The Penultimate Scheme for Systems of Conservation Laws: Finite Difference ENO with Marquina's Flux Splitting[1]

Ronald P. Fedkiw

Computer Science Department
Stanford University, Stanford, California 94305
Email:fedkiw@cs.stanford.edu

Barry Merriman

Department of Mathematics
University of California Los Angeles, Los Angeles, California, 90095
Email:barry@math.ucla.edu

Rosa Donat

Departament de Matemàtiques Aplicades
Universitat de València, Spain
Email:donat@uv.es

Stanley Osher

Department of Mathematics
University of California Los Angeles, Los Angeles, California, 90095
Email:sjo@math.ucla.edu

Keywords: Euler Equations, Compressible Flow, Finite Difference Methods, ENO, Flux Splitting

Abstract: This paper provides a users' guide to a new, general finite difference method for the numerical solution of systems of convection dominated conservation laws. We include both extensive motivation for the method design, as well as a detailed formulation suitable for direct implementation.

Essentially Non-Oscillatory (ENO) methods are a class of high accuracy, shock capturing numerical methods for hyperbolic systems of conservation laws, based on upwind biased differencing in local characteristic fields. The earliest ENO methods used control volume discretizations, but subsequent work [12] has produced a simpler finite difference form of the ENO method. While this method has achieved excellent results in a great variety of compressible flow problems, there are still special situations where noticeable spurious oscillations develop. Why this occurs is not always understood, and there has been no elegant way to eliminate these problems.

Based on the extensive work of Donat and Marquina [1], it appears that these difficulties arise from using a *single* transformation to local characteristic

[1]This paper was presented in "Solutions of PDE" Conference in honour of Prof. Roe on the occassion of his 60th birthday, July 1998, Arachaon, France

variables at cell walls in the course of computing wall fluxes. In concrete terms this is the practice of evaluating the flux Jacobian matrix at cell walls using an average of adjacent cell states, such as the Roe average or linear average. When the states differ greatly across the cell wall, using such an intermediate state in the transformation adds subtle spurious features to the solution. As an alternative, Donat and Marquina recommend obtaining the wall flux from a splitting procedure based on fluxes computed separately from the left and right sides. This approach avoids introducing artificial intermediate states, and seems to improve the robustness of many characteristic based methods.

Applying their splitting in the ENO framework, the left and right sided fluxes are evaluated by the ENO interpolation technique, i.e. using the smoothest high order interpolations from each side. In the resulting method, the spurious oscillations are eliminated without sacrificing high resolution. Thus this seems to be an ideal scheme for general hyperbolic systems: it provides high accuracy and shock capturing without numerical artifacts, problem dependent "fixes", or free parameters that must be "tuned". (Of course, for scalar equations this "fix" is unnecessary and nonexistent.)

This paper is intended as a self-contained guide to this new approach, in the context of solving general systems of convection-diffusion-reaction conservation laws. We provide all the conceptual background needed to understand the design of numerical methods for systems of hyperbolic conservation laws in general, and the finite difference ENO method and Marquina's flux splitting procedure in particular. We then give a detailed presentation of the preferred form of ENO with Marquina's splitting. We conclude with one example where this eliminates a severe, non-physical oscillation in a complicated ENO based calculation.

1 Introduction

Essentially Non-Oscillatory (ENO) methods were developed to address the special difficulties that arise in the numerical solution of systems of nonlinear conservation laws, such as those arising in high speed gas dynamics and other convective transport problems. Numerical methods for these problems must be able to handle steep gradients—shocks and contact discontinuities—that may develop spontaneously and then persist in these flows. Classical numerical schemes had a tendency to either produce large spurious oscillations near steep gradients, or to greatly smear out both these gradients and the fine details of the flow. An excellent introductory discussion of these difficulties and the methods developed to deal with them can be found in Leveque's book [8].

The primary goal of the ENO effort has been to develop a general purpose numerical method for systems of conservation laws that has high accuracy (at least third order) in smooth regions and captures the motion of unresolved steep gradients in the flow, without creating spurious oscillations and without the use of problem dependent fixes or tunable parameters. An additional priority has been to formulate the methods within a systematic mathematical framework.

The philosophy underlying the ENO methods is simple: when reconstructing a profile for use in a convective flux term, one should not use high order polynomial interpolation across a steep gradient in the data. Such an interpolant would be highly oscillatory and ultimately corrupt the computed solution. ENO methods use an adaptive polynomial interpolation constructed to avoid steep gradients in the data. The polynomial is also biased to extrapolate from data from the direction of information propagation—"upwind"— for physical consistency and stability. In the case of a system, this interpolation must be done in the local characteristic fields, since it is these quantities—not the primitive conserved variables such as mass, momentum and energy—that are properly thought of as propagating in various directions. The ENO approach is completed by combining this interpolation method with a discrete conservation form for the equations. This form insures that shocks and other steep gradients in the flow are "captured", i.e. move at the right speed even if they are not fully resolved.

The original ENO schemes were based on the conservative control volume discretization of the equations, which yields discrete evolution equations for grid cell averages of the conserved quantities, e.g. mass, momentum and energy. This formulation has the disadvantage of requiring complicated transfers between cell averages and cell center nodal values in the algorithm. In particular, the transfer process becomes progressively more complicated in one, two and three spatial dimensions. The formulation also results in space and time discretizations that are coupled in a way that becomes complicated for higher order accurate versions.

To eliminate these complications, Shu and Osher [12] developed a conservative finite difference form of the ENO method, which uses only nodal values of the conserved variables. Their method is faster and easier to implement than the cell averaged formulation. In addition, the finite difference ENO method extends to higher dimensions in a "dimension by dimension" fashion, so that the 1D method applies unchanged to higher dimensional problems. They also use the method of lines for time integration, which decouples the time and space discretizations. To complete the scheme, Shu and Osher developed a special family of Runge-Kutta time integration schemes that are easy to implement, have good stability properties, and also have a "Total Variation Diminishing" (TVD) property. The TVD property prevents the time stepping scheme from introducing spurious spatial oscillations into upwind-biased spatial discretizations. We emphasize that this is *not* dimensional splitting in time, which has accuracy limitations unlike the "dimension by dimension" approach.

While both the cell averaged and finite difference formulations of ENO perform well on a great variety of compressible flow calculations, there are still special circumstances under which they produce spurious oscillatory results. Some of these situations are well known, such as the case of a slow moving shock. In this case, the cause of the oscillations is largely understood, but this has not resulted in a general, elegant way to eliminate the problem. In other cases, such as the examples provided in section 8, the cause of the oscillations is not understood due to the complexity of the physical problem.

It is apparent now—based mainly on the work of Donat and Marquina [1], as well as a model problem described in [2]—that the manner in which the transformation to local characteristic variables is evaluated within the cell wall flux calculation is responsible for these occasional spurious oscillations. In particular, the problem is due to evaluating the transformation—or, equivalently, the flux Jacobian matrix—at a cell wall that separates two very different states. The common approach in all characteristic-based methods is to evaluate this transformation at some reasonable average of the adjacent states. However, there is clearly a great deal of ambiguity in choosing this average, and any particular choice seems to introduce subtle spurious features into the solution. To avoid this ambiguity, Marquina introduced a flux splitting technique based on the unambiguous data on the left and right sides of the cell wall. There, the transformations to characteristic variables and subsequent flux calculations are well defined, Marquina combines the results in an upwind fashion to determine the cell wall flux. The details of these old and new approaches are described in section 2.7.

When the Marquina's splitting technique is applied to the standard ENO flux calculation, it fixes all known problematic cases. Thus the resulting finite difference ENO method with Marquina's splitting seems to meet the original goal of an elegant, general, accurate, robust, parameter-free method for hyperbolic systems of conservation laws. If this turns out to be the case, it may be the ultimate conservative difference scheme [3, 4, 5, 6, 7]. Since only further experience can determine its limitations, for now we propose it as the penultimate method. In any case, it is an important enhancement of the original ENO method, and should replace it for future applications.

Our primary goal here is to present—in a self-contained, accessible form—this new hybrid method consisting of Shu-Osher finite difference ENO with Marquina's flux splitting technique. We hope this will encourage widespread application of this technique.

This paper divides naturally into two parts. The first part is a tutorial on scheme design for hyperbolic systems of conservation laws, and is directed mainly at those not familiar with this field. The goal is to motivate the many details that go into the final scheme design described in the second part. The second part of the paper is a users' guide for the preferred form of the new method.

The first part provides the conceptual background needed to appreciate characteristic based methods for systems of convective conservation laws. This includes both the basic ingredients that go into the numerical method design, such as CFL restriction and shock capturing, as well as the advanced issue of conservative finite difference discretization, the upwind biased ENO interpolation technique, and the Jacobian evaluation problem that motivates Marquina's splitting and distinguishes it from previous practice.

In the second part, we start by showing how this method fits in as part of a comprehensive space and time discretization that can handle general systems of conservation laws that arise in physical problems. Next, we present the preferred form of the new method in a concise, detailed fashion suitable for direct application in numerical calculations. We finish with a few new examples illustrating the effectiveness of this approach.

2 Background and Motivation

In order to make this presentation self-contained, we provide some conceptual motivation and background for the various ingredients used in the new scheme. We will motivate the use of characteristic based schemes, discuss the form of the finite difference discretization used in the ENO methods, and discuss ENO interpolation. We then contrast Marquina's splitting with the traditional approach used in characteristic based schemes.

A general introduction to the properties of systems of conservation laws and their associated numerical methods can be found in LeVeque's book [8]. Comments on the original motivation and development of finite difference ENO can be found in the first paper of Shu and Osher, [11]. Marquina's splitting is motivated and introduced in the papers of Donat and Marquina, [1, 9].

Our goal here is to introduce convective—or hyperbolic—systems of conservation laws, and understand how their fundamental features impact the design of appropriate numerical methods.

2.1 General Conservation Laws

A continuum physical system is described by the laws of conservation of mass, momentum, and energy. That is, for each conserved quantity, the rate of change of the total amount in some region is given by its flux (convective or diffusive) through the region boundary, plus whatever internal sources exist. The integral form of this conservation law is

$$\frac{d}{dt}\int_R U dV + \int_{\partial R} \vec{F}(U)\cdot dA = \int_R S(U) dV \tag{1}$$

where U is the density of the conserved quantity, $\vec{F}(U)$ is the flux, and $S(U)$ is the source rate, and the volume and surface integrals indicated are over the region R and its boundary ∂R. By taking R to be an infinitesimal volume and applying the divergence theorem, we get the differential form of the conservation law,

$$\frac{\partial U}{\partial t} + \nabla\cdot\vec{F}(U) = S(U) \tag{2}$$

which is the basis for the numerical modeling of all continuum systems. Any physical system will be described by a system of such equations, i.e. a system of conservation laws. These also form the basis for their numerical modeling.

We will write most equations in one dimension, in which case our notation for the differential conservation equation 2 takes the more compact form:

$$U_t + F(U)_x = S(U) \tag{3}$$

2.2 Convective (Hyperbolic) Conservation Laws

A conserved quantity, such as mass, can be transported by convective or diffusive fluxes. The distinction is that diffusive fluxes are driven by gradients in density, while convective fluxes persist even in the absence of gradients. Here we will concentrate on the convective transport, ignoring diffusion (mass diffusion, viscosity and thermal conductivity) and also the source terms (such as chemical reactions, atomic excitations, and ionization processes). We take this simplified approach because the convective transport requires specialized numerical treatment. If present, diffusive and reactive effects can be treated by standard numerical methods that are independent of those for the convective terms. Stiff reactions, however, do present numerical difficulties.

Conservation laws with only convective fluxes are known as "hyperbolic" conservation laws (a more careful definition is given is section 2.5). A vast array of physical phenomena are modeled by such systems. The physics of explosives and high speed aircraft were two major driving forces in the development of these models. They also provide the basis for modeling astrophysical and fusion reactor plasma, mixed phase flow in fission reactor cooling systems, and combustion in jet engines, to mention a few of the important technological applications.

2.3 Convective Phenomena, Models and Numerical Implications

Our goal here is to mention the most universal aspects of the physics of hyperbolic systems, and relate it to the design of appropriate numerical methods.

The important physical phenomena exhibited by convective conservation laws are bulk convection, waves, contact discontinuities, shocks, and rarefactions. We will briefly describe the physical features and mathematical model equations for each effect, and most importantly, note the implications they have for numerical method design.

Bulk Convection and Waves Bulk convection is simply the bulk movement of matter—carrying it from one spot to another, like water streaming from a hose. Waves are small amplitude smooth rippling disturbances that transmit through the system without any bulk transport—like ripples on a water surface or sound waves through air. Whereas convective transport occurs at the gross velocity of the material, waves propagate at the "speed of sound" in the system (relative to the bulk convective motion of the system). Waves interact by superposition, so that they can either cancel out (interfere) or enhance each other.

The simplest model equation that describes bulk convective transport is the linear convection equation

$$\rho_t + v\rho_x = 0, \tag{4}$$

where v is a constant, equal to the convection velocity. This has the equivalent, but less often used, conservation form

$$\rho_t + (v\rho)_x = 0, \tag{5}$$

The solution to this is simply that ρ translates at the constant speed v. This same equation can also be taken as a simple model of wave motion, if ρ is a sine wave and v is interpreted as the sound speed.

The linear convection equation is also an important model for understanding smooth transport in any conservation law: as long as U has no jumps in it, and F is smooth, the general law $U_t + F(U)_x = 0$ can be expanded to

$$U_t + vU_x = 0, \tag{6}$$

where $v = F'(U)$. Thus, locally in smooth parts of the flow, any conservation law behaves like bulk convection with convective velocity $F'(U)$. This is called the characteristic velocity of the flow.

Bulk convection and waves are important because they imply that signals propagate in definite directions at definite speeds. This in contrast to a phenomena like diffusion which propagates signals in all directions at arbitrarily large speeds depending on the severity of the driving gradients. Thus we anticipate suitable numerical methods for hyperbolic systems will also have directional biases in space—which leads to the idea of upwind differencing, see section 2.4—and a definite relation between the space and time step (discrete propagation speed)—which will roughly be that the discrete propagation speed $\Delta x/\Delta t$ must be the same as the physical propagation speeds (characteristic speeds) in the problem. The general form of this relation is called the Courant-Friedrichs-Lewy (CFL) restriction, and it says the discrete speed must be at least as large as any characteristic speed in the problem.

Also, note that wave motion and bulk convection don't create any new sharp features in the flow. The other remaining phenomena are all special because they involve discontinuous jumps in the transported quantities. Because smooth features can be accurately represented by a polynomial interpolation, we expect to be able to develop extremely high accuracy numerical methods for the wave and convective effects. Conversely, since jump functions are poorly represented by polynomials, we expect little accuracy and perhaps great difficulty in numerically approximating the discontinuous phenomena.

The linear convection model also has an important implication for the time integration numerical method, i.e. the numerical method used to discretize

$\partial U/\partial t$. If we Fourier transform the linear advection equation we end up with an ordinary differential equation needing only time integration:

$$\hat{\rho}_t - ikv\hat{\rho} = 0, \tag{7}$$

where $\hat{\rho}(k)$ is the Fourier transform of $\rho(x)$. The important thing to note is that this is an ODE of the form $y_t = \lambda y$, where the growth rate λ is purely imaginary. Thus we must use an ODE integration method that is stable for imaginary growth rates. This is true for standard third order and fourth order Runge-Kutta methods, for example. But it is not true for common first order ("explicit Euler") or second order ("Heun's method") Runge-Kutta schemes. *Use of these common low order schemes is not compatible with an accurate spatial discretization of convection.* That is, using these methods with standard hyperbolic spatial discretizations would lead to the development of severe grid point to grid point oscillations, due solely to the poor choice of time stepping procedure. For explicit time stepping stability, a third or fourth order Runge-Kutta method should be used.

Contacts A contact discontinuity is a persistent, discontinuous jump in mass density moving by bulk convection through the system. Since there is negligible mass diffusion, such a jump persists. These jumps usually appear at the point of contact of different materials, for example, a contact discontinuity separates oil from water. Contacts move at the local bulk convection speed, or more generally, the characteristic speed, and can be modeled by using step-function initial data in the bulk convection equation 4. Since contacts are simply a bulk convection effect, they retain any perturbations they receive. Thus we expect contacts to be especially sensitive to numerical methods—any spurious alteration of the contact will tend to persist and accumulate.

Shocks A shock is a spatial jump in material properties—like pressure and temperature—that develops spontaneously from smooth distributions and then persists. That is, the shock jump is self-forming and also self-maintaining. This is unlike a contact, which must be put in the system initially, and will not re-sharpen itself if it is smeared out by some other process. Shocks develop through a feedback mechanism in which strong impulses move faster than weak ones, and thus tend to steepen themselves up into a "step" profile as they travel through the system. Familiar examples are the "sonic boom" of a jet aircraft, or the "bang" from a gun. These sounds are our perceptions of a sudden jump in air pressure.

The simplest model equation that describes shock formation is Burgers' equation

$$u_t + (\frac{u^2}{2})_x = 0. \tag{8}$$

Formally, this looks like the convection equation 4, with a non-constant convective speed of $v = u$. Thus larger u values move faster, and they will overtake smaller values, ultimately resulting in the development of a right-going shock if the initial data for u is any positive, decreasing function, e.g. $1 - \tanh(x)$.

Shocks move at a speed that is not simply related to the bulk flow speed or characteristic speed, and is not immediately evident from examining the flux, in contrast to contacts. Shock speed is controlled simply by the difference between influx and outflux of conserved quantity into the region. Specifically, suppose a conserved quantity U with conservation law 3 has a step function profile with one constant value extending to the left, U_L, and a lower constant value to the right, U_R, with a single shock jump transition between these two, and this jump location is moving with speed s to the right. Then the integral form of the conservation law 1, applied to any interval containing the shock, gives the relation

$$s(U_R - U_L) = F_R - F_L, \tag{9}$$

which is, of course, just another statement that the rate at which U appears, $s(U_R - U_L)$, in the interval of interest is given by the difference in fluxes across the interval. However, it also determines the shock speed s in terms of densities and fluxes well away from the shock itself.

Thus we see that the proper speed of the shock is directly determined by—and only by—conservation of U via the flux F. This has an important implication for numerical method design: namely, a numerical method will only "capture" the correct shock speeds if it has "conservation form", i.e. if the rate of change of U at some node is the difference of fluxes which are accurate approximations of the real flux F.

The self-sharpening feature of shocks has two implications for numerical methods. First, it means that even if the initial data is smooth, steep gradients and jumps will form spontaneously; thus our numerical method must be prepared to deal with shocks even if none are present in the initial data. Second, there is a beneficial effect from self-sharpening, because modest numerical errors introduced near a shock (smearing or small oscillations) will tend to be eliminated, and will not accumulate. The shock is naturally driven towards its proper shape. Because of this, computing strong shocks is mostly a matter of having a conservative scheme in order to get their speed correct—the basic jump itself will be preserved by the physical self-sharpening.

Rarefactions A rarefaction is a discontinuous jump or steep gradient in properties that dissipates as a smooth expansion. A common example is the jump in air pressure from outside to inside a balloon, which dissipates as soon as the balloon is burst and the high pressure gas inside is allowed to expand. Such an expansion also occurs when the piston in an engine is rapidly pulled outward from the cylinder. The expansion (density drop) associated with a rarefaction propagates outward at the sound speed of the system, relative to the underlying bulk convection speed.

A rarefaction can be modeled by Burgers' equation 8, with initial data that starts out as a steep increasing step, for example $u(x) = \tanh(\frac{x}{\epsilon})$, where ϵ is a small (perhaps 0) width for the step. This step will broaden and smooth out during the evolution.

A rarefaction tends to smooth out local features, which is somewhat good for numerical modeling. It tends to diminish numerical errors over time and make the solution easier to represent by polynomials, which form the basis for our numerical representation. However, a rarefaction often connects to a smooth (e.g. constant) solution region and this results in a "corner" which is notoriously difficult to capture accurately.

The main numerical problem posed by rarefactions is that of initiating the expansion. If the initial data is a perfect, symmetrical step, such as $u(x) = \mathrm{sign}(x)$, it may be "stuck" in this form, since the steady state Burgers' equation is satisfied identically (i.e. the flux $\frac{u^2}{2}$ is constant everywhere, and similarly in any numerical discretization). However, local analysis can identify this stuck expansion, because the characteristic speed u on either side points away from the jump, suggesting its potential to expand. In order to get the initial data unstuck, some small amount of smoothing must be applied to introduce some intermediate state values and thus have a non-constant flux to drive expansion. In numerical methods, this smoothing applied at a jump where the effective local velocity indicates expansion should occur is called an "entropy fix", since it allows the system to evolve from the artificial low entropy (i.e. very symmetrical) initial state to the proper increased entropy state of a free expansion.

Systems of Equations In general, a hyperbolic system will simultaneously contain all these processes: smooth processes of bulk convection and wave motion, and discontinuous processes involving contacts, shocks and rarefactions. For example, if a gas in a tube is initially prepared with a jump in the states (density, velocity and temperature) across some surface, as the evolution proceeds in time these jumps will break up into a combination of shocks, rarefactions and contacts, in addition to any bulk motion and sound waves that may exist or develop.

Based on these considerations, in a general system we expect that the density of mass, momentum and energy will be smooth in large regions, separated by these discontinuous jumps in properties. Further, these jumps are moving through the system, interacting in complex ways. What we can generally hope for is numerical methods that are high accuracy in the smooth regions, don't distort the jumps too much (smear them or add oscillations), and move the jumps at the correct speeds, which are usually not known a priori. We want high accuracy methods to effectively model smooth convection and wave motion. We expect contacts to be the most sensitive indicators of numerical errors on discontinuities. We expect the shocks to be robust features, and we expect rarefactions to not be a problem as long as their initial expansion from a jump can be made to occur.

The simplest system of *physically realistic* model equations for convective transport is the Euler equations for gas dynamics, which describe the conservative transport of mass, momentum and energy in a gas in one spatial dimension (e.g. in a long tube):

$$\rho_t + (\rho v)_x = 0 \tag{10}$$
$$(\rho v)_t + (\rho v^2 + p)_x = 0 \tag{11}$$
$$E_t + ((E + p)v)_x = 0 \tag{12}$$

where ρ is the mass density, v is the flow velocity, p is the pressure and E is the total energy (kinetic plus internal) density. To be completely specified, these equations require an "equation of state" for the pressure, i.e. a relation $p = p(\rho, E)$. One of the simplest reasonable forms is the gamma law gas relation, $p = p_0(\rho/\rho_0)^\gamma$, where $\gamma \geq 1$ is a constant and p_0, ρ_0 are the reference pressure and density.

Despite their simple form—looking like linear convection and Burgers' equations—the Euler equations support extremely complex dynamic behavior which can be difficult to understand and predict, due to the nonlinear, coupled form of the equations. However, it is true that any isolated jump discontinuity in the state variables will, as time goes on, break up into some "combination" of a shock, a contact and a rarefaction. This justifies a simple intuitive model for the structure of this system: a "toy" version of the Euler equations consists of three independent scalar equations: one convection (for representing the effects of bulk convection, waves and contacts), one Burgers' equation (for shock formation) and another independent Burgers' equation (for independent rarefaction formation):

$$(u_1)_t + (u_1)_x = 0 \tag{13}$$
$$(u_2)_t + \left(\frac{u_2^2}{2}\right)_x = 0 \tag{14}$$
$$(u_3)_t + \left(\frac{u_3^2}{2}\right)_x = 0 \tag{15}$$

where the initial data are step functions with jumps at $x = 0$ as follows: an arbitrary jump in u_1 (a contact, moving right at speed 1), a decreasing jump in u_2 (a shock, or shock precursor if smoothed out slightly), and an increasing jump in u_3 (a rarefaction). Since these three equations are independent, the subsequent evolution is obvious. However, let us form a new, equivalent system by multiplying this system by a 3×3 invertible matrix R. If the original system in vector form is written as

$$\vec{U}_t + [\vec{F}(\vec{U})]_x = 0 \tag{16}$$

then the new system can be written as

$$\vec{V}_t + [\vec{G}(\vec{V})]_x = 0 \tag{17}$$

where $\vec{V} = R\vec{U}, \vec{G} = R\vec{F}$. When considered in terms of the "mixed" variables $\vec{V} = (v_1, v_2, v_3)$, the behavior of this system is not at all obvious, and the simple contact, shock and rarefaction present in the system will cause a rather complicated evolution of $\vec{V}$. The intuitive point to understand is that the real Euler equations, as well as other hyperbolic systems we encounter in physical problems, are written in what are effectively the mixed variables, where the apparent behavior is quite complicated. It requires some transformation to decouple them back into unmixed fields that exhibit the pure contact, shock and rarefaction phenomena (as well as bulk convection and waves). In this toy model here, there is a single linear transformation that perfectly decouples the mixed equations, namely the inverse R^{-1}. In a real system, this perfect decoupling is not possible because the mixing is nonlinear, but it can be achieved approximately—over a small space and time region—and this provides the basis for the theoretical understanding of the structure of general hyperbolic systems of conservation laws. This is called a transformation to characteristic variables, and we will present it in detail in section 2.5. As we shall see, this transformation also provides the basis for designing appropriate numerical methods.

Summary of Numerical Implications As we have examined the properties of hyperbolic systems, we have compiled a list of associated implications for numerical method design. For clarity, we will summarize these here.

- **CFL Condition** to correctly model the propagation of information, the space and time grids must satisfy $\Delta x/\Delta t \geq s_{\max}$, where $s_{\max}$ is the largest propagation speed (characteristic speed) in the problem.
- **Upwind Biasing** the directed propagation of signals implies there will be directional biases in the choice of spatial nodes—in the "upwind direction"—used to discretize the equations.
- **High Accuracy** modeling of weak wave propagation and smooth convection benefits greatly from numerical methods based on high order accurate polynomial interpolation.
- **Time Integration** A third or fourth order accurate Runge-Kutta method (or other method stable for imaginary growth rates) must be used for the numerical time integration, to avoid instability.
- **Entropy Fix** proper numerical modeling of rarefactions require a small amount of smoothing of a jump where the nearby characteristic speeds indicates potential for expansion.
- **Sharp Contacts** the most sensitive indicator of how well a numerical method handles jumps is the treatment of contacts discontinuities (in a linear convection equation).

- **Conservation Form** in order to capture shock speeds, the numerical method must have conservation form, i.e. be written in terms of a discrete difference in fluxes.

- **Characteristic Decomposition** systems of conservation laws can be best understood by transforming to local characteristic variables that display largely decoupled scalar behavior.

By addressing all these points, we can design methods that accurately and efficiently compute the behavior of hyperbolic systems of conservations laws, or the hyperbolic parts of general systems of conservation laws.

2.4 Upwind Biased ENO Interpolation

Here we give a more detailed motivation for upwind biased discretization, and the Essentially Non-Oscillatory (ENO) interpolation technique that forms the basis for our numerical methods.

As noted in the summary of section 2.2, to assess the quality of our numerical method we can focus on the treatment of contact discontinuities in the linear convection equation 4. Since the time discretization can be handled by a high accuracy Runge-Kutta method, we will focus on the spatial discretization and assume the time evolution takes place exactly—i.e. at each time step Δt, the spatial profile just translates rigidly by the amount $v\Delta t$.

Spatially, the contact is initially represented by a discrete step function, i.e. nodal values that are constant at one value ρ_L on nodes $x_1, \dots, x_J$, and then constant at a different value, ρ_R, at all remaining nodes $x_{J+1}, \dots, x_N$.

To update in time the value ρ_i at a given node x_i, we first reconstruct the graph of a function $\rho(x)$ near x_i by interpolating nearby nodal ρ values, shift that $\rho(x)$ graph spatially by $v\Delta t$ (the exact time evolution), and then reevaluate it at the node x_i to obtain the updated ρ_i. We require our local interpolant be smooth at the point x_i, since in actual practice we are going to use it to evaluate the derivative term $(v\rho)_x$ there.

The simplest symmetrical approach to smooth interpolation near a node x_i is to run a parabola through the nodal data at x_{i-1}, x_i, x_{i+1}. This interpolation is an accurate reconstruction of $\rho(x)$ in smooth regions, and this approach will work well there. However, near the jump, at x_J and x_{J+1}, the parabola will greatly overshoot the nodal ρ data itself, by an amount comparable to the jump $\rho_L - \rho_R$, and this overshoot will show up in the nodal values once the shift is performed. Successive time steps will further enhance these spurious oscillations. In this way, repeated parabolic interpolation and shifting introduces severe oscillations that totally destroy the structure of the contact. This approach corresponds to standard central differencing applied to the convection equation 4.

To avoid the oscillations from parabolic interpolation, we could instead try to use a smooth linear interpolation near x_i. However, there are two to choose from, namely the line through data at nodes x_i and x_{i-1}, or through data at x_i

and x_{i+1}. The direction of information propagation determines which one will result in a non-oscillatory reconstruction. Assuming the convection speed v is positive, the data is moving from the left to the right. Thus, for a short time, the only ρ values that will arrive at node x_i in the exact solution are those over the interval (x_{i-1}, x_i). If we use a linear interpolation based on these two upwind nodes, when we shift it right by $v\Delta t$ we will not introduce any new extrema in ρ at x_i, since the result will lie between ρ_{i-1} and ρ_i (as long as the shift $v\Delta t$ is less than the width of the interval $\Delta x = x_i - x_{i-1}$, which is exactly the CFL restriction on the time step). In contrast, if the linear interpolation were based on the "downwind" nodes (x_i, x_{i+1}), a shift right would cause a part of this line *not* between ρ_i and ρ_{i+1} appear as the new ρ value at x_i, and this can and will introduce new, spurious extreme values and oscillations into the nodal data. In this way, we see clearly why it necessary to base the linear interpolant on the upwind point x_{i-1}: interpolating from that direction represents the data that is supposed to arrive at the point of interest, so that no spurious values are introduced.

The main problem with the linear upwind biased interpolant is that it has low accuracy. Each interpolate and shift step will smear out the contact jump over more nodes. If we naively go to higher accuracy by using a higher order upwind biased interpolant, such as running a parabola through x_i, x_{i-1}, x_{i-2} to advance ρ_i, we will run into the spurious oscillation problem again—at nodes x_{J+1} and x_{J+2} this upwind parabola will interpolate across the jump and thus have large overshoots just as for the centrally interpolated parabolas. By forcing the parabola to cross a jump, it no longer reflects the data on the interval (x_i, x_{i-1}) that will be arriving at x_i during the next time step.

A solution to the problem of achieving more accuracy while avoiding spurious overshoots in the interpolant is to use the upwind biased, Essentially Non-Oscillatory (ENO) interpolation technique [11, 12]. The motivation for this approach is that we must use a higher degree polynomial interpolant to achieve more accuracy, and it must involve the immediate upwind node to properly represent the propagation of data. But, as we saw, we must also avoid polluting this upwind data with spurious oscillations that come from interpolating across jumps in data. Thus the remaining interpolation nodes are chosen based on smoothness considerations. Specifically, to update ρ_i using a degree k interpolant requires $k+1$ interpolation nodes. We will choose $k+1$ consecutive nodes that include the immediate upwind node from x_i, which is x_{i-1} if $v > 0$. Still, that leaves k different lists of nodes—"stencils"—to choose from. Of these, we will use the one for which the resulting interpolating polynomial is the smoothest, by some measure. (For example, we can measure smoothness by the size of the k^{th} derivative, or the total variation, or by any other convenient means. For more detailed considerations see [10].) In particular, this approach will—if at all possible—not run an interpolant across a jump in the data. Thus, it avoids introducing large, spurious overshoots. However very small interpolation overshoots do occur near extrema in the nodal data, as they must, since any smooth function will slightly overshoot its values as sampled at discrete points near extrema. This is the sense in which the method is only *Essentially* Non-Oscillatory (ENO)—it is not a failing; it simply reflects the real relation between smooth functions and their discretely sampled values.

In practice, there are simple, efficient ways to generate the upwind biased ENO interpolant of any desired order, based on the divided difference table of the nodal data.

2.5 Characteristic Based Schemes for Hyperbolic Systems

In this section we describe the use of characteristic decomposition for designing suitable upwind biased numerical schemes.

Consider a system of N convective conservation laws in one spatial dimension,

$$\vec{U}_t + [\vec{\mathcal{F}}(\vec{U})]_x = 0 \tag{18}$$

The basic idea of characteristic numerical schemes is to transform this nonlinear system to a system of (nearly) independent scalar equations of the form

$$u_t + vu_x = 0 \tag{19}$$

discretize each scalar equation independently in an v-upwind biased fashion, and then transform the discretized system back into the original variables.

In a smooth region of the flow, we can get a better understanding of the structure of the system by expanding out the derivative as

$$\vec{U}_t + J\vec{U}_x = 0 \tag{20}$$

where $J = \frac{\partial \vec{\mathcal{F}}}{\partial \vec{U}}$ is the Jacobian matrix of the convective flux function. Note that if J were a diagonal matrix, with real diagonal elements, this system would be decoupled into N independent scalar equations as desired.

In general J is not of this form, but we can hope to transform this system to that form by multiplying through by a matrix that diagonalizes J. If this is possible, we call the system *hyperbolic*. The fortunate thing is that most physical convective transport equations turn out to be "hyperbolic". In this case, the necessary matrices turn out to be the matrices of left-multiplying and right-multiplying eigenvectors of J. Specifically, for a hyperbolic system we require following properties (which allow our strategy to work): first, we require that J have N real eigenvalues $\lambda^p, p = 1, \ldots, N$, and that there be N eigenvectors for multiplying against J from the right. If we use these as columns of a matrix R, this is expressed by the matrix equation

$$JR = RDiag(\lambda^p) \tag{21}$$

where $Diag(\lambda^p)$ denotes a diagonal matrix with the elements $\lambda^p, p = 1, \ldots, N$ on the diagonal. Similarly, we also require that there be N eigenvectors for multiplying against J from the left; when these are used as the rows of a matrix L, this is expressed by the matrix equation

$$LJ = Diag(\lambda^p)L \tag{22}$$

We finally require that these matrices L and R can be chosen to be inverses

$$LR = RL = I \tag{23}$$

These matrices transform to a system of coordinates in which J is diagonalized as desired:

$$LJR = Diag(\lambda^p) \tag{24}$$

Suppose we want to discretize our equation at the node x_0, where L and R have values L_0 and R_0. To get a locally diagonalized form, we multiply our system equation by the *constant* matrix L_0 which nearly diagonalizes J over the region near x_0 (we require a constant matrix so that we can move it inside all derivatives):

$$[L_0\vec{U}]_t + L_0JR_0[L_0\vec{U}]_x = 0 \tag{25}$$

We have inserted $I = R_0L_0$ to put the equation in a more recognizable form. The spatially varying matrix L_0JR_0 is exactly diagonalized at the point x_0, with eigenvalues λ_0^p, and it is nearly diagonalized at nearby points. Thus the equations are sufficiently decoupled for us to apply upwind biased discretizations independently to each component, with λ_0^p determining the upwind biased direction for the p-th component equation. Once this system is fully discretized, we multiply the entire system by $L_0^{-1} = R_0$ to return to the original variables.

In terms of our original equation 18, our procedure for discretizing at a point x_0 is simply to multiply the entire system by the left eigenvector matrix L_0,

$$[L_0\vec{U}]_t + [L_0\vec{\mathcal{F}}(\vec{U})]_x = 0 \tag{26}$$

and discretize the $p = 1, \ldots, N$ scalar components of this system

$$[(L_0\vec{U})_p]_t + [(L_0\vec{\mathcal{F}}(\vec{U}))_p]_x = 0 \tag{27}$$

independently, using upwind biased differencing with the upwind direction for the p-th equation determined by the sign of λ^p. We then multiply the resulting spatially discretized system of equations by R_0 to recover the spatially discretized fluxes for the the original variables:

$$\vec{U}_t + R_0\Delta(L_0\vec{\mathcal{F}}(\vec{U})) = 0 \tag{28}$$

where Δ stands for the upwind biased discretization operator.

We call λ^p the p-th characteristic velocity or speed, $(L_0\vec{U})_p = \vec{L}_0^p \cdot \vec{U}$ the p-th characteristic state or field (here L^p denotes the p-th row of L, i.e. the p-th left eigenvector of J), and $(L_0\vec{\mathcal{F}}(\vec{U}))_p = \vec{L}_0^p \cdot \vec{\mathcal{F}}(\vec{U})$ the p-th characteristic flux. According to the local linearization, it is approximately true the p-th characteristic field rigidly translates in space at the p-th characteristic velocity. Thus this decomposition corresponds to the local physical propagation of independent "waves" or "signals".

2.6 The Conservative Finite Difference Form

To ensure that shocks and other steep gradients are captured by the scheme—i.e. they move at the right speed even if they are unresolved—we must write the equation in a discrete conservation form. That is, a form in which the rate of change of conserved quantities is equal to a difference of fluxes. This form guarantees that we conserve the total amount of the states $\vec{U}$ (e.g. mass, momentum and energy) present, in analogy with the integral form given by equation 1. More importantly, this can be shown to imply that steep gradients or jumps in the discrete profiles must propagate at the physically correct speeds [8] as discussed in section 2.3.

Usually, conservation form is derived for *control volume* methods, that is methods that evolve cell average values in time rather than nodal values. In this approach, a grid node x_i is assumed to be the center of a grid cell $(x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}})$, and we integrate the conservation law 3 across this control volume to obtain (set the source to 0 for simplicity)

$$\bar{U}_t + (F(U_{i+\frac{1}{2}}) - F(U_{i-\frac{1}{2}})) = 0 \tag{29}$$

where $\bar{U}$ is the integral of U over the cell, and $U_{i\pm\frac{1}{2}}$ are the (unknown) values of U at the cell walls. This has the desired conservation form, in that the rate of change of the cell average is a difference of fluxes. The difficulty with this formulation is that it requires transforming between cell averages of U (which are directly evolved in time by the scheme) and cell wall values of U (which must be reconstructed) to evaluate the needed fluxes. While this is manageable in 1-D, in higher dimensional problems the series of transformations necessary to convert the cell averages to cell wall quantities becomes increasingly complicated. The distinction between cell average and midpoint values is usually ignored for schemes whose accuracy is no higher than second order (e.g. TVD schemes). This is because the cell average and the midpoint value differ by $O(\Delta x^2)$.

Instead, we seek a fully finite difference scheme—i.e. a scheme that directly evolves nodal values in time. For the finite difference approach, the derivation of conservation form is less obvious. We define the "numerical flux function", $\vec{F}$, by the property that the real flux divergence is a finite difference of numerical fluxes:

$$\vec{\mathcal{F}}(\vec{U})_x = \frac{\vec{F}(x + \frac{\Delta x}{2}) - \vec{F}(x - \frac{\Delta x}{2})}{\Delta x} \tag{30}$$

at every x (here Δx is some constant spacing). We call it the numerical flux since we require it in our numerical scheme, and also to distinguish it from the closely related "physical flux", $\vec{\mathcal{F}}(\vec{U})$. It is not obvious that the numerical flux function exists, but from relationship 30 one can solve for its Taylor expansion (or, using a Fourier transform gives a quick derivation). The result is

$$\vec{F} = \vec{\mathcal{F}}(\vec{U}) - \frac{(\Delta x)^2}{24}\vec{\mathcal{F}}(\vec{U})_{xx} + \frac{7(\Delta x)^4}{5760}\vec{\mathcal{F}}(\vec{U})_{xxxx} - \cdots \tag{31}$$

Note that to second order accuracy in Δx the physical and numerical flux functions are the same. As described in section 6.2, direct use of the Taylor series is not the most convenient way to compute the numerical flux in the ENO algorithm. The series is simply useful for understanding the relation between physical and numerical fluxes.

The finite difference discretization is not based directly on the differential form of the conservation law 18; rather, it is based on the equivalent conservative finite difference form

$$\vec{U}_t + \frac{\vec{F}(x+\frac{\Delta x}{2}) - \vec{F}(x-\frac{\Delta x}{2})}{\Delta x} = 0 \tag{32}$$

The discretization is based on characteristic upwind differencing, but now it is the numerical flux $\vec{F}$ that must be discretely approximated, rather than the physical flux derivative, $\vec{\mathcal{F}}(\vec{U})_x$. To accomplish this, we generalize the conclusions of the previous characteristic variables analysis to the following procedure: to determine $\vec{F}$ at a point x_0, we should multiply all the local nodal physical flux vectors by L_0, and then use these, component by component, to construct scalar characteristic numerical fluxes in an upwind biased fashion. We then project these back to original variable numerical fluxes by multiplying by R_0.

2.7 Comments on Jacobian Evaluation

We will briefly outline the significance of the Jacobian evaluation for characteristic based methods, and how Marquina's procedure differs from the evaluation commonly used in the ENO method [11, 12].

The Jacobian matrix of the convective flux vector is quite important to any characteristic based scheme, as it defines the local linearization of the nonlinear problem. As previously described, it determines the transformation to the local characteristic fields, and thus what the upwind directions are as well as what quantities are to be upwind differenced.

In finite volume methods it is natural that the fluxes—and thus the transformation to characteristic fields needed to evaluate them in an upwind way—be evaluated at cell walls. The analogous situation occurs in the conservative finite difference formulation as well. There we want to discretely approximate the conservation equation, 18, at grid nodes, x_i. Thus, by the numerical flux relation, 30, we require values of the *numerical flux* at the midpoint between nodes, $x_0 = x_{i+\frac{1}{2}}$. We refer to these midpoints as "cell walls", in analogy with the finite volume case.

Thus, in order to transform to characteristic fields to evaluate the numerical fluxes, we require values of the Jacobian (and its eigensystem) at cell walls. In the finite difference setting, we only know values for $\vec{U}$ at the nodes, so evaluation of a Jacobian at the cell walls requires some form of interpolation. In standard ENO schemes it was thought that the precise form of this interpolation was not so important. But recent developments show that in fact it can make a great deal of difference in causing or eliminating spurious oscillations.

The standard ENO method uses a single Jacobian evaluated at the linear average of the states at nodes adjacent to the midpoint,

$$J_{i+\frac{1}{2}} = J\left(\frac{\vec{U}_i + \vec{U}_{i+1}}{2}\right) \tag{33}$$

In smooth regions, this centered linear approximation is second order accurate. Moreover, in a smooth region it makes little difference whether the derivatives are computed in an upwind biased fashion or in some combination of upwind and downwind. Thus the precise determination of the Jacobian (and the transformation to characteristic fields)—in addition to having little uncertainty anyway—is not so important. It is between nodes in an unresolved steep gradient that the centrally averaged Jacobian might cause problems. It can differ significantly from the left and right Jacobians interpolated from left and right nodal state values, and there is no clear reason why this central Jacobian value is the proper choice for a midpoint Jacobian. The only justification for its use is that in practice it seems to work well for many problems. However, based on the following considerations we can see that it has the *potential* to allow spurious oscillations under special circumstances.

In the Separating Box Problem [2], we showed that small perturbations to the Jacobian matrix can lead to large oscillations in an ENO numerical solution. The intuition developed there was that small errors in the Jacobian would cause one to transform into the wrong characteristic variables, i.e. ones which were mixtures of the true characteristics. Upwinding on these slightly mixed fields amounts to a small amount of downwind differencing on the "true" characteristic fields, combined with the desired upwind differencing. This small amount of downwind differencing can create noticeable oscillations near unresolved steep gradients in the flow. We used this example to argue that one should use the exact formulas for the Jacobian and associated transformation to characteristic variables, rather than simplified approximate expressions that often seem attractive in complex problems.

Donat and Marquina [1, 9] independently took this idea much further. They realized that near an unresolved steep gradient in the flow, in which the states vary by a large amount from one node to the next, there is no clear way to determine "the" value of the Jacobian midway between the nodes (where it is required for ENO and other methods). There may be unambiguous values of the Jacobian when extrapolating from nodal data from the left or from the right of the midpoint, but these left sided and right sided Jacobians can differ substantially. They propose to make use of these two Jacobians separately, in an upwind fashion, rather than attempt to define a single representative midpoint Jacobian. In doing so, they seem to have avoided the substantial uncertainty in the value of the Jacobian that results from trying to choose a single one from the large range of possible values "between" the left and right Jacobians. This uncertainty insures that any single choice of Jacobian will be a large perturbation from the true Jacobian of the exact solution of the underlying flow problem, and, as before, the result is an inaccurate transformation to characteristic fields. This allows for a mix of upwind and downwind differencing with the associated potential for oscillations.

Donat and Marquina make use of the left and right Jacobian in a consistent upwind way to compute the discretized convective flux terms. In the context of ENO methods, they propose to evaluate the left Jacobian with the left side biased interpolation of the conserved variables, and the right Jacobian with the right side biased interpolation of the conserved variables. Each of these interpolations is done in a high order accurate ENO fashion, e.g. the left interpolant is chosen as the smoothest possible polynomial interpolant of the desired degree that includes the left node in its stencil. Using each of these two Jacobian matrices separately, we are to compute convective flux derivatives in each characteristic field using the ENO method. Of these, only the right moving fluxes from the left and left moving fluxes from the right are actually used, the rest are discarded. The fluxes are then taken out of the characteristic fields, yielding two vector flux functions for the conserved variables. Adding these two vector fluxes together gives a consistent, high order accurate numerical flux function.

In [1, 9], the authors show many examples illustrating the advantages of using Marquina's Jacobian. In section 8 we present one more, since it is of special interest to us and also illustrates how their technique can be of value in more complicated applications.

We note a special case that could occur when using a second order accurate approximation to the conserved variables. If the smoothest possible approximation to the conserved variables from the left and from the right both happen to be the central linear average, then the resulting scheme is equivalent that using the standard ENO Jacobian evaluation.

It is tempting to think that some other way of constructing a Jacobian—or a $\vec{U}$ value—at the midpoint would yield a more appropriate value for the ENO scheme. Some form of interpolation must be used, since only nodal variable values are available; the linear average is merely a symmetrical choice. One might suppose that instead an upwind biased interpolation should be used to determine a midpoint value of the variables, since this would better reflect what information actually reaches the midpoint during the course of a time step. However, this idea is complicated by the fact that the upwind directions are only defined for the characteristic fields, while the conserved variables to be interpolated are mixtures of those fields. Thus, it is not possible to pick an upwind direction for a single conserved variable.

Still, this idea can be developed to determine a more physically reasonable midpoint Jacobian evaluation: based on the value of the Jacobian at the left and right adjacent nodes, one can transform to characteristic fields on the left side and the right side using these respective Jacobians. Then, one can interpolate all right moving characteristic fields from the left side to the midpoint, and all left moving characteristic fields from the right side to the midpoint, combine them into a single characteristic state vector, and transform this vector back to primitive variables to obtain a properly "upwind interpolated" midpoint state vector. The Jacobian evaluated at this state provides a single Jacobian for use with the ENO method. Our experiments show that this is indeed a superior choice over the standard linear average, in that it does greatly reduce spurious oscillations in those special cases when they occur. However, it does not perform as well as Marquina's procedure for making use of separate left and right Jacobians. It seems that no single midpoint Jacobian adequately represents the situation when the left and right nodal Jacobians differ greatly.

3 Model Equations and Discretizations

The numerical method we will present can be applied to a general system of convection-diffusion-reaction conservation equations in any number of spatial dimensions. For example, in two spatial dimensions (x, y) the vector form of the equations is

$$\vec{U}_t + [\vec{\mathcal{F}}(\vec{U})]_x + [\vec{\mathcal{G}}(\vec{U})]_y = [\vec{\mathcal{F}}_d(\vec{\nabla}\vec{U})]_x + [\vec{\mathcal{G}}_d(\vec{\nabla}\vec{U})]_y + \vec{\mathcal{S}}(\vec{U}) \tag{34}$$

where $\vec{U} = \vec{U}(x, y, t)$ is the vector of conserved variables, $\vec{\mathcal{F}}(\vec{U})$ and $\vec{\mathcal{G}}(\vec{U})$ are the vectors of convective fluxes, $\vec{\mathcal{F}}_d(\vec{\nabla}\vec{U})$ and $\vec{\mathcal{G}}_d(\vec{\nabla}\vec{U})$ are the vectors of diffusive fluxes, and $\vec{\mathcal{S}}(\vec{U})$ is the vector of reaction terms. Subscripts t, x, y denote the corresponding time and space partial derivatives. Note that our techniques can also be extended to apply to systems that have convective or diffusive terms that are not in conservation form. For example, non-conservative convective terms arise as the "thermal forces" in the Braginskii equations describing transport in a multi-species plasma.

4 Spatial Discretization

The diffusion terms can be evaluated with standard second or fourth order conservative central differencing. The reaction terms, which involve no derivatives, are simply evaluated at point values.

The finite difference ENO method is used to evaluate convection terms. It is applied independently to $[\vec{\mathcal{F}}(\vec{U})]_x$ and to $[\vec{\mathcal{G}}(\vec{U})]_y$—a "dimension by dimension" discretization. On a rectangular 2-D grid, we sweep through the grid from bottom to top performing ENO on 1-D horizontal rows of grid points to evaluate the $[\vec{\mathcal{F}}(\vec{U})]_x$ term. The $[\vec{\mathcal{G}}(\vec{U})]_y$ term is evaluated in a similar way by sweeping through the grid from left to right performing ENO on 1-D vertical rows of grid points. The ENO method will be fully described in subsequent sections.

The spatial discretization can be extended to cover equations that include first and second derivative terms that are not expressible in conservation form. Second derivative terms can still be treated with standard central differences, second or fourth order. For non-conservative convective terms, the ENO procedure must be based on a linearized convective matrix of the entire first order system, not just the part that is in conservation form. We will describe this extension in detail in a future report.

5 Time Discretization

Once we have a numerical approximation to each of the spatial terms in equation 34, we can write it abstractly as a system of Ordinary Differential Equations (ODEs)

$$\vec{U}_t = \vec{f}(\vec{U}) \tag{35}$$

This equation could be discretized in time by any ODE integrating method that has suitable accuracy and stability properties. If the spatial reaction or diffusion terms are particularly strong, to the point where their time step restrictions are much more limiting than that of the Courant-Friedrich-Lewy (CFL) restriction for the convective terms, it is better to handle them separately via a time splitting procedure and a stiff ODE integrator as described in [2].

For the general time integration of this ODE, the Total Variation Diminishing (TVD) Runge-Kutta methods of Shu and Osher [12] are particularly well suited. In addition to the simplicity of Runge-Kutta methods, they are specially designed for time integrating spatially discretized convection equations in a way that will not create spurious oscillations in the solution.

First order TVD Runge-Kutta is simply the forward Euler method,

$$\vec{U}^{n+1} = \vec{U}^n + \triangle t \vec{f}(\vec{U}^n) \tag{36}$$

Second order TVD Runge-Kutta is Heun's predictor-corrector method,

$$\vec{U}^\star = \vec{U}^n + \triangle t \vec{f}(\vec{U}^n) \tag{37}$$

$$\vec{U}^{n+1} = \vec{U}^n + \triangle t \left(\frac{1}{2}\vec{f}(\vec{U}^n) + \frac{1}{2}\vec{f}(\vec{U}^\star) \right) \tag{38}$$

A third order TVD Runge-Kutta method is given by,

$$\vec{U}^\star = \vec{U}^n + \triangle t \vec{f}(\vec{U}^n) \tag{39}$$

$$\vec{U}^{\star\star} = \vec{U}^n + \triangle t \left(\frac{1}{4}\vec{f}(\vec{U}^n) + \frac{1}{4}\vec{f}(\vec{U}^\star) \right) \tag{40}$$

$$\vec{U}^{n+1} = \vec{U}^n + \triangle t \left(\frac{1}{6}\vec{f}(\vec{U}^n) + \frac{1}{6}\vec{f}(\vec{U}^\star) + \frac{2}{3}\vec{f}(\vec{U}^{\star\star}) \right) \tag{41}$$

There are no convenient fourth order or higher TVD Runge-Kutta methods; they do exist, but they only maintain the TVD property when used with special, more complicated spatial discretizations. The standard fourth order accurate Runge-Kutta method can be used, but it is not TVD. This means it could cause spurious spatial oscillations, though in practice this has not been a problem.

The third order TVD method is generally recommended, since it has the greatest accuracy and largest time step stability region of the TVD schemes. Due to its large stability region (which includes a segment of purely imaginary linear growth rates), for a sufficiently small time step it is guaranteed to be linearly stable for the entire class of problems considered here. In contrast, the first and second order methods both require some spatial diffusion terms in order to be stable. Without that, no matter how small the time step is they may be mildly unstable (although it turns out the ENO spatial discretization can prevent this instability to some extent). For this reason, they should not be used unless there is substantial spatial diffusion in the problem.

6 The Finite Difference ENO Scheme

We are now ready to proceed with the precise presentation of the ENO finite difference discretization.

6.1 Reducing a System to Independent Scalar Equations

First, we show how the discretization for a system is reduced to that of independent scalar problems.

Consider the Jacobian matrix J of the $\vec{\mathcal{F}}(\vec{U})$ term in equation 18. We assume that this $N \times N$ Jacobian matrix has a complete eigensystem consisting of eigenvalues , $\lambda^p(\vec{U})$, left eigenvectors, $\vec{L}^p(\vec{U})$, and right eigenvectors, $\vec{R}^p(\vec{U})$, for $p = 1, \ldots, N$ that satisfy inversion and diagonalization relations 23 and 24.

At a specific point $x_{i_0+\frac{1}{2}}$ midway between two grid nodes, we wish to find the numerical flux function $\vec{F}_{i_0+\frac{1}{2}}$. We evaluate the eigensystem at $\vec{U}_{i_0+\frac{1}{2}}$. Our method for approximating the value of $\vec{U}_{i_0+\frac{1}{2}}$ for use in the left sided and right sided Jacobian evaluations is explained in section 7.1.

In the p-th characteristic field we have an eigenvalue $\lambda^p(\vec{U}_{i_0+\frac{1}{2}})$, left eigenvector $\vec{L}^p(\vec{U}_{i_0+\frac{1}{2}})$, and right eigenvector $\vec{R}^p(\vec{U}_{i_0+\frac{1}{2}})$. We put $\vec{U}$ values and $\vec{\mathcal{F}}(\vec{U})$ values into the p-th characteristic field by taking the dot product with the left eigenvector,

$$u = \vec{L}^p(\vec{U}_{i_0+\frac{1}{2}}) \cdot \vec{U} \tag{42}$$

$$f(u) = \vec{L}^p(\vec{U}_{i_0+\frac{1}{2}}) \cdot \vec{\mathcal{F}}(\vec{U}) \tag{43}$$

where u and $f(u)$ are scalars. Once in the characteristic field we perform a scalar version of ENO, obtaining a scalar numerical flux function $F_{i_0+\frac{1}{2}}$ in the scalar field. We take this flux out of the characteristic field by multiplying with the right eigenvector,

$$\vec{F}^p_{i_0+\frac{1}{2}} = F_{i_0+\frac{1}{2}} \vec{R}^p(\vec{U}_{i_0+\frac{1}{2}}) \tag{44}$$

where $\vec{F}^p_{i_0+\frac{1}{2}}$ is the portion of the numerical flux function $\vec{F}_{i_0+\frac{1}{2}}$ from the p-th field.

Once we have evaluated the contribution to the numerical flux function from each field, we get the total numerical flux from summing the contributions from each field,

$$\vec{F}_{i_0+\frac{1}{2}} = \sum_p \vec{F}^p_{i_0+\frac{1}{2}} \tag{45}$$

6.2 Finite Difference ENO for Scalar Equations

Once the system has been reduced to independent scalar conservation equations, we need only develop ENO in this simple setting.

ENO is performed in each scalar characteristic field, on the scalar equation

$$u_t + f(u)_x = 0 \tag{46}$$

where u and $f(u)$ come from equations 42 and 43 respectively.

We define the numerical flux function F through the relation

$$f(u_i)_x = \frac{F_{i+\frac{1}{2}} - F_{i-\frac{1}{2}}}{\triangle x} \tag{47}$$

where the $F_{i\pm\frac{1}{2}}$ are the values of the numerical flux function at the cell walls. To obtain a convenient algorithm for computing this numerical flux function, we proceed as follows: define $h(x)$ implicitly through the following equation,

$$f(u(x)) = \frac{1}{\triangle x} \int_{x-\frac{\triangle x}{2}}^{x+\frac{\triangle x}{2}} h(y)dy \tag{48}$$

taking a derivative on both sides of equation 48 yields,

$$f(u(x))_x = \frac{h(x+\frac{\triangle x}{2}) - h(x-\frac{\triangle x}{2})}{\triangle x} \tag{49}$$

which shows that h is identical to the numerical flux function F at the cell walls. That is $F_{i\pm\frac{1}{2}} = h(x_{i\pm\frac{1}{2}})$ for all i. We can in turn calculate h by finding its primitive

$$H(x) = \int_{x_{-\frac{1}{2}}}^{x} h(y)dy \tag{50}$$

and then taking a derivative. From relation 48, it turns out that $f(u(x_i))$ provides values for the first divided differences of H on the grid, which allows us to accurately and efficiently interpolate the derivative of H to any other necessary points.

We will calculate H at the cell walls with polynomial interpolation. Our goal is to calculate $h = H'$, so we do not need the zeroth order divided differences of H that vanish with the derivative. The zeroth order divided differences, $D^0_{i+\frac{1}{2}}$, and all higher order *even* divided differences of H exist at the cell walls and will have the subscript $i \pm \frac{1}{2}$. The first order divided differences D^1_i and all higher order *odd* divided differences of H exist at the grid points and will have the subscript i.

The first order divided differences of H are,

$$D^1_i H = \frac{H(x_{i+\frac{1}{2}}) - H(x_{i-\frac{1}{2}})}{\triangle x} = f(u(x_i)) \tag{51}$$

where the second equality sign comes from

$$H(x_{i+\frac{1}{2}}) = \int_{x_{-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} h(y)dy = \sum_{j=0}^{i} \left(\int_{x_{j-\frac{1}{2}}}^{x_{j+\frac{1}{2}}} h(y)dy \right) = \triangle x \sum_{j=0}^{i} f(u(x_j)) \tag{52}$$

which follows from equations 50 and 48. The higher divided differences are,

$$D^2_{i+\frac{1}{2}} H = \frac{f(u(x_{i+1})) - f(u(x_i))}{2\triangle x} = \frac{1}{2} D^1_{i+\frac{1}{2}} f \tag{53}$$

$$D^3_i H = \frac{1}{3} D^2_i f \tag{54}$$

and they continue in that manner.

According to the rules of polynomial interpolation, we can take any path along the divided difference table to construct H, although they do not all give good results. ENO reconstruction consists of two important features:

1. Choose $D^1_i H$ in the upwind direction.
2. Choose higher order divided differences by taking the smaller in absolute value of the two possible choices.

Once we construct $H(x)$, we evaluate $H'(x_{i+\frac{1}{2}})$ to get the numerical flux $F_{i+\frac{1}{2}}$.

It is important to note that there are other ways to choose the higher order divided differences. For example, in step 2 one can bias the decision towards the more central divided difference, which lowers the truncation error of the scheme in smooth regions.

6.3 The ENO-Roe Discretization (Third Order Accurate)

For a specific cell wall, located at $x_{i_0+\frac{1}{2}}$, we find the associated numerical flux function $F_{i_0+\frac{1}{2}}$ as follows:

If $\lambda^p(\vec{U}_{i_0+\frac{1}{2}}) > 0$, then $k = i_0$. Otherwise, set $k = i_0 + 1$. Define

$$Q_1(x) = (D_k^1 H)(x - x_{i_0+\frac{1}{2}}) \tag{55}$$

If $|D^2_{k-\frac{1}{2}}H| \le |D^2_{k+\frac{1}{2}}H|$, then $c = D^2_{k-\frac{1}{2}}H$ and $k^\star = k - 1$. Otherwise, $c = D^2_{k+\frac{1}{2}}H$ and $k^\star = k$. Define

$$Q_2(x) = c(x - x_{k-\frac{1}{2}})(x - x_{k+\frac{1}{2}}) \tag{56}$$

If $|D^3_{k^\star}H| \le |D^3_{k^\star+1}H|$, then $c^\star = D^3_{k^\star}H$. Otherwise, $c^\star = D^3_{k^\star+1}H$. Define

$$Q_3(x) = c^\star(x - x_{k^\star-\frac{1}{2}})(x - x_{k^\star+\frac{1}{2}})(x - x_{k^\star+\frac{3}{2}}) \tag{57}$$

Then,

$$F_{i_0+\frac{1}{2}} = H'(x_{i_0+\frac{1}{2}}) = Q_1'(x_{i_0+\frac{1}{2}}) + Q_2'(x_{i_0+\frac{1}{2}}) + Q_3'(x_{i_0+\frac{1}{2}}) \tag{58}$$

which simplifies to

$$F_{i_0+\frac{1}{2}} = D_k^1 H + c\,(2(i_0 - k) + 1)\,\triangle x + c^\star\left(3(i_0 - k^\star)^2 - 1\right)(\triangle x)^2 \tag{59}$$

by using equations 55, 56, and 57.

6.4 The Entropy Fix

The ENO-Roe discretization can admit entropy violating expansion shocks near sonic points. That is, at a place where a characteristic velocity changes sign—a "sonic point"—it is possible to have a stationary "expansion shock" solution with a discontinuous jump in value. If this jump were smoothed out even slightly, it would break up into an expansion "fan" (i.e. rarefaction) and dissipate, which is the desired physical solution. For a specific cell wall, $x_{i_0+\frac{1}{2}}$, if there are no "nearby" sonic points, then we use ENO-Roe. Otherwise, we add *high order* dissipation to our calculation of $F_{i_0+\frac{1}{2}}$ which is extremely small if the solution is locally smooth, but is large enough to break up an expansion shock. We explain when a sonic point is considered "nearby" in the next section. This approach retains a uniformly high order accurate scheme in smooth regions, and eliminates any "entropy violating" expansion shocks.

Consider two primitive functions H^+ and H^-. We compute a divided difference table for each of them. Their first divided differences are,

$$D_i^1 H^\pm = \frac{1}{2} f(u_i) \pm \frac{1}{2}\alpha_{i_0+\frac{1}{2}} u_i \tag{60}$$

where $\alpha_{i_0+\frac{1}{2}}$ is defined in section 7.4. We define the second divided differences $D^2_{i+\frac{1}{2}}H^{\pm}$ and the third divided differences $D^3_i H^{\pm}$ in the standard way, like those of H.

For H^+, set $k = i_0$. Then, replacing H with H^+ everywhere, define $Q_1(x)$, $Q_2(x)$, $Q_3(x)$, and finally $F^+_{i_0+\frac{1}{2}}$ by using the algorithm above. For H^-, set $k = i_0 + 1$. Then, replacing H with H^- everywhere, define $Q_1(x)$, $Q_2(x)$, $Q_3(x)$, and finally $F^-_{i_0+\frac{1}{2}}$ by using the algorithm above. Then,

$$F_{i_0+\frac{1}{2}} = F^+_{i_0+\frac{1}{2}} + F^-_{i_0+\frac{1}{2}} \tag{61}$$

is the new numerical flux function with added high order dissipation.

7 Marquina's Flux Splitting

7.1 Finding $\vec{U}_{i_0+\frac{1}{2}}$ (Third Order Accurate Algorithm)

We will need two approximations for the value of $\vec{U}$ at the cell wall $x_{i_0+\frac{1}{2}}$. The value from the left, $\vec{U}^L_{i_0+\frac{1}{2}}$, is interpolated from the x_{i_0} side. The value from the right, $\vec{U}^R_{i_0+\frac{1}{2}}$, is interpolated from the x_{i_0+1} side.

We need the divided differences of $\vec{U}$ for polynomial interpolation. This takes no extra work, since we have already computed the divided difference of $\vec{U}$ for use in the entropy fix portion of the ENO algorithm. The interpolation is done for each conserved variable, v, and we phrase the algorithm in terms of v. The divided differences of v are: $D^0_i v = v_i$, $D^1_{i+\frac{1}{2}}v$, and $D^2_i v$. For each conserved variable, v, we choose the zeroth order divided difference in the left or right direction based on whether we are looking for v^L or v^R. Then, we choose the higher order divided differences by taking the smaller in absolute value of the two possible choices.

For a specific cell wall, located at $x_{i_0+\frac{1}{2}}$, we find the approximations to $v^L_{i_0+\frac{1}{2}}$ and $v^R_{i_0+\frac{1}{2}}$ as follows:

If we are looking for $v^L_{i_0+\frac{1}{2}}$, then $k = i_0$. If we are looking for $v^R_{i_0+\frac{1}{2}}$, then $k = i_0 + 1$. Define

$$Q_0(x) = D^0_k v = v_k \tag{62}$$

If $|D^1_{k-\frac{1}{2}}v| \le |D^1_{k+\frac{1}{2}}v|$, then $c = D^1_{k-\frac{1}{2}}v$ and $k^\star = k-1$. Otherwise, $c = D^1_{k+\frac{1}{2}}v$ and $k^\star = k$. Define

$$Q_1(x) = c(x - x_k) \tag{63}$$

If $|D^2_{k^\star}v| \le |D^2_{k^\star+1}v|$, then $c^\star = D^2_{k^\star}v$. Otherwise, $c^\star = D^2_{k^\star+1}v$. Define

$$Q_2(x) = c^\star(x - x_{k^\star})(x - x_{k^\star+1}) \tag{64}$$

Then,

$$v_{i_0+\frac{1}{2}} = Q_0(x_{i_0+\frac{1}{2}}) + Q_1(x_{i_0+\frac{1}{2}}) + Q_2(x_{i_0+\frac{1}{2}}) \tag{65}$$

which simplifies to one of the following

$$v^L_{i_0+\frac{1}{2}} = v_{i_0} + \frac{c\triangle x}{2} + c^\star \left((i_0 - k^\star)^2 - \frac{1}{4} \right) (\triangle x)^2 \tag{66}$$

$$v^R_{i_0+\frac{1}{2}} = v_{i_0+1} - \frac{c\triangle x}{2} + c^\star \left((i_0 - k^\star)^2 - \frac{1}{4} \right) (\triangle x)^2 \tag{67}$$

depending on which one was being computed.

7.2 Constructing Marquina's Left and Right Jacobians

Consider a cell wall, $x_{i_0+\frac{1}{2}}$, where we wish to calculate the numerical flux function $\vec{F}_{i_0+\frac{1}{2}}$. We have two estimates for $\vec{U}_{i_0+\frac{1}{2}}$: one estimate from the left, $\vec{U}^L_{i_0+\frac{1}{2}}$, and one estimate from the right, $\vec{U}^R_{i_0+\frac{1}{2}}$. We use these estimates to compute two Jacobian matrices,

$$J^L = J(\vec{U}^L_{i_0+\frac{1}{2}}), \qquad J^R = J(\vec{U}^R_{i_0+\frac{1}{2}}) \tag{68}$$

and their associated eigensystems.

In the p-th characteristic field, we have an eigenvalue, left eigenvector, and right eigenvector for each of the two Jacobians:

$$(\lambda^p)^L = \lambda^p(\vec{U}^L_{i_0+\frac{1}{2}}), \qquad (\lambda^p)^R = \lambda^p(\vec{U}^R_{i_0+\frac{1}{2}}) \tag{69}$$

$$(\vec{L}^p)^L = \vec{L}^p(\vec{U}^L_{i_0+\frac{1}{2}}), \qquad (\vec{L}^p)^R = \vec{L}^p(\vec{U}^R_{i_0+\frac{1}{2}}) \tag{70}$$

$$(\vec{R}^p)^L = \vec{R}^p(\vec{U}^L_{i_0+\frac{1}{2}}), \qquad (\vec{R}^p)^R = \vec{R}^p(\vec{U}^R_{i_0+\frac{1}{2}}) \tag{71}$$

7.3 Constructing Left and Right Numerical Flux Functions

Consider a cell wall, $x_{i_0+\frac{1}{2}}$, where we wish to find the numerical flux in the p-th characteristic field: $\vec{F}^p_{i_0+\frac{1}{2}}$. For each of the two Jacobians, we find a numerical flux in the p-th characteristic field: $(\vec{F}^p_{i_0+\frac{1}{2}})^L$ and $(\vec{F}^p_{i_0+\frac{1}{2}})^R$. Then we sum them,

$$\vec{F}^p_{i_0+\frac{1}{2}} = (\vec{F}^p_{i_0+\frac{1}{2}})^L + (\vec{F}^p_{i_0+\frac{1}{2}})^R \tag{72}$$

to get the total numerical flux in the p-th field. This is done for each field, and then the total numerical flux is defined by equation 45.

7.4 Constructing the ENO Numerical Scheme

Consider a cell wall, $x_{i_0+\frac{1}{2}}$. If the left and right eigenvalues evaluated at this cell wall agree on the upwind direction then there is no sonic point "nearby", and we use the ENO-Roe discretization. If the eigenvalues disagree at the cell wall, then there is a sonic point "nearby", and we use the version of ENO with the entropy fix. There are 3 cases:

1. If $(\lambda^p)^L > 0$ and $(\lambda^p)^R > 0$, then upwind is from the left. We calculate $(\vec{F}^p_{i_0+\frac{1}{2}})^L$ using ENO-Roe. We set $(\vec{F}^p_{i_0+\frac{1}{2}})^R = 0$.

2. If $(\lambda^p)^L < 0$ and $(\lambda^p)^R < 0$, then upwind is from the right. We calculate $(\vec{F}^p_{i_0+\frac{1}{2}})^R$ using ENO-Roe. We set $(\vec{F}^p_{i_0+\frac{1}{2}})^L = 0$.

3. If $(\lambda^p)^L(\lambda^p)^R \leq 0$, then the eigenvalues disagree. We use the entropy fix version of ENO. For this, we define

$$\alpha_{i_0+\frac{1}{2}} = \max\left(|(\lambda^p)^L|, |(\lambda^p)^R|\right) \tag{73}$$

as our dissipation coefficient. In the evaluation of $(\vec{F}^p_{i_0+\frac{1}{2}})^L$, we evaluate $F^+_{i_0+\frac{1}{2}}$ normally, but set $F^-_{i_0+\frac{1}{2}} = 0$. Thus, equation 61 becomes $F_{i_0+\frac{1}{2}} = F^+_{i_0+\frac{1}{2}}$. In the evaluation of $(\vec{F}^p_{i_0+\frac{1}{2}})^R$, we evaluate $F^-_{i_0+\frac{1}{2}}$ normally, but set $F^+_{i_0+\frac{1}{2}} = 0$. Thus, equation 61 becomes $F_{i_0+\frac{1}{2}} = F^-_{i_0+\frac{1}{2}}$.

This completes the description of the finite difference ENO discretization using Marquina's Jacobians.

8 Examples

8.1 Example 1: Reflecting Shock in a Thermally Perfect Gas

We are currently developing numerical methods for treating an interface separating a liquid drop and a high speed gas flow. The droplet is an incompressible Navier-Stokes fluid. The gas is a compressible, multi-species, chemically reactive Navier-Stokes fluid. A level set is used for domain decomposition. (This research will be described in detail in a future UCLA CAM report.)

In this example, a 1D "Sod" shock tube was set up in the middle of the domain, with the generated shock moving from left to right. The water droplet is off to the right hand side of the domain. The shock hits the water droplet, reflects off in the opposite direction, and proceeds toward the contact discontinuity. We implement standard 3rd order ENO with the Jacobian matrix evaluated at the linear average of the points adjacent to the flux. This is a second order accurate, central approximation to the Jacobian. Using standard finite difference ENO, there is a great deal of "noise" generated when the shock approaches the contact discontinuity, after reflection off the water droplet. See Figure 1. Note, however, that standard 2nd order ENO (which is a TVD scheme) with the Jacobian matrix evaluated at the linear average does not generate much noise at all.

We run the same problem with 3rd order ENO, but this time we used Marquina's Jacobian evaluated with 3rd order accurate left side and right side biased approximations to the conserved variables. There is no significant noise. See Figure 2.

(Note that the actual values for the density and the pressure of the water droplet are not shown. We use "place holder" values in the figures. However, the values for the velocity and the temperature are unaltered.)

8.2 Example 2: Importance of High Order Accurate Jacobians

We emphasize that it is important to use Marquina's Jacobian with a high order accurate approximation to the conserved variables at the cell walls. To illustrate this, consider the previous problem with 3rd order ENO. The Jacobian is evaluated with 1st, 2nd, and 3rd order accurate approximations to the conserved variables. The results are shown in Figures 3, 4, and 5 respectively. Note that all the ENO algorithms are 3rd order, only the approximations to the conserved variables for the left and right Jacobian evaluations vary in order. Based on these results, we recommend using 3rd order ENO with Marquina's Jacobian also evaluated to 3rd order.

8.3 Further Examples

See [1, 9] for more numerical examples using Marquina's Jacobian to fix a variety of spurious oscillatory effects.

9 Conclusions

ENO methods are a class of high accuracy, shock capturing numerical methods for general hyperbolic systems of conservation laws. They are based on using upwind biased interpolations in the characteristic fields without interpolating across steep gradients in the flow.

The finite difference formulation of the ENO method allows an efficient and convenient implementation that readily applies to any number of spatial dimensions.

This method works well on a great variety of gas dynamics problems, as well as other convective problems, but there are still special circumstances in which spurious oscillations develop.

Based on recent work, we have identified the source of these oscillations as the centered linear average interpolation used to evaluate the Jacobian and eigensystem of the convective flux at the midpoints between nodes, prior to transforming to characteristic fields. This effect can be understood intuitively as well, in terms of unintentionally performing downwind differencing of the true characteristic fields near steep gradients.

Marquina recently devised a way to make use of left side and right side Jacobians at the midpoint, without the need to construct a single Jacobian. The general technique seems to fix all known cases in which serious spurious oscillations have occurred.

We presented a detailed description of the preferred (third order accurate in space and time) finite difference ENO scheme using Marquina's Jacobian evaluation procedure, so that others can readily make use of this (pen)ultimate scheme.

We presented examples demonstrating that this approach fixes a large, non-physical oscillation in a complicated gas dynamics problem. We also showed that it is important to evaluate the Jacobian and eigensystem to high order accuracy from the left and from the right at the midpoint, as this has a large impact on the practical resolution of the scheme. This is contrary to what one would naively expect, since the formal order of accuracy of the scheme is unchanged by the Jacobian evaluation strategy.

More analysis is required to understand why this two sided approach works so well, and why it has such a large effect on resolution without altering the formal order of accuracy. For now, however, it does seem to allow a robust, general, accurate, parameter-free ENO scheme which we expect will have wide application for problems which include a hyperbolic system.

10 Acknowledgements

We dedicate this paper to the memory of Ami Harten whose creativity and personality inspired everyone in the field.

Research of the first, second and fourth authors supported in part by ARPA URI-ONR-N00014-92-J-1890, NSF #DMS 94-04942, and ARO DAAH04-95-1-0155. Research of the third author supported in part by a University of Valencia grant and DGYCIT PB94-0987.

References

[1] Donat, R., and Marquina, A., *Capturing Shock Reflections: An Improved Flux Formula*, J. Comput. Phys. 25, 42-58 (1996).

[2] Fedkiw, R., Merriman, B., and Osher, S., *Numerical Methods for a Mixture of Thermally Perfect and/or Calorically Perfect Gaseous Species with Chemical Reactions*, J. Comput. Phys. 132, 175-190 (1997).

[3] B. van Leer, *Towards the Ultimate Difference Scheme I. The Quest for Monotonicity*, Springer Lecture Notes in Physics 18, 163-168 (1973).

[4] B. van Leer, *Towards the Ultimate Difference Scheme II. Monotonicity and Conservation Combined in a Second Order Scheme*, J. Comput. Phys. 14, 361-370 (1974).

[5] B. van Leer, *Towards the Ultimate Difference Scheme III. Upstream-Centered Finite Difference Schemes for Ideal Compressible Flow*, J. Comput. Phys. 23, 263-275 (1977).

[6] B. van Leer, *Towards the Ultimate Difference Scheme IV. A New Approach to Numerical Convection*, J. Comput. Phys. 23, 276-299 (1977).

[7] B. van Leer, *Towards the Ultimate Difference Scheme V. A Second Order Sequel to Gudonov's Method*, J. Comput. Phys. 32, 101-136 (1979).

[8] Randall J. LeVeque, *Numerical Methods for Conservation Laws*, Birhauser Verlag, Boston, USA. 1992. ISBN 3-8176-2723-5.

[9] Marquina, A., and Donat, R., *Capturing Shock Reflections: A Nonlinear Local Characteristic Approach*, UCLA CAM Report 93-31, April 1993.

[10] Shu, C.W., *Numerical experiments on the accuracy of ENO and modified ENO schemes*, J. Sci. Comput. 5, 127-149 (1990).

[11] Shu, C.W. and Osher, S., *Efficient Implementation of Essentially Non-Oscillatory Shock Capturing Schemes*, J. Comput. Phys. 77, 429-471 (1988).

[12] Shu, C.W. and Osher, S., *Efficient Implementation of Essentially Non-Oscillatory Shock Capturing Schemes II (two)*, J. Comput. Phys. 83, 32-78 (1989).

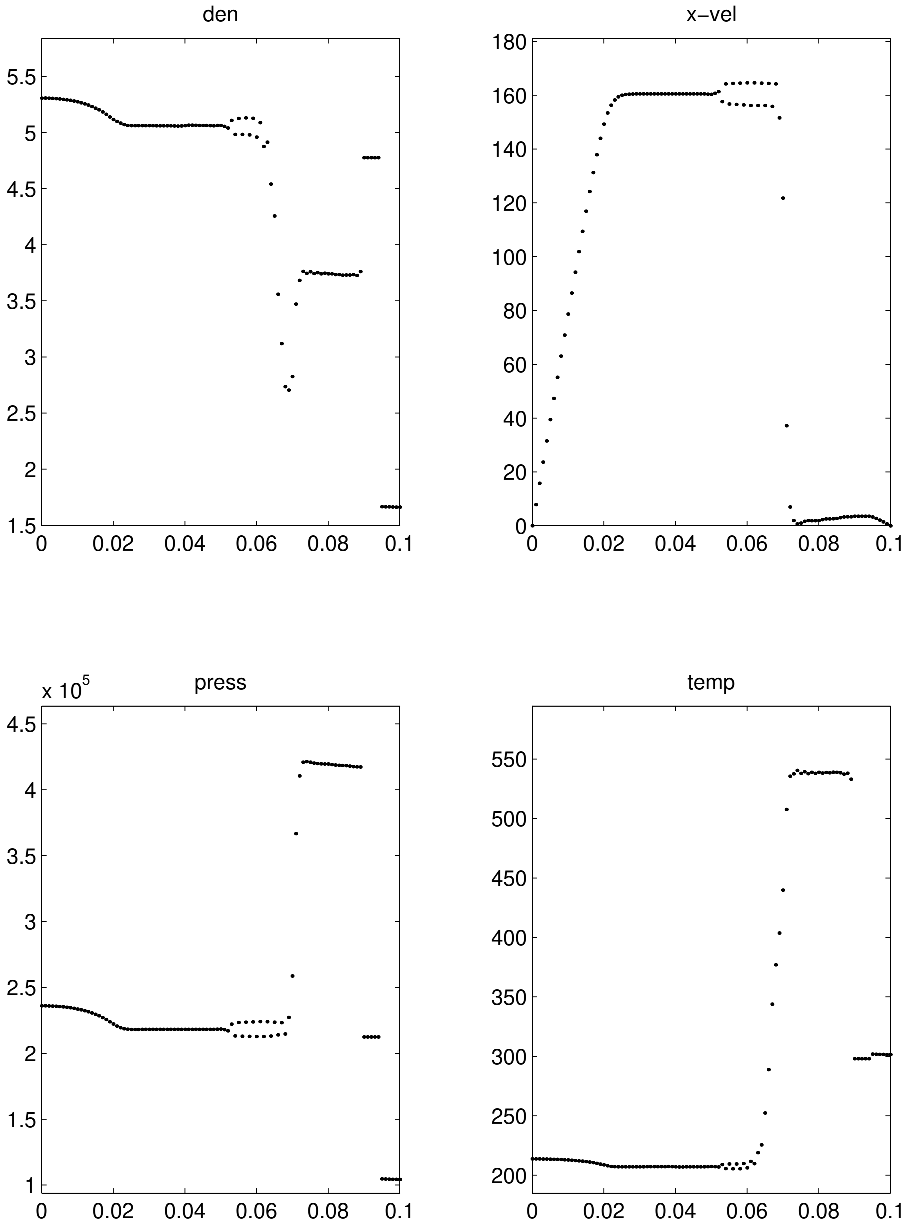

Figure 1: 3rd order ENO, Jacobian matrix evaluated at the linear average. Note the large spurious oscillations near $x = 0.06$.

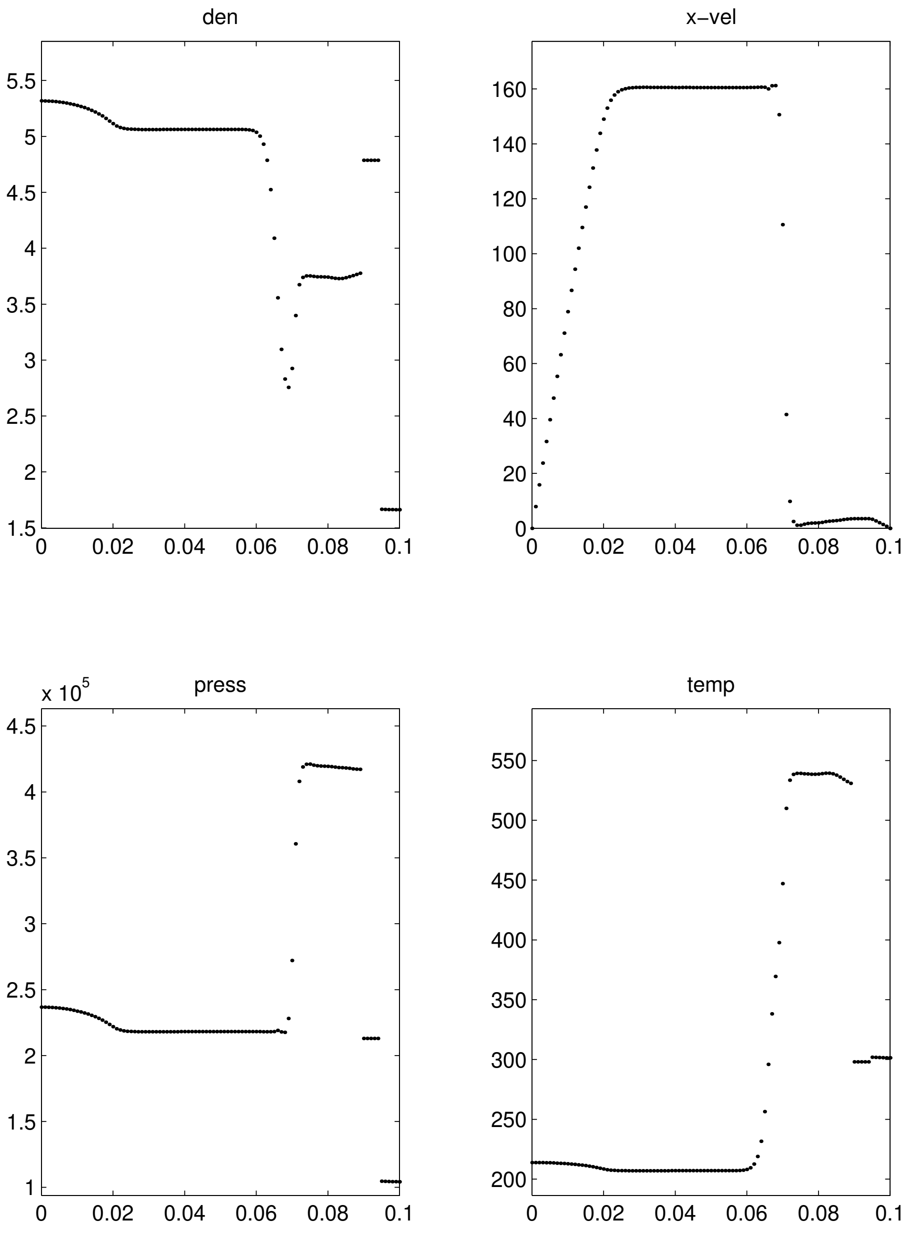

Figure 2: 3rd order ENO, 3rd order Marquina's Jacobian. The spurious oscillations of ENO are eliminated.

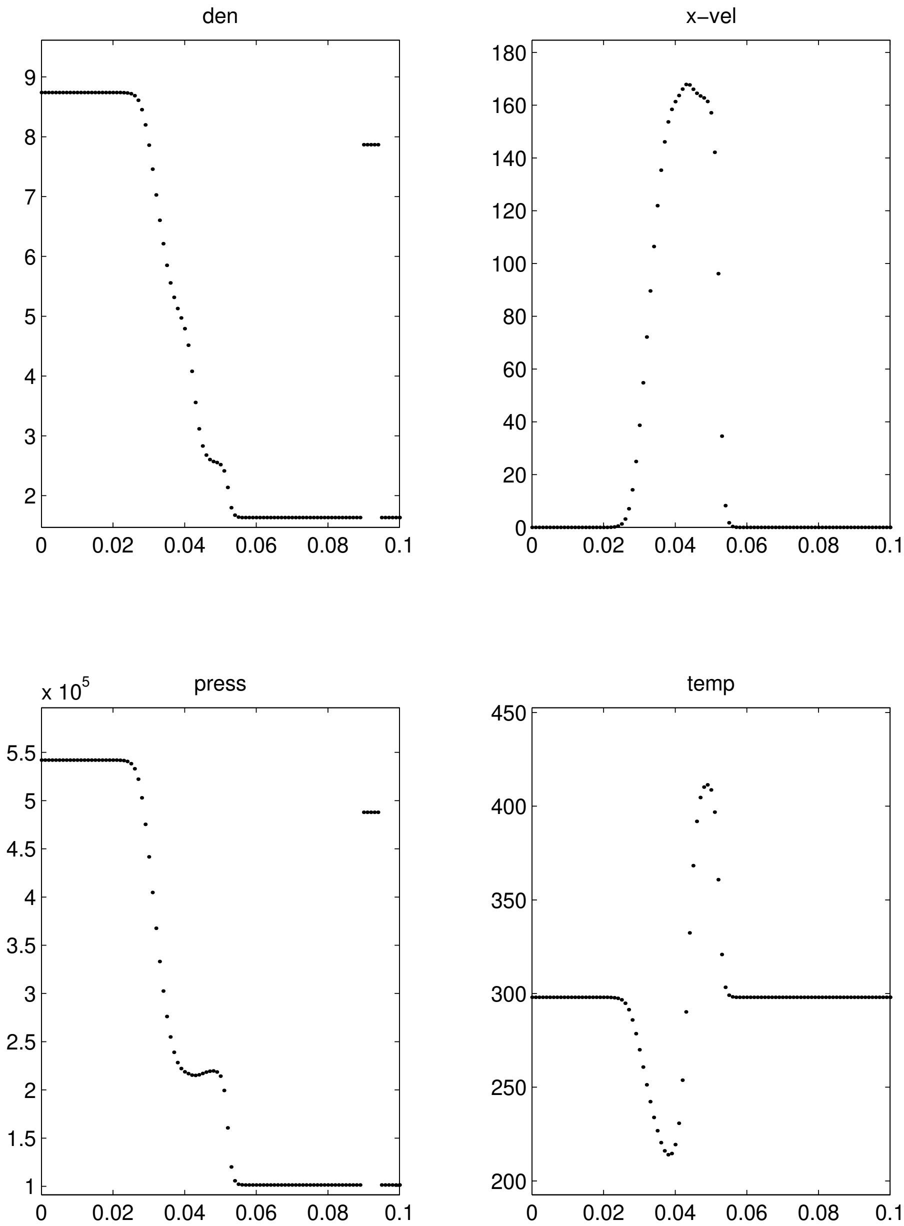

Figure 3: 3rd order ENO, 1st order Marquina's Jacobian. Note the smoothed out features, particularly near $x = 0.04$, due only to the low accuracy of the Jacobian evaluation.

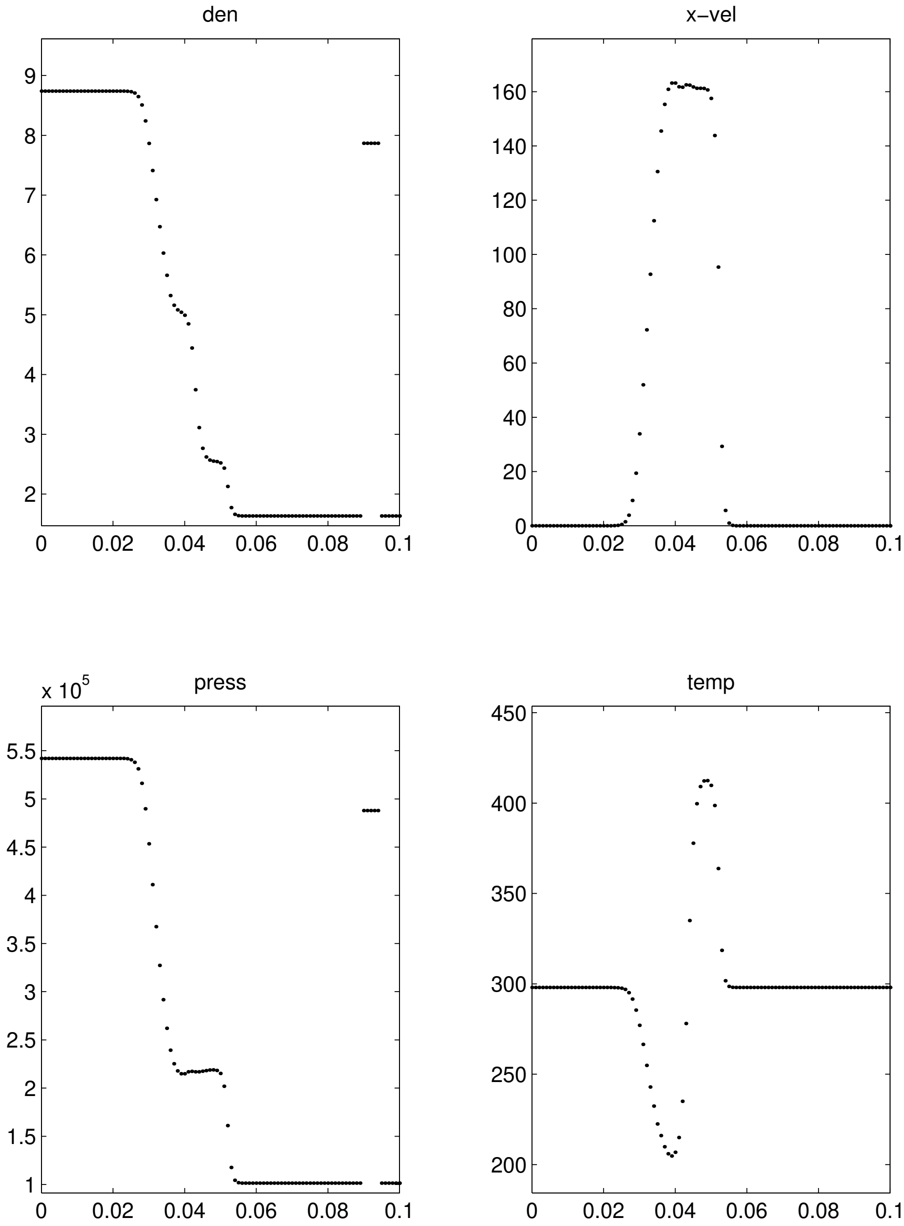

Figure 4: 3rd order ENO, 2nd order Marquina's Jacobian. The features at $x = 0.04$ are sharpened as the Jacobian accuracy is increased.

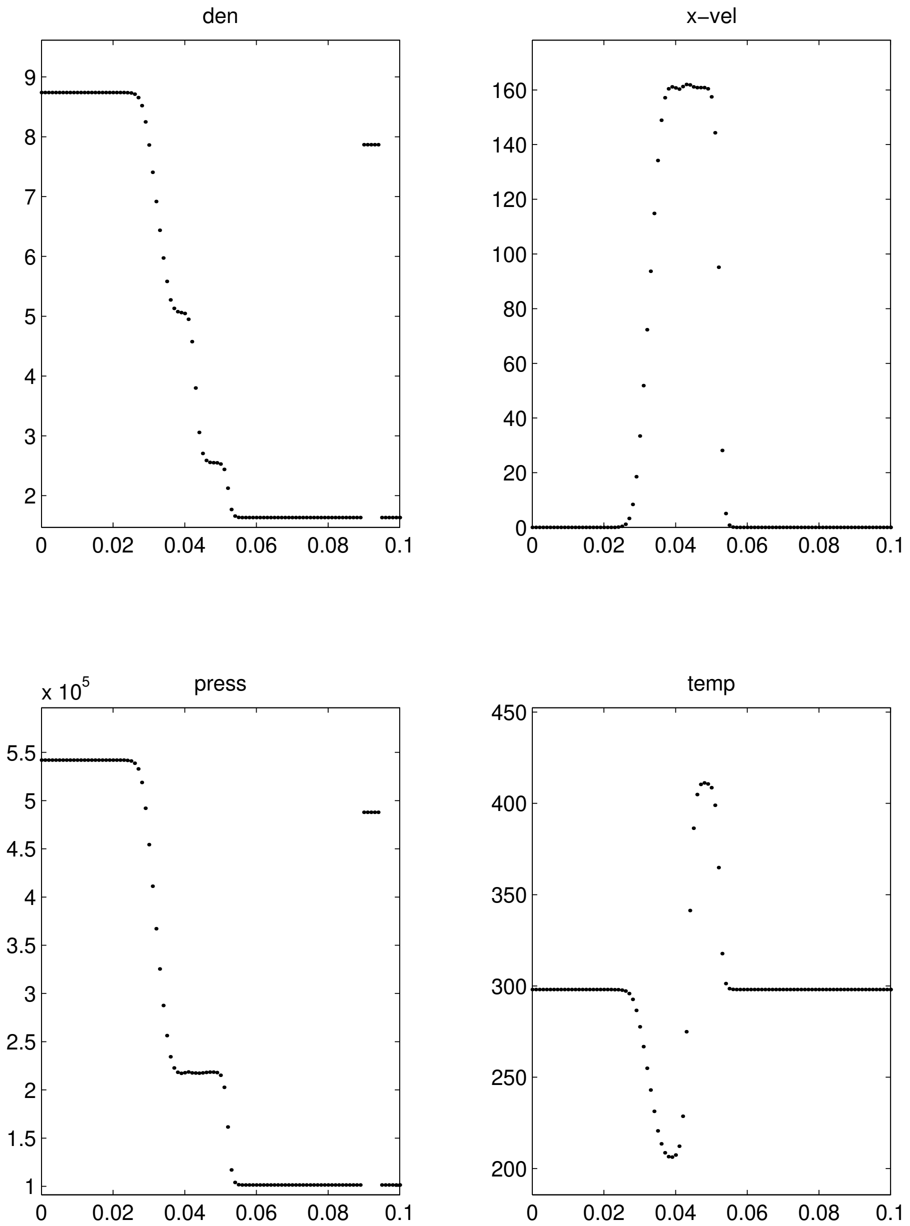

Figure 5: 3rd order ENO, 3rd order Marquina's Jacobian. The features at $x = 0.04$ are now well resolved, due only to the high accuracy of the Jacobian evaluation.