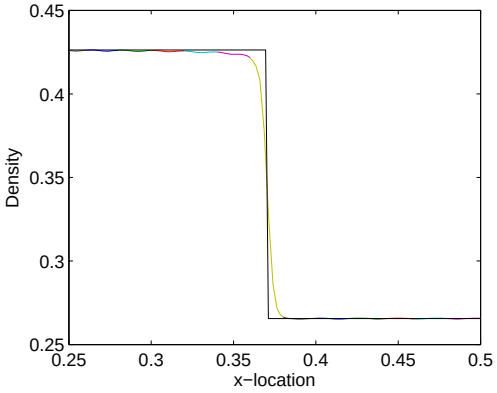


(a) Modal Decay Sensor

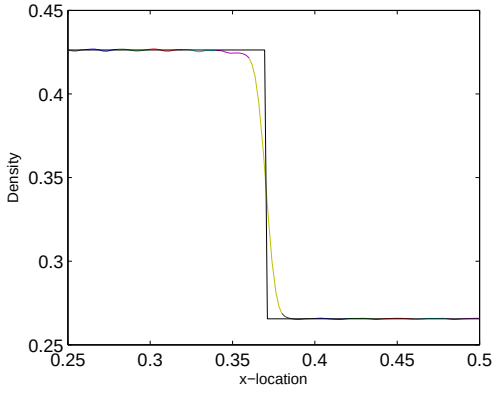


(b) Entropy Residual Sensor

Figure 5.9: Density Profile of Shock tube (Rarefaction Wave) at Final Time 0.4

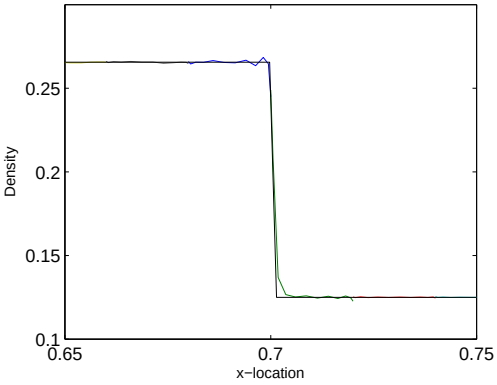


(a) Modal Decay Sensor

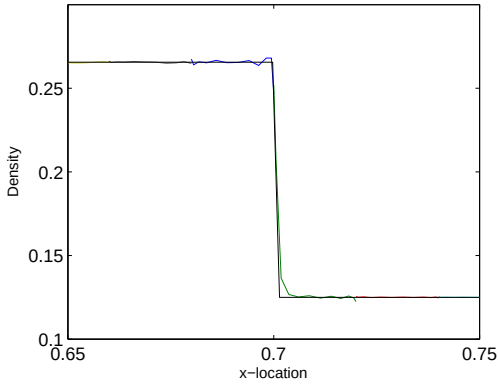


(b) Entropy Residual Sensor

Figure 5.10: Density Profile of Shock tube (Contact Discontinuity) at Final Time 0.4



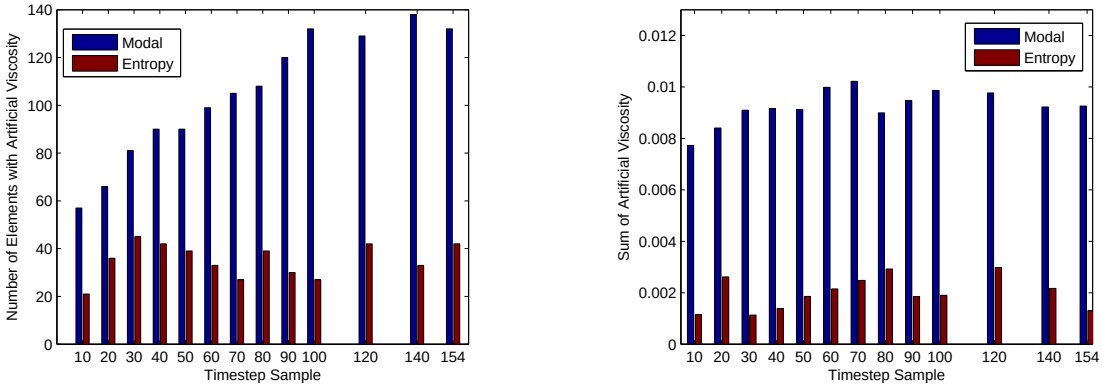
(a) Modal Decay Sensor



(b) Entropy Residual Sensor

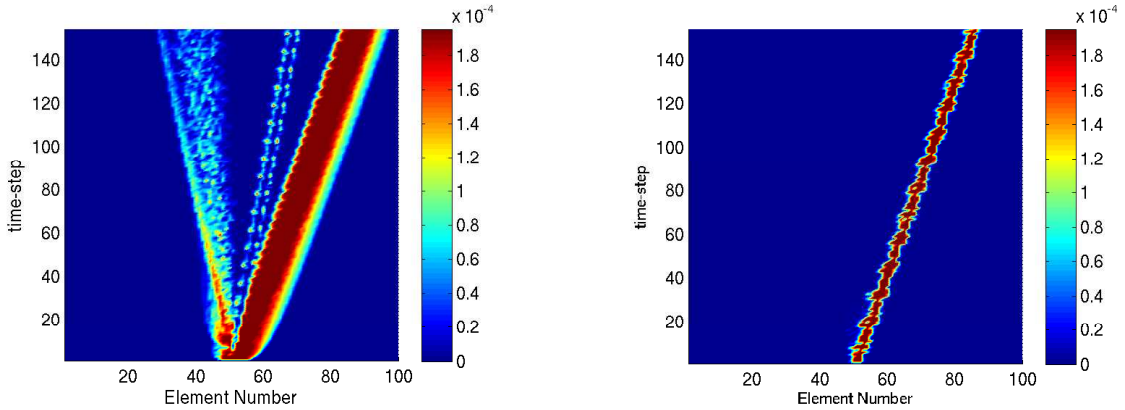
Figure 5.11: Density Profile of Shock tube (Shock Wave) at Final Time 0.4

The artificial viscosity statistics are presented in Figure 5.12 for the modal and entropy sensors. The results show that the modal sensor activated more elements with artificial viscosity and also applied a substantially larger total sum of artificial viscosity at each sampled timestep when compared to the entropy sensor. To aid in explaining these results, Figure 5.13 presents a graphical plot of the total amount of artificial viscosity applied per element at each timestep. The modal sensor applied the largest amounts of artificial viscosity to the expansion region and shock wave continuously, while the contact discontinuity was diffused with varying amounts of viscosity corresponding to the crossing of the element interfaces. The entropy sensor activated elements with the highest amounts of artificial viscosity in a region confined to the shock wave as it moved downstream.



(a) Total Number of Elements Activated (b) Total Amount of Artificial Viscosity Applied

Figure 5.12: Stats for Artificial Viscosity Sensors for Sod Shock tube



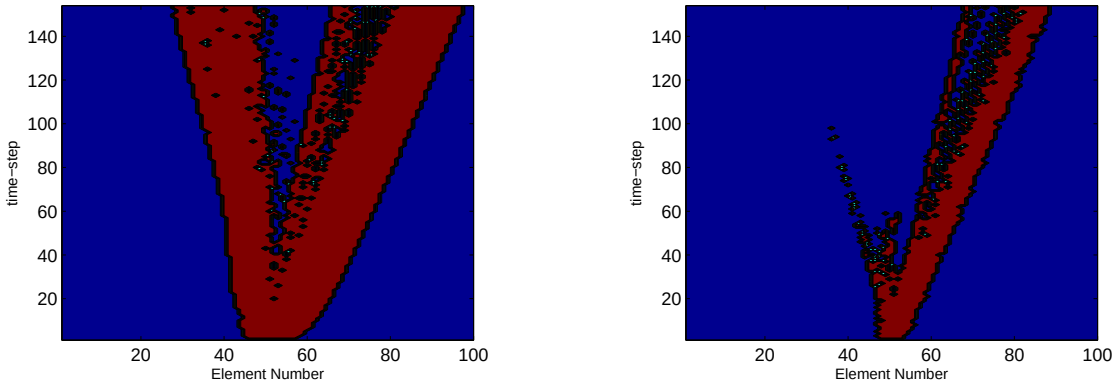
(a) Modal Decay Sensor

(b) Entropy Residual Sensor

Figure 5.13: Time History of Artificial Viscosity for Sod Shock tube Case

One selling point of the development of the entropy based sensor is in the ability to distinguish between a shock wave and a contact discontinuity, with the entropy sensor being the only method known to apply artificial viscosity only to the shock wave [2, 11]. Unfortunately, the results of the present work did not fully support this. Figure 5.14 presents the elements activated with artificial viscosity regardless of magnitude. Any element activated with artificial viscosity is shown in red and the elements with no artificial viscosity are shown in blue. The modal sensor activated elements in the expansion and shock wave regions consistently throughout the simulation while also consistently activating elements before and after the contact discontinuity. In the region of the expansion, the modal sensor decreased the amount artificial viscosity as the simulation was marching through time. As the simulation progressed the modal sensor applied a larger amount of artificial viscosity over a longer simulation time compared to the entropy sensor for the region encompassing the expansion. The entropy sensor shows intermittent activation in the expansion region at the initial timesteps. Once the expansion was diffused to the extent that the numerical method could properly resolve the flow feature, the entropy sensor stopped activating elements. The region of elements starting at the contact discontinuity and extending through the shock wave were activated consistently up to a timestep of approximately 40. After the timestep of 40, the regions of activated elements began to separate for the two separate

flow features. The activation of elements around the contact discontinuity exhibits a fairly consistent pattern while the elements surrounding the shock wave were activated constantly.



(a) Modal Decay Sensor

(b) Entropy Residual Sensor

Figure 5.14: Time History of Activated Elements for Sod Shock tube Case

Figure 5.15 presents the percentage occurrences of the total amount of artificial viscosity applied throughout the sampled timesteps. The modal sensor activates more elements overall than the entropy sensor. The modal sensor shows that 8.5% of the total number of activated elements where withing 0 to 10% of ν_0, whereas the entropy sensor activated 7.4% of the total number of elements throughout the simulation where within the same range. The modal sensor also exhibits a 8.8% occurence of elements within 90% to 100% of the maximum artificial viscosity compared to 2.8% for the entropy sensor.

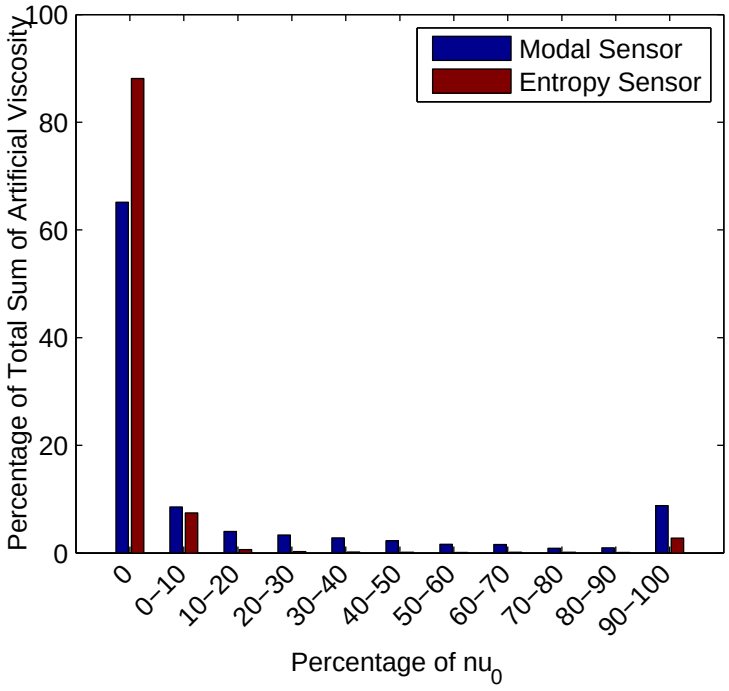


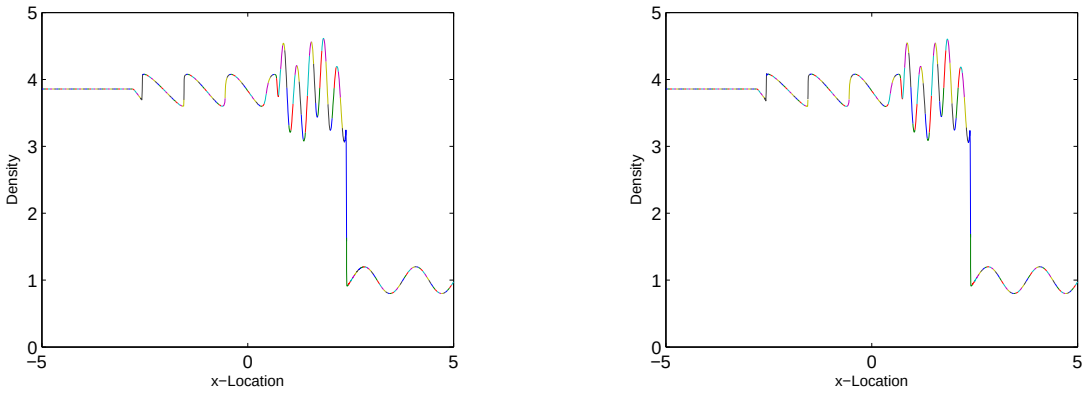
Figure 5.15: Percentage Distribution of Applied Artificial Viscosity for Kelvin-Helmholtz Case

5.3 Shu-Osher Planar Problem

The results of the Shu-Osher test case presented here utilized a 9^{th}-order polynomial approximation and the flow domain was discretized using 200 elements in the i-direction. Over the entire simulation, the entropy sensor took 2.4% less wallclock time, activated 8.68% more elements with artificial viscosity, and applied 67.7% less total artificial viscosity compared to the modal sensor.

The approximated density flowfield values for both sensors at the final simulation time of t = 1.8 are given in Figure 5.16. Moving from upstream to downstream, the final solution consists of a constant density post-shock region, a post-shock low-frequency region, a post-shock high-frequency region, the normal shock, and the pre-shock sinusoidal region. Upon a visual inspection, the resulting flowfields do not exhibit any noticeable differences. Both sensors successfully diffuse all pre- and post-shock spurious activity. The only resulting

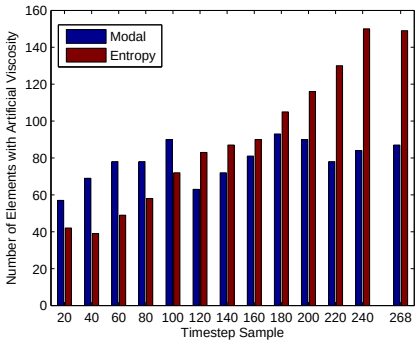
difference requires zooming into the post-shock low-frequency/high-frequency interface. The interface point approximated using the modal sensor exhibits a slightly more diffusive effect than the entropy based approximation. The resulting percent difference between the two density values at the interface is 0.9285%, which shows a very similar approximation.



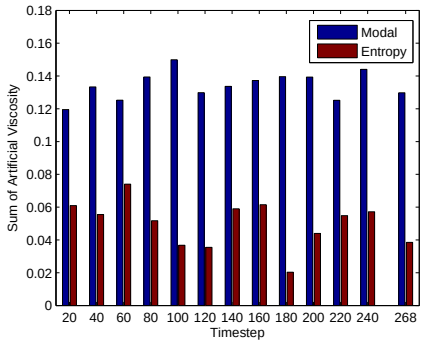
(a) Modal Decay Sensor (b) Entropy Residual Sensor

Figure 5.16: Density Profile of Shu-Osher Case at Final Time 1.8

The total number of elements activated and the total amount of applied artificial viscosity of the modal and entropy sensors at selected timesteps can be seen in Figure 5.17. The modal sensor stays reasonably consistent in the number of elements being activated with artificial viscosity, whereas the entropy sensor increases the number of activated elements throughout the simulation. Despite the increasing number of activated elements, the entropy sensor applies at least half the amount of artificial viscosity throughout the simulation compared to the modal sensor.



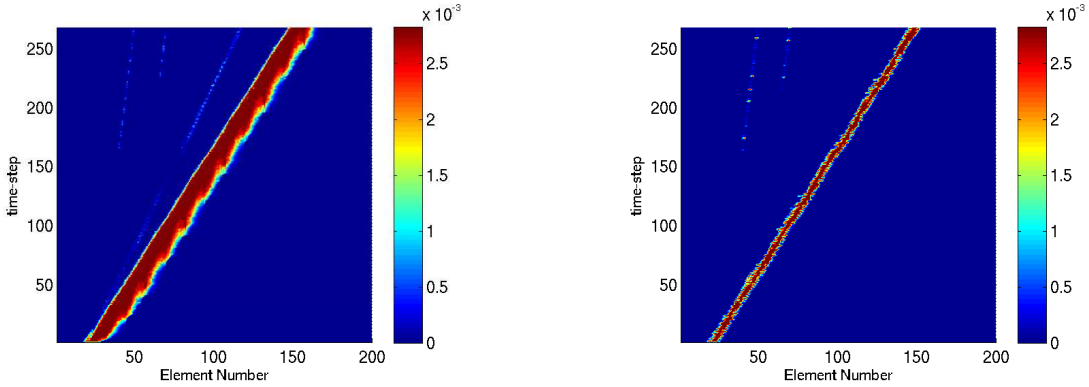
(a) Total Number of Elements Activated



(b) Total Amount of Artificial Viscosity Applied

Figure 5.17: Stats for Artificial Viscosity Sensors for Shu-Osher Test Case

The timestep history of the total amount of artificial viscosity applied in each element is presented in Figure 5.18. The modal sensor activated elements at the shock wave and applied the maximum artificial viscosity in a 10-15 element region downstream of the shock wave. Conversely, the entropy sensor only activated the maximum artificial viscosity in a 3-8 element region downstream of the shock wave. The variation in these two number sets corresponds to the shock wave crossing the element boundaries, with the higher number occurring as the shock crosses the element interfaces. The modal sensor also activated at the interface of the resulting flow where the post-shock high-frequency solution meets the post-shock low-frequency solution. The entropy sensor did not activate with a significant amount of viscosity for this same region. At the location where the smooth upstream density field was meeting the post-shock result, both sensors began activating around an approximate timestep of 150.

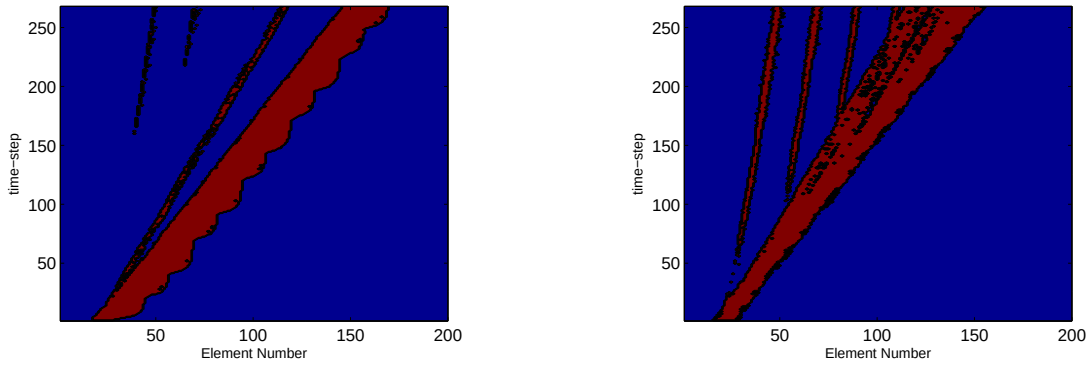


(a) Modal Decay Sensor

(b) Entropy Residual Sensor

Figure 5.18: Time History of Artificial Viscosity for Shu-Osher Case

To highlight the fact that the entropy sensor is activating in 8.68% more elements, Figure 5.19 shows the location of activated elements regardless of the magnitude of artificial viscosity applied. It becomes apparent that the entropy sensor is applying artificial viscosity in regions that were not initially evident, whereas the modal sensor still exhibits the same profiles. The modal sensor applied artificial viscosity at and downstream of the shock wave location, whereas the entropy sensor activates elements at the shock wave and continues activating upstream across the high-frequency post-shock solution as the timestep increases. Both sensors exhibit artificial viscosity activation at the constant density/initial-frequency post-shock interface as well as the initial-frequency post-shock/high-frequency post-shock interface.



(a) Modal Decay Sensor

(b) Entropy Residual Sensor

Figure 5.19: Time History of Elements Activated with Artificial Viscosity for the Shu-Osher Case

Figure 5.20 presents the percentage distribution of the range of artificial viscosity magnitudes used throughout the entire simulation. Both sensors indicate a similar approximately 85% of the total number of elements where not activated with artificial viscosity for the sampled timesteps. The entropy sensor does show that 11.15% of the elements were activated within 0 to 10% of ν_0 compared to 3.2% for the modal sensor. It is again apparent that the modal sensor activates more elements within 90% to 100% of ν_0 as was discovered in previous results.

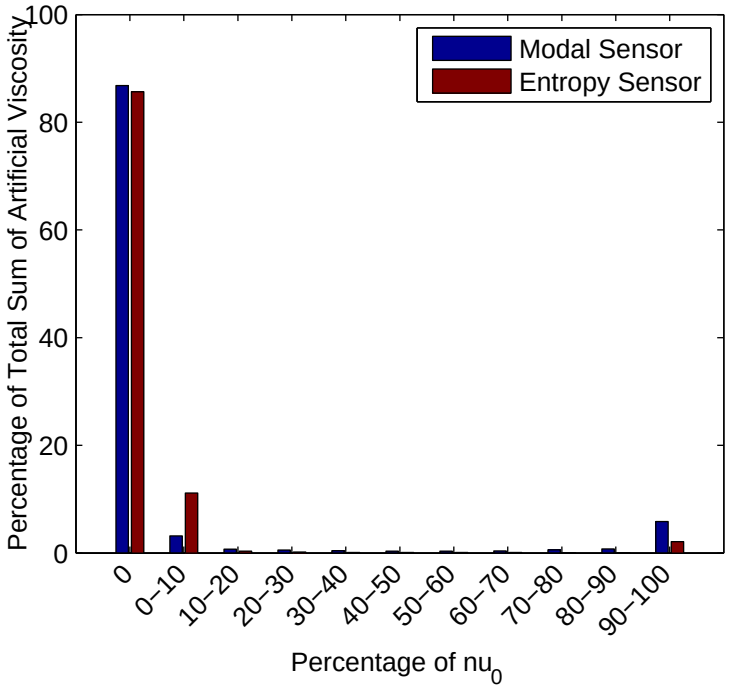


Figure 5.20: Percentage Distribution of Applied Artificial Viscosity for Shu-Osher Case

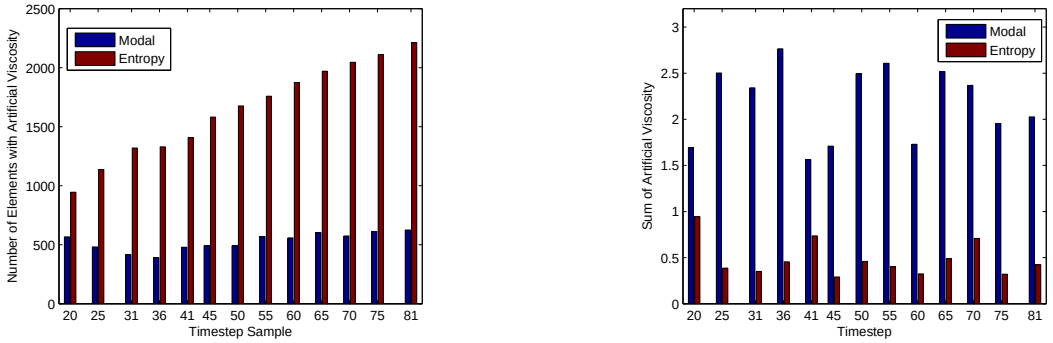
The results have shown that although a considerable difference existed in the way the two sensors applied artificial viscosity to the test case, the resulting flowfields exhibited the same solution. The entropy sensor did show a time-savings over the modal sensor while also applying a smaller sum of the total amount of artificial viscosity applied.

5.4 Shock-Vortex Interaction

The results of the shock-vortex interaction presented below utilized a 9^{th}-order polynomial approximation inside each element. The flow domain was discretized using 60 elements in the i-direction and 45 elements in the j-direction. The artificial viscosity sensors were tested using the maximum artificial viscosity parameter of $c_{max} = 1$ and 4. The increased value of ν_0 allowed comparisons to be made for the resulting behaviors of modal and entropy

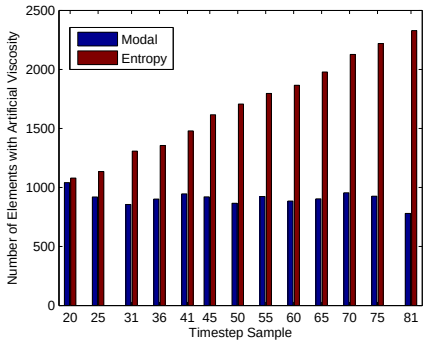
sensors. Results were also included for the entropy sensor parameter $c_s = 0.1$ in order to show the effects of the choice in the parameter.

The artificial viscosity statistics are presented for $c_{max} = 1$ in Figure 5.21 and $c_{max} = 4$ in Figure 5.22. The results show the entropy sensor is activating elements increasingly through the simulation for both c_{max} values, while the modal sensor uses roughly the same amount of elements at each timestep for each c_{max} case. Focusing on Figure 5.21a and Figure 5.22a, a significant difference is noticed using the two c_{max} values for the modal sensor while the entropy sensor stays more consistent. The modal sensor activated 66.11% more elements while the entropy sensor activated 3.68% more elements for using $c_{max} = 4$. Comparing Figures 5.21b and 5.22b for the total artificial applied at each timestep results in a 310.74% increase for the modal sensor and only a 17.72% increase for the entropy sensor when using $c_{max} = 4$. The percent increases are for the average values at each timestep because an evaluation over the entire simulation run would be skewed by the increased number of timesteps due to a larger ν_0 value. The modal sensor showed a substantial increase in both the total number of elements activated and the total amount of artificial viscosity compared to entropy sensor when c_{max} is increased from 1 to 4.

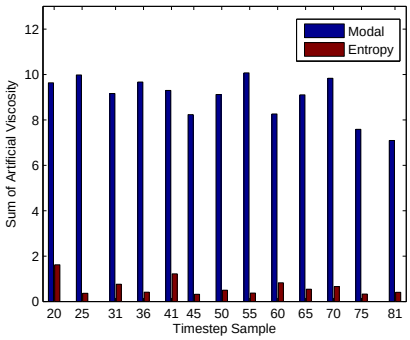


(a) Total Number of Elements Activated (b) Total Amount of Artificial Viscosity Applied

Figure 5.21: Stats for Artificial Viscosity Sensors for Shock-Vortex Test Case, $c_{max} = 1$



(a) Total Number of Elements Activated



(b) Total Amount of Artificial Viscosity Applied

Figure 5.22: Stats for Artificial Viscosity Sensors for Shock-Vortex Test Case, $c_{max} = 4$

Expanding upon the results of the previous section, Figure 5.23 presents the percentage distribution of the range of artificial viscosity magnitudes that each sensor used throughout the selected timesteps. The modal sensor exhibits an increase in the number of elements not activated with artificial viscosity compared to the entropy sensor as was demonstrated in previous the previous results. The entropy sensor shows a significant increase in the percentage of elements (49.1%) that are activated with 0 to 10% of ν_0 compared to the modal sensor (5.75%). The percentage of elements activated without viscosity or with the 0 to 10% range constitutes 96.45% of the total number of elements sampled, with the remainder of the ranges being relatively close. The modal sensor activates 5.4% of the elements with the upper range of artificial viscosity compared to 0.56% for the entropy sensor in the same range. The entropy sensor tends to activate a large percentage of elements with small amounts of viscosity whenever sharp gradients are present. This trend was also evident for the Kelvin-Helmholtz test case. For test cases with sharp gradients, the entropy sensor presents a better capability of handling the resulting flow features.

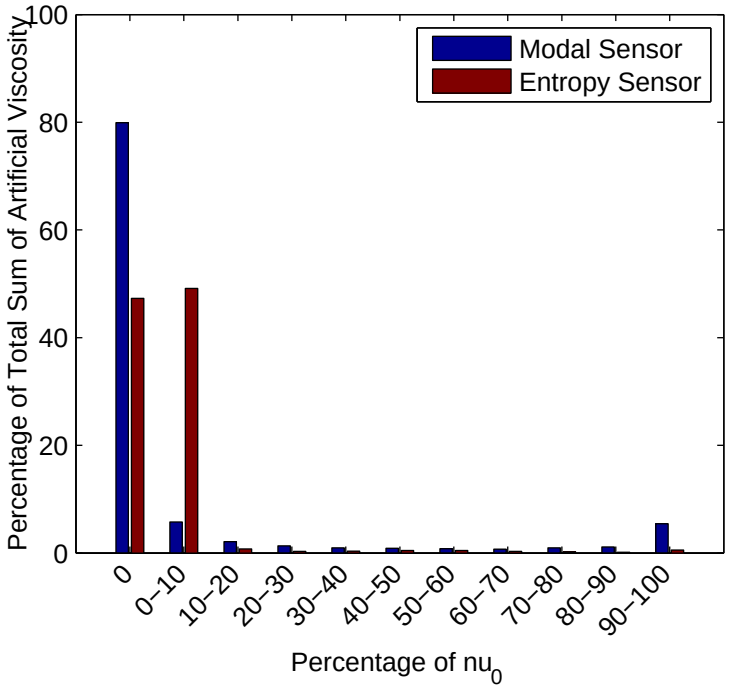
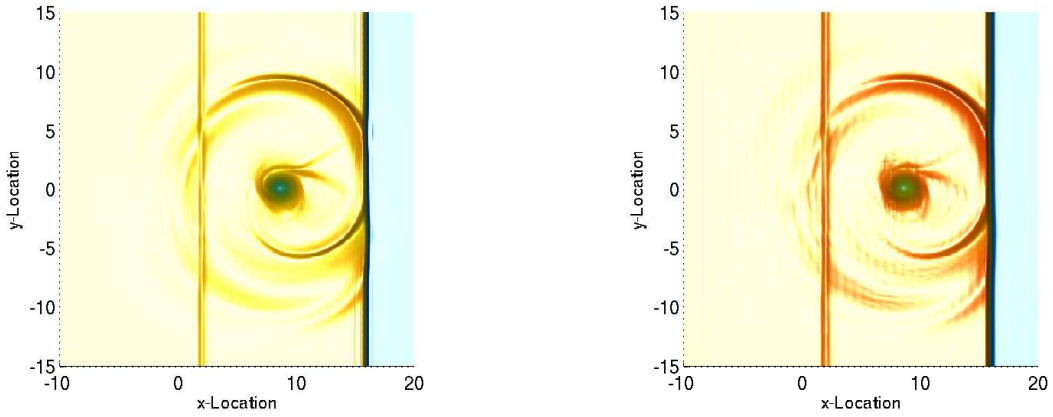


Figure 5.23: Percentage Distribution of Applied Artificial Viscosity for Shock-Vortex Case

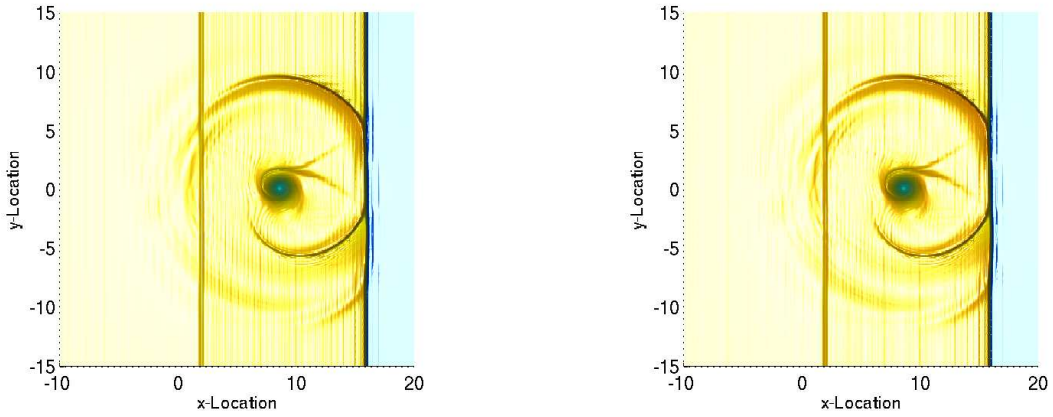
The density profile at the end of the simulation are presented in Figure 5.24 for the modal sensor and Figure 5.25 for the entropy sensor. One flow feature worth pointing out is the wiggle present in each density profile. This feature is the first vertical line present in each of the figures. The resulting wiggle is a numerical remnant of starting each simulation run using a discontinuous solution for the shock wave. This may be avoided by first solving for a steady state solution of the shock wave and then introducing the vortex and allowing the shock wave to convect downstream. Figure 5.24 shows the density field computed using the modal sensor for both c_{max} values. Severe dissipation is present for the case of $c_{max} = 4$ as can be seen by the increase of the normal shock width and the dissipation of the double shock inside the vortex. Using c_{max} of 4 results in a 161.87% increase in wallclock time compared to using $c_{max} = 1$. The density field is given Figure 5.25 and shows the comparison using the two c_{max} values for the entropy sensor. The $c_{max} = 4$ results show a reduction in the number of spurious oscillations when compared to the $c_{max} = 1$ test case without noticeable

added effects of dissipation. Comparing the $c_{max} = 4$ to the $c_{max} = 1$ case for the entropy sensor resulted in a wallclock time increase of 175.19%. Compared to the modal sensor, the entropy sensor showed only a slightly higher increase in wallclock time, but a significantly lower total number of activated elements and total artificial viscosity applied when the c_{max} value was increased.



(a) $c_{max} = 1$ (b) $c_{max} = 4$

Figure 5.24: Density Field for Modal Decay Sensor at Final Time = 16.2



(a) $c_{max} = 1$ (b) $c_{max} = 4$

Figure 5.25: Density Field for Entropy Residual Sensor at Final Time = 16.2

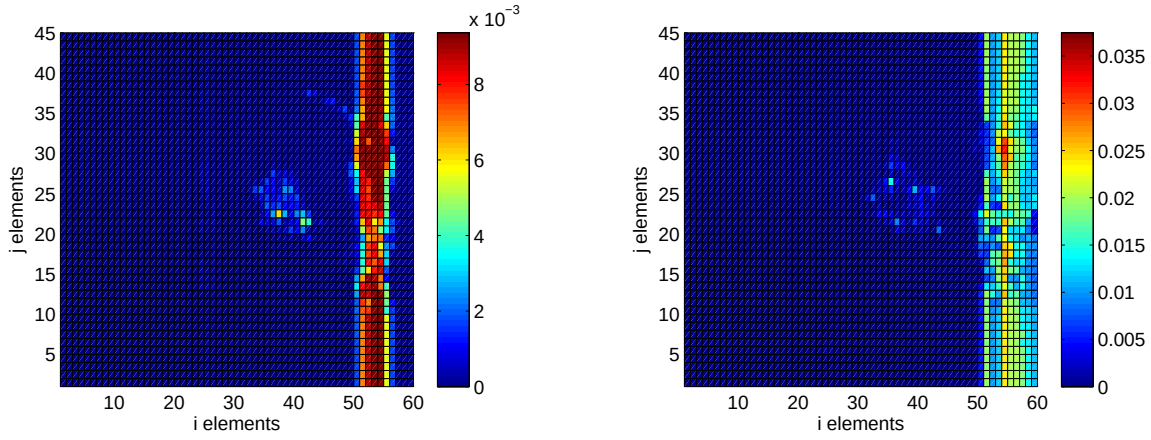
Table 5.6 gives the values of the percent difference of the entropy sensor compared to the modal sensor for each value of c_{max}. The percent differences given are the difference in wallclock computation time, the difference in the total number of activated elements over

the course of the simulation, and the total sum of artificial viscosity applied throughout the simulation.

c_{max}	Wallclock	Activated Elements	Sum of A.V.
1	-3.44%	159.58%	-79.33%
4	1.47%	55.38%	-94.77%

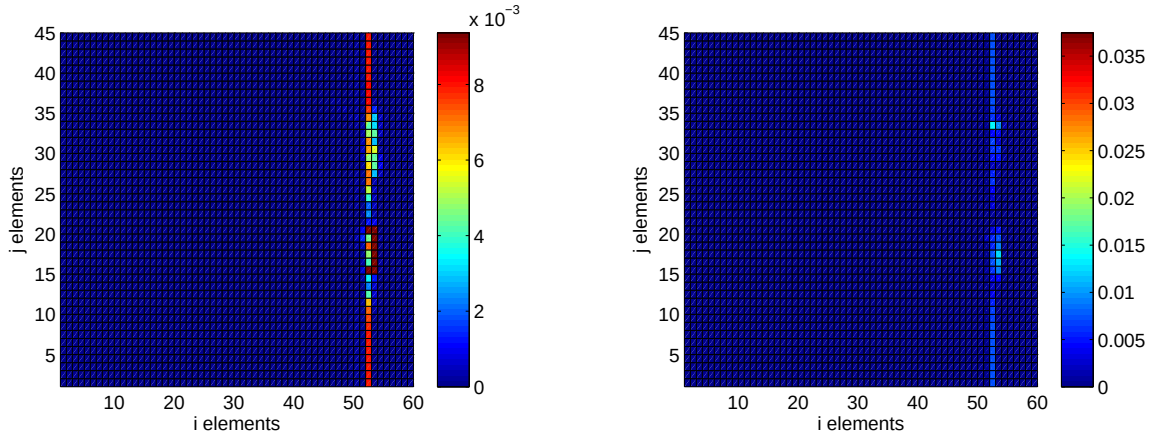
Table 5.6: Percent Difference of Entropy Sensor Compared to the Modal Sensor

The artificial viscosity values at the final simulation time are given in Figure 5.26 for the modal sensor and Figure 5.27 for the entropy sensor. When c_{max} was increased to 4, both methods show a substantial decrease in activating the elements with the maximum amount of artificial viscosity. One important aspect of the results is the activation of artificial viscosity that occurs in the vortex for the modal sensor. The vortex, while undergoing a disturbance caused by the shock interaction, still maintains a smooth profile. The modal sensor applies a noticeable amount of artificial viscosity to the vortex whereas the entropy sensor does not apply a noticeable amount of artificial viscosity observed in the figure.



(a) $c_{max} = 1$ (b) $c_{max} = 4$

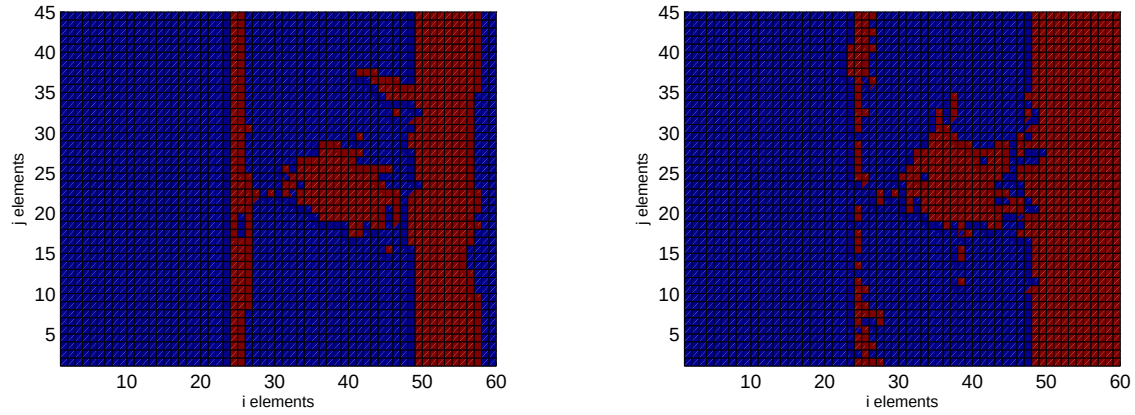
Figure 5.26: Artificial Viscosity for Modal Decay Sensor at Final Time = 16.2



(a) $c_{max} = 1$ (b) $c_{max} = 4$

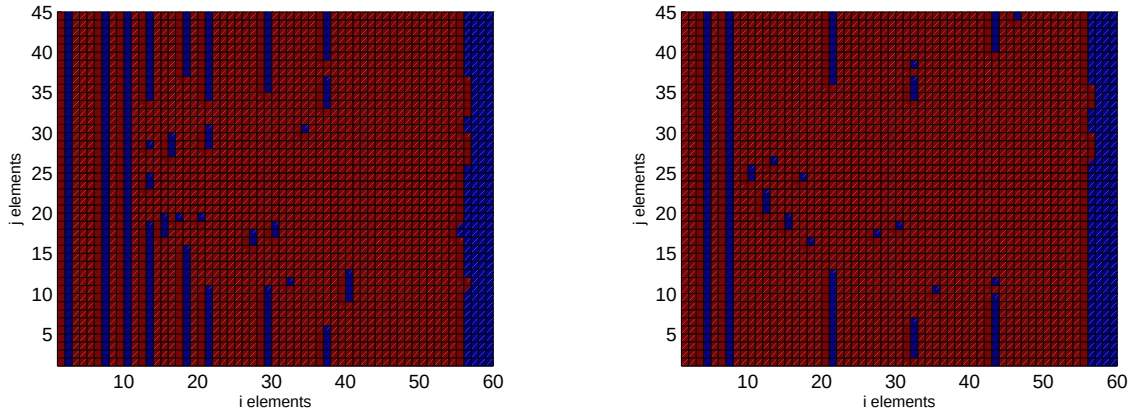
Figure 5.27: Artificial Viscosity for Entropy Residual Sensor at Final Time = 16.2

Figures 5.28 and 5.29 give the plots indicating which elements are activated with artificial viscosity, regardless of magnitude, for the modal and entropy sensors, respectively. Comparing these figures to Figure 5.26 for the modal sensor and Figure 5.27, the modal sensor is applying artificial viscosity mainly in large magnitudes and is applying viscosity downstream of the normal shock. The entropy sensor is activating a small magnitude of artificial viscosity focusing in the region upstream of the normal shock with large magnitudes of artificial viscosity activating in elements containing large gradients of the sensing variable.



(a) $c_{max} = 1$ (b) $c_{max} = 4$

Figure 5.28: Elements Activated with Artificial Viscosity for Modal Decay Sensor at Final Time = 16.2

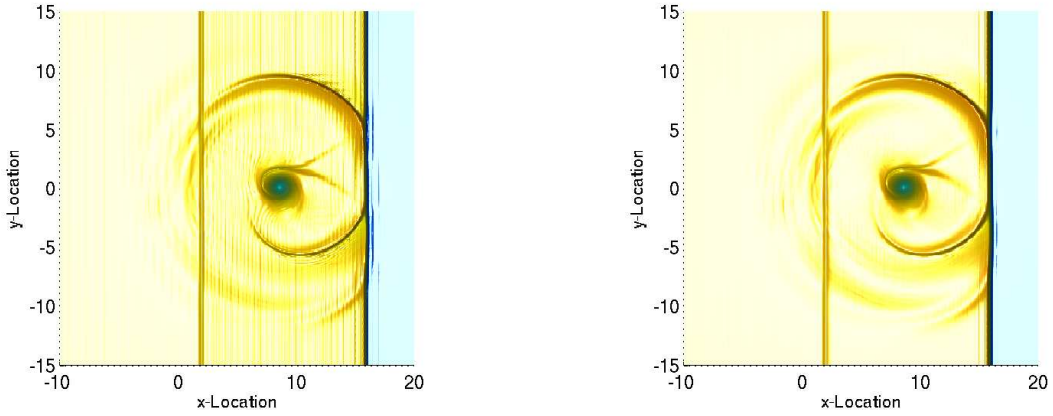


(a) $c_{max} = 1$ (b) $c_{max} = 4$

Figure 5.29: Elements Activated with Artificial Viscosity for Entropy Residual Sensor at Final Time = 16.2

The entropy sensor has the drawback of being a method that requires a user defined parameter in order to scale the artificial viscosity for a given flow simulation. One advantage that does come from the selection of the parameter is the ability to fine tune the amount of artificial viscosity that is applied to the flowfield. The entropy sensor was found to continuously activate more elements with artificial viscosity when compared to the modal sensor. Whereas both sensors typically applied the maximum artificial viscosity when a strong shock was encountered, the ability to alter the $|\Delta\rho s|_{ref}$ value will adjust the small and intermediate values of artificial viscosity that occur with the entropy sensor. Figure 5.30 presents the final density field values for the entropy sensor with $|\Delta\rho s|_{ref} = 0.5$ and 0.1. The entropy sensor is activating approximately the same amount of elements, -0.35% difference from the $c_s = 0.5$ case, but is applying 69.18% more total artificial viscosity. The increase in viscosity removes the spurious oscillations at the cost of diffusing the flowfield. Using $c_s = 0.1$ results in the final density field being subjected to more diffusive errors while reducing the dispersive errors. The increased diffusion is especially noticeable in both the normal shock and the double shock wave emanating from the vortex. The resulting profile of both flow features have been widened, or smeared, over the course of the simulation.

The ability to adjust the scaling of the artificial viscosity allows the user to experimentally determine the amount of artificial viscosity that is applied throughout the approximation. While having to determine the c_s parameter reduces the robustness of the entropy sensor, it does allow a way to ensure that the minimum amount of artificial viscosity is being applied throughout the flowfield and physically correct features are not being overly diffused.



(a) $c_s = 0.5$ (b) $c_s = 0.1$

Figure 5.30: Density Field for Entropy Residual Sensor at Final Time = 16.2

Chapter 6

Conclusions and Future Work

6.1 Conclusions

The present work focused on the application of artificial viscosity as a way to stabilize the high-order approximation of the Euler equations by the discontinuous Galerkin method. In order to apply artificial viscosity in the elements where the Gibb's phenomenon are occurring, as opposed to a constant artificial viscosity across the entire domain, two methods of detecting where the spurious oscillations where occurring where analyzed.

The modal sensor, based on the modal decay of the approximate solution, resulted in stable solutions for all the test cases presented by analyzing the numerical behaviour of the modal coefficients at each timestep. Although the modal sensor proved to be overly diffusive for certain test cases, the sensor is considered robust since no user parameters are needed in order to scale the artificial viscosity. The modal sensor was more efficient for lower-order polynomial approximations inside each element due to the required computational expense involved in acquiring the modal coefficients. The modal sensor also was more efficient in the artificial viscosity applied per element thus resulting in a large decrease in the amount of elements that are activated for each timestep. This translates into saved computational expense due to the calculation of the second-order derivatives in the elements where artificial viscosity is being applied.

The entropy sensor, which was based on the entropy production residual, resulted in a method that is based on the physics of the resulting flowfield. The entropy sensor successfully diffused each test case so that the resulting DG scheme remained stable. The entropy sensor become computationally more efficient as the approximating polynomial increased. Although the entropy sensor requires the user input of the parameter, $|\Delta\rho s|_{ref}$, the amount

of applied artificial viscosity may be tailored to more accurately apply the minimum amount of diffusion in order that the scheme remains stable. The entropy sensor applied less total artificial viscosity during the entire simulation as compared to the modal sensor. This came at the cost of applying a smaller magnitude of artificial viscosity to more elements than the modal sensor. The larger number of activated elements results in more computational expense being required at each timestep.

Both artificial viscosity sensors tested had benefits as well as drawbacks when comparing the two methods. The modal sensor is ideally suited as a plug-and-chug sensor that requires no user input but still results in a stable numerical approximation. Unfortunately, this comes at the cost of an over diffusive behavior for certain flow conditions. The modal sensor also exhibited a higher threshold before activating artificial viscosity in the approximation as encountered in the isentropic vortex test case. This is beneficial if a quick turnaround is required and a coarse mesh is to be used. If a low-order polynomial is also used, the computational cost required by the modal sensor may also be offset. Because the goal of the DG method is to use higher-order polynomial approximations in each element in order to achieve higher accuracy, the entropy sensor is favored due to the reduced computational cost for higher-order polynomial approximations. This comes at the cost of the user-defined parameter, $|\Delta\rho s|_{ref}$, that must be determined for each resulting flowfield. The choice of the parameter seems to be non-trivial, as a starting value may be estimated by inspection of the initial flowfield values, ρs, and then refined using a course mesh. The entropy sensor also exhibited a less diffusive nature when the approximated solution resulted in small-scale features as was encountered in the shock-vortex test case. Another flow feature that proved advantageous for the entropy sensor was sharp gradients opposed to discontinuities. This was evident in the Kelvin-Helmholtz test case where the entropy sensor significantly exhibited less diffusive effects compared to the modal sensor. The entropy sensor proved to be more accurate for the isentropic vortex case and also for the expansion and shock wave approximations of the shock tube case, whereas the modal sensor more accurately approximated the

contact discontinuity for the shock tube test case. Considering the difference in the accuracy at the contact discontinuity, and the fact that the difference decreases with grid refinement, the entropy sensor is viewed as the advantageous method.

The two sensors presented have both demonstrated the ability to successfully stabilize the numerical approximation of the Euler equations when the DG method is employed. Although the requirement of a user-defined parameter of the entropy sensor leads to an additional unknown, the advantages presented outweigh the complication of determining the unknown parameter. In addition, the determination of $|\Delta\rho s|_{ref}$ enables the minimum amount of artificial viscosity to be exploited such that physically correct flow features are not unduly diffused.

6.2 Future Work

The two sensors presented have been compared as close to the original authors methods as allowable. It has been observed that other steps may be taken to improve the results obtained from each method. Because only the elements containing artificial viscosity undergo the determination of a second-order differential to apply the artificial viscosity, it is beneficial to activate only the necessary elements in order to maintain stability. The entropy sensor applied a small amount of artificial viscosity, compared to the maximum value in the entire domain, to a large number of elements. Throughout the research, the reason behind these small magnitudes seemed to be corresponding with the grid resolution of the test case. As the number of elements increased, the number of elements with small amounts of artificial viscosity would decrease due to the higher resolution. Instead of having to apply a fine mesh in order to achieve this, it may be beneficial to filter out any artificial viscosity occurring below a certain tolerance. It has been observed that filtering off any value of artificial viscosity approximately below 10% of ν_0 would significantly reduce the number of elements activated. An investigation of how to determine what the given tolerance should be and the effects that this would have on the stability of the approximated solution should be carried

out. The reduction of the activated elements would decrease the associated computational expense resulting in a decrease in wallclock time.

Extending the sensor comparison to 3-D would be beneficial in comparing the wallclock times for each method. The modal sensor, having to perform an $[(N+1)^2, (N+1)^2]$ matrix multiplication for a quadrilateral 2D case, would have to perform an $[(N+1)^3, (N+1)^3]$ matrix multiplication for a cubic 3D case. Considering that each sensor is being computed inside each Runge-Kutta step, the increased cost of computing the modal sensor should prove to be increasingly more expensive than the entropy sensor. It is also pointed out that these problems were run for relatively simple test cases. The addition of the third dimension would result in a significant number of elements being added to the flowfield and allow a better comparison to the benefits of the entropy sensor. Both sensors can be extended to 3-D with non-trivial changes to the methods already employed.

The present work computed the needed artificial viscosity at each Runge-Kutta step. It has been observed through the research that the methods were stable if the sensor was, for instance, utilized at only two of the three Runge-Kutta steps. The modal sensor may also be computed at the beginning of each Runge-Kutta step with the same value used for the entire time integration step. An analysis of the overall effect of these procedures on the time savings, resulting accuracy, and needed stabilization would provide additional insight. It appears the modal sensor would benefit from this type of analysis due to the reduction in the number of times the matrix multiplication would need to be carried out.

The artificial viscosity applied in this work was piecewise discontinuous across the discretized domain. Barter and Darmofal [14] have shown that the application of a non-smooth artificial viscosity can itself cause oscillations in the solution approximation downstream. Upgrading the artificial viscosity such that a smooth transition occurs throughout the entire flow domain would benefit the overall goal of reducing unwanted oscillations in the flowfield. Whether this is accomplished through the use of a simple linear interpolation between each element or through a pde-based model [14], should be determined by the improvement in

the stabilization and accuracy versus the incurred computational expense required by each method. In addition the application of the artificial viscosity may be applied using the form presented by Zinghan and Guermond in Equation 3.10. This form may be beneficial at reducing oscillations that occur upstream due to the parabolic regularization form of viscosity terms that were employed here.

The modal sensor, being a parameter free method, does not allow the user to adjust the amount of applied artificial viscosity for different flowfield conditions. Both methods exhibited different approaches to stabilizing the numerical approximation: the modal sensor using a large amount of artificial viscosity in a small number elements and the entropy sensor applying a smaller amount of artificial viscosity in a larger number of elements. It has been observed that a simple scaling factor to the determination of the maximum artificial viscosity, ν_0, could successfully limit the over diffusive nature of the method. Scaling ν_0 by multiplying the original value by a factor of $^1/_{100}$ for the Kelvin-Helmholtz test case produced a result that rivaled the entropy sensor. With the introduction of a scaling parameter, the increased computational expense may prove to be offset by the reduction in the number of activated elements for 2-D problems.

Bibliography

[1] Klöckner, A., Warburton, T., and Hesthaven, J., "Viscous Shock Capturing in a Time-Explicit Discontinuous Galerkin Method," *Mathematical Modelling of Natural Phenomena*, 2010, pp. 1–42.

[2] Zingan, V., Guermond, J. L., Morel, J., and Popov, B., "Implementation of the Entropy Viscosity Method with the Discontinuous Galerkin Method," *Computational Methods Applied Mechanical Engineering*, Vol. 253, 2013, pp. 479–490.

[3] Shelton, A. B., *A Multi-Resolution Discontinuous Galerkin Method for Unsteady Compressible Flows*, Ph.D. thesis, Georgia Institute of Technology, 2012.

[4] Hesthaven, J. and Warburton, T., *Nodal Discontinuous Galerkin Methods - Algorithms, Analysis, and Applications*, Springer, New York, NY, 1st ed., 2008.

[5] Tannehill, J. C., Anderson, D. A., and Pletcher, R. H., *Computational Fluid Dynamics and Heat Transfer*, Taylor and Francis, Philadelphia, PA, 2nd ed., 1997.

[6] Persson, P. O. and Peraire, J., "Sub-Cell Shock Capturing for Discontinuous Galerkin Method," *In Proc. of the 44th AIAA Aerospace Sciences Meeting and Exhibit*, Vol. 112, 2006.

[7] Atkins, H. L. and Pampell, A., "Robust and Accurate Shock Capturing Method for High-Order Discontinuous Galerkin Methods," *AIAA*, 2011.

[8] Huerta, A., Peraire, J., and Casoni, E., "A Simple Shock-Capturing Technique for High-Order Discontinuous Galerkin Methods," *International Journal of Numerical Methods in Fluids*, 2011, pp. 1614–1632.

[9] Gautschi, W., *Orthogonal Polynomials Computation and Approximation*, Oxford University Press, Oxford, NY, 1st ed., 2004.

[10] Hesthaven, J. and Warburton, T., "From Electrostatics to Almost Optimal Nodal Sets for polynomial Interpolation in a Simplex," *SIAM Journal of Numerical Analysis*, Vol. 35, No. 2, 1998, pp. 655–676.

[11] Zingan, V., *Discontinuous Galerkin Finite Element Method for the Nonlinear Hyperbolic Problems with Entropy-Based Artificial Viscosity Stabilization*, Ph.D. thesis, Texas A&M University, 2012.

[12] Gottlieb, S. and Shu, C.-W., "Total Variation Diminishing Runge-Kutta Schemes," *Mathematics of Computation of the American Mathematical Society*, Vol. 67, No. 221, 1989, pp. 73–84.

[13] Liou, M.-S., "A sequel to AUSM, Part II: AUSM+-up for all speeds," *Journal of Computational Physics*, Vol. 214, No. 1, 2006, pp. 137–170.

[14] Barter, G. E. and Darmofal, D. L., "Shock Capturing with PDE-based Artificial Viscosity for DGFEM: Part I. Formulation," *Journal of Computational Physics*, Vol. 229, No. 5, 2010, pp. 1810–1827.

[15] Guermond, J. L., Pasquetti, R., and Popov, B., "Entropy Viscosity Method for Nonlinear Conservation Laws," *Journal of Computational Physics*, 2011.

[16] Guermond, J.-L., Pasquetti, R., and Popov, B., "From Suitable Weak Solutions to Entropy Viscosity," *Journal of Scientific Computing*, Vol. 49, No. 1, 2011, pp. 35–50.

[17] Fidkowski, K. and Roe, P., "An Entropy Adjoint Approach to Mesh Refinement," *SIAM Journal of Scientific Computing*, 2010.

[18] Toro, E. F., *Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction*, Springer-Verlag Berlin Heidelberg, 3rd ed., 2009.

[19] Shu, C.-W. and Osher, S., "Efficient Implementation of Essentially Non-Oscillatory Shock-Capturing Schemes, II," *Journal of Computational Physics*, Vol. 83, No. 1, 1989, pp. 32–78.

[20] Inoue, O. and Hattori, Y., "Sound Generation by Shock-Vortex Interactions," *Journal of Fluid Mechanics*, Vol. 380, No. 3, 1999, pp. 81–116.