/N-34
013265

NASA Technical Memorandum 107367
AIAA-97-2037

Accurate Monotonicity-Preserving Schemes With Runge-Kutta Time Stepping

A. Suresh
NYMA, Inc.
Brook Park, Ohio

and

H.T. Huynh
Lewis Research Center
Cleveland, Ohio

Prepared for the
13th CFD Conference
sponsored by the American Institute of Aeronautics and Astronautics
Snowmass, Colorado, June 29—July 2, 1997

National Aeronautics and
Space Administration

ACCURATE MONOTONICITY-PRESERVING SCHEMES WITH RUNGE-KUTTA TIME STEPPING

A. Suresh
NYMA, Inc.,
Brookpark, Ohio 44142

and

H. T. Huynh
NASA Lewis Research Center
Cleveland, Ohio 44135

Abstract

A new class of high-order monotonicity-preserving schemes for the numerical solution of conservation laws is presented. The interface value in these schemes is obtained by limiting a higher-order polynomial reconstruction. The limiting is designed to preserve accuracy near extrema and to work well with Runge-Kutta time stepping. Computational efficiency is enhanced by a simple test that determines whether the limiting procedure is needed. For linear advection in one dimension, these schemes are shown to be monotonicity-preserving and uniformly high-order accurate. Numerical experiments for advection as well as the Euler equations also confirm their high accuracy, good shock resolution, and computational efficiency.

Introduction

We consider higher-order schemes (at least third-order) for the numerical solution of the Euler equations. Typical solutions to these equations have smooth structures interspersed with discontinuities. The challenge is to develop schemes that are highly accurate in smooth regions and have sharp nonoscillatory transitions at discontinuities.

Achieving this dual objective remains a daunting task. Among the first attempts, Colella and Woodward [2] introduced a piecewise parabolic method (PPM), which employs a four-point centered stencil to define the interface value; this value is then limited to control oscillations. The centered stencil, however, results in a scheme with a large dispersion error, and the limiting procedure causes accuracy to degenerate to first-order near extrema.

The essentially nonoscillatory (ENO) schemes of Harten et al. [3] were developed via a different line of thought. In these schemes, an adaptive stencil is used to select the "smoothest" data, thereby avoiding interpolations across discontinuities. While an adaptive stencil does avoid spurious oscillations near discontinuities, it does not make use of all the available data. The weighted-ENO (WENO) schemes by Liu et al. [11] and Jiang and Shu [9] make better use of the available data by defining the interface value as a weighted average of the interface values from all stencils. The weights are designed so that in smooth regions the scheme nearly recovers a very accurate interface value using all stencils but, near discontinuities, it recovers the value from the smoothest stencil. The WENO schemes, however, are still diffusive: they smear discontinuities nearly as much as the ENO schemes.

In this paper, we follow the limiting approach. The interface value is defined by a five-point stencil. As a result, the leading error is dissipative, and the dispersive error is considerably smaller than that of the four-point stencil. This interface value also combines well with the three-stage Runge-Kutta time stepping. Similar to PPM, oscillations are controlled by a limiting procedure. The key differences, however, are that our limiting is designed to preserve accuracy near extrema and to work well with Runge-Kutta time stepping. The resulting scheme is accurate in smooth regions, resolves discontinuities with high resolution, and is also efficient.

Note that a piecewise linear scheme of this type was presented by Huynh [6]. Extensions to piecewise parabolic schemes were presented by Suresh [18] and Huynh [7]. The present scheme incorporates these ideas within a Runge-Kutta time integration framework.

In §1, the spatial discretization and the Runge-Kutta time integration are reviewed. Section 2 describes the reconstruction procedure, which is the key feature of our scheme. Extensions of this scheme to systems of equations and multi-dimensions are dealt with in §3. Numerical experiments appear in §4. Finally, conclusions are presented in §5.

1. Discretization

For simplicity, we describe the methods for the advection equation with constant speed a,

$$u_t + au_x = 0, \tag{1.1a}$$

$$u(x,0) = u_0(x) \tag{1.1b}$$

where t is time, x is distance, and $u_0(x)$ is the initial condition. For the moment, the solution is assumed to be periodic in x so that no boundary conditions are needed.

Let x_j be the cell center of a uniform mesh, $x_{j+1/2}$ the interface between the j-th and $j+1$-th cells, and h the cell width. Denote by $\bar{u}_j(t)$ the cell average of u at time t,

$$\bar{u}_j(t) = \frac{1}{h}\int_{x_{j-1/2}}^{x_{j+1/2}} u(x,t)\,dx. \tag{1.2}$$

Integrating (1.1a) over the cell $[x_{j-1/2}, x_{j+1/2}]$ yields

$$\frac{d\bar{u}_j}{dt} + \frac{a}{h}\left[u(x_{j+1/2},t) - u(x_{j-1/2},t)\right] = 0. \tag{1.3}$$

At time $t^n = n\tau$ where τ is the time step, assume that we know v_j^n which approximates $\bar{u}_j(t^n)$. We wish to calculate v_j^{n+1}. For simplicity of notation, we omit the superscript n when there is no confusion, e.g., v_j denotes v_j^n.

An approximation to $u(x_{j+1/2},t^n)$ is called the interface value and is denoted by $v_{j+1/2}$. The calculation of the interface value from the known cell averages is accomplished in two steps. In the first or *reconstruction* step, nonoscillatory approximations of $u(x_{j+1/2},t^n)$ to the left and right sides of the interface $x_{j+1/2}$ denoted by $v_{j+1/2}^L$ and $v_{j+1/2}^R$ are constructed. This step determines the scheme's order of accuracy and is the main concern of this paper. In the next or *upwind* step, the interface value is determined by the wind direction: If $a > 0$, $v_{j+1/2} = v_{j+1/2}^L$; otherwise, $v_{j+1/2} = v_{j+1/2}^R$. Thus for advection, we need only one of the two values $v_{j+1/2}^L$ and $v_{j+1/2}^R$. For the

Euler equations, however, we will need both, and we employ well-known methods for the upwind step.

Equation (1.3) can be integrated by a standard Runge-Kutta method. Here we use the three-stage scheme of Shu and Osher [15]. With v representing $\{v_j\}$, denote by $L(v)$ the spatial operator

$$L(v)_j = -(v_{j+1/2} - v_{j-1/2}). \tag{1.4}$$

Then this scheme is given by

$$\begin{aligned} w^{(0)} &= v^n \\ w^{(1)} &= w^{(0)} + \sigma L(w^{(0)}) \\ w^{(2)} &= \tfrac{3}{4}w^{(0)} + \tfrac{1}{4}(w^{(1)} + \sigma L(w^{(1)})) \\ w^{(3)} &= \tfrac{1}{3}w^{(0)} + \tfrac{2}{3}(w^{(2)} + \sigma L(w^{(2)})) \\ v^{n+1} &= w^{(3)} \end{aligned} \tag{1.5}$$

where $\sigma = a\tau/h$ is the CFL number.

Observe that Runge-Kutta schemes like (1.5) are made up of repeated applications of a single stage scheme given by $w^{(k)} + \sigma L(w^{(k)})$, $k = 0, 1$, and 2. Moreover, each stage is an explicit Euler scheme, e.g.,

$$w_j^{(1)} = v_j - \sigma(v_{j+1/2} - v_{j-1/2}). \tag{1.6}$$

Therefore, we first design a monotonicity-preserving scheme for (1.6) and then extend it to the full scheme (1.5).

2. Reconstruction

Without loss of generality, we discuss the reconstruction only for $v^L_{j+1/2}$, i.e., we assume $a > 0$. The reconstruction is carried out in two steps. In the first step an accurate and stable formula is used to compute the interface value which is called the *original* value. In the second step, this value is then modified or *limited* appropriately to achieve a monotonicity-preserving scheme.

A straightforward choice for $v^L_{j+1/2}$ using the five cell averages $v_{j-2}, \ldots, v_{j+2}$ (the same stencil as the third-order ENO scheme) is

$$v^L_{j+1/2} = (2v_{j-2} - 13v_{j-1} + 47v_j + 27v_{j+1} - 3v_{j+2})/60. \tag{2.1}$$

Other choices include a low phase error fourth-order formula [8]

$$v^L_{j+1/2} = (9v_{j-2} - 56v_{j-1} + 194v_j + 104v_{j+1} - 11v_{j+2})/240, \tag{2.2}$$

or a fifth-order accurate implicit formula given by

$$(3v^L_{j-1/2} + 6v^L_{j+1/2} + v^L_{j+3/2})/10 = (v_{j-1} + 19v_j + 10v_{j+1})/30. \tag{2.3}$$

The implicit formula has the advantage of low dispersive and dissipative errors; its disadvantage is that the tridiagonal matrix inversion costs more.

2.1 Monotonicity constraint

We derive constraints for the interface value so that monotonicity is preserved by (1.6). First, we need a few definitions. Let the median of three numbers be the number that lies between the other two. Let minmod (x, y) be the median of x, y, and 0. Equivalently,

$$\text{minmod}\,(x,y) = \tfrac{1}{2}\,[\text{sgn}(x) + \text{sgn}(y)]\,\min\,(|x|,|y|). \tag{2.4}$$

Conversely, the median function can be expressed in terms of minmod,

$$\text{median}\,(x,y,z) = x + \text{minmod}\,(y-x, z-x). \tag{2.5}$$

The minmod function can be extended to any number of arguments. For k arguments, minmod $(z_1, \ldots, z_k)$ returns the smallest argument if all arguments are positive, the largest if all are negative, and zero otherwise. This function can be coded as

$$\text{minmod}\,(z_1,\ldots,z_k) = s\min(|z_1|,\ldots,|z_k|) \tag{2.6a}$$

where

$$s = \tfrac{1}{2}(\text{sgn}(z_1)+\text{sgn}(z_2))\left|\tfrac{1}{2}(\text{sgn}(z_1)+\text{sgn}(z_3))\ldots\tfrac{1}{2}(\text{sgn}(z_1)+\text{sgn}(z_k))\right|. \tag{2.6b}$$

Also denote by $I\,[z_1,\ldots,z_k]$ the interval $[\min(z_1,\ldots,z_k), \max(z_1,\ldots,z_k)]$.

We can now derive a simple condition that preserves monotonicity. At interface $j-1/2$, suppose the value $v^L_{j-1/2}$ lies between v_{j-1} and v_j:

$$v^L_{j-1/2} \in I\,[v_{j-1}, v_j]. \tag{2.7}$$

Next, for the interface $j+1/2$, denote

$$v^{UL} = v_j + \alpha(v_j - v_{j-1}) \tag{2.8}$$

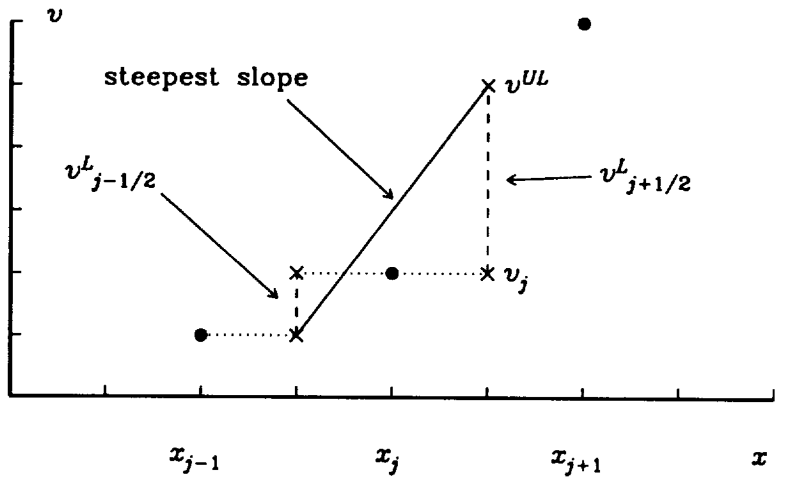

Figure 2.1: Monotonicity-preserving constraint (2.7) and (2.9).

where UL stands for upper limit, and $\alpha \geq 2$ (more on α momentarily). Suppose the value $v^L_{j+1/2}$ lies between v_j and v^{UL},

$$v^L_{j+1/2} \in I[v_j, v^{UL}]. \tag{2.9}$$

Then, after one stage via (1.6), the solution $w^{(1)}_j$ lies between v_{j-1} and v_j provided that the time step satisfies the condition

$$\sigma \leq 1/(1+\alpha). \tag{2.10}$$

Indeed, for increasing data, (2.7) and (2.9) imply that the steepest slope $v^{UL} - v_{j-1}$ satisfies $v^{UL} - v_{j-1} \leq (\alpha+1)(v_j - v_{j-1})$; therefore, (2.10) implies $v_{j-1} \leq w^{(1)}_j \leq v_j$. See Fig. 2.1.

Note that for parabolic reconstruction schemes, α is typically 2. For Runge-Kutta time stepping, we find that $\alpha = 4$ works well, while $\alpha = 2$ tends to cause stair-casing. With $\alpha = 4$, (2.10) leads to a CFL number $\sigma \leq 0.2$ but, in practice, $\sigma = 0.4$ still yields nonoscillatory results.

Next, assume that (2.7) and (2.9) hold for all j. Expression (2.7) with index j replaced by $j+1$ takes the form

$$v^L_{j+1/2} \in I[v_j, v_{j+1}]. \tag{2.11}$$

The above and (2.9) result in the condition that $v^L_{j+1/2}$ lies in the intersection of the two intervals $I[v_j, v_{j+1}]$ and $I[v_j, v^{UL}]$. One end of this intersection is v_j. The other is the median of v_j, v_{j+1}, and v^{UL}, and is denoted by v^{MP}. Using v_j as the pivot, v^{MP} can be expressed by using the minmod function,

$$v^{MP} = v_j + \text{minmod}\,[v_{j+1} - v_j, \alpha(v_j - v_{j-1})]. \tag{2.12}$$

Thus, (2.9) and (2.11) imply

$$v_{j+1/2}^L \in I\,[v_j, v^{MP}]. \tag{2.13}$$

The simplest way to satisfy this constraint is to replace the original $v_{j+1/2}^L$ by the median of $v_{j+1/2}^L$, v_j, and v^{MP}:

$$v_{j+1/2}^L \leftarrow \text{median}\,(v_{j+1/2}^L, v_j, v^{MP}). \tag{2.14}$$

Expression (2.14) preserves monotonicity in the following sense: under the CFL restriction (2.10) if the data $\{v_j\}$ are monotone, then after one stage, $\{w_j^{(1)}\}$ are also monotone. This fact follows because $w_j^{(1)}$ lies between v_{j-1} and v_j for all j.

The monotonicity-preserving property extends easily to the full scheme (1.5). Indeed, given monotone data $\{v_j\}$, we have just shown that $\{w^{(1)}\}$ are monotone provided the interface values are given by (2.14) and the above CFL restriction is satisfied. Since $\{w^{(1)}\}$ are monotone, the quantities $\{w^{(1)} + \sigma L(w^{(1)})\}$ are also monotone because they result from applying a single stage scheme to $\{w^{(1)}\}$. Next, for each j, $w_j^{(2)}$ is a combination of v_j and $(w^{(1)} + \sigma(L(w^{(1)}))_j$ with positive weights independent of j. Therefore, $\{w_j^{(2)}\}$ are monotone as well. Repeating this argument, it follows that $\{w_j^{(3)}\}$ are also monotone. Thus, if the data $\{v_j\}$ are monotone, and the interface values are obtained by (2.14), then the cell averages at the next time level $\{v_j^{n+1}\}$ are monotone under the CFL restriction (2.10).

The drawback of (2.14) is that near an extremum, it causes accuracy to degenerate to first-order. Figures 2.2(a) and 2.2(b) show the loss of accuracy caused by the constraints (2.11) and (2.9), respectively. Note that the data are on a parabola.

2.2 Accuracy-preserving constraint

To avoid the loss of accuracy, we enlarge the intervals in (2.11) and (2.9) in such a way that these intervals remain the same for monotone data but, near an extremum, these intervals are larger, and both contain the original $v_{j+1/2}^L$.

First, the interval in (2.11) is enlarged by adjoining the value v^{MD} defined below (MD stands for median). At interface $j + 1/2$, let v^{FL} and v^{FR} be the values extrapolated linearly from the left and right, respectively,

$$v^{FL} = v_j + \tfrac{1}{2}(v_j - v_{j-1}), \qquad v^{FR} = v_{j+1} + \tfrac{1}{2}(v_{j+1} - v_{j+2}). \tag{2.15}$$

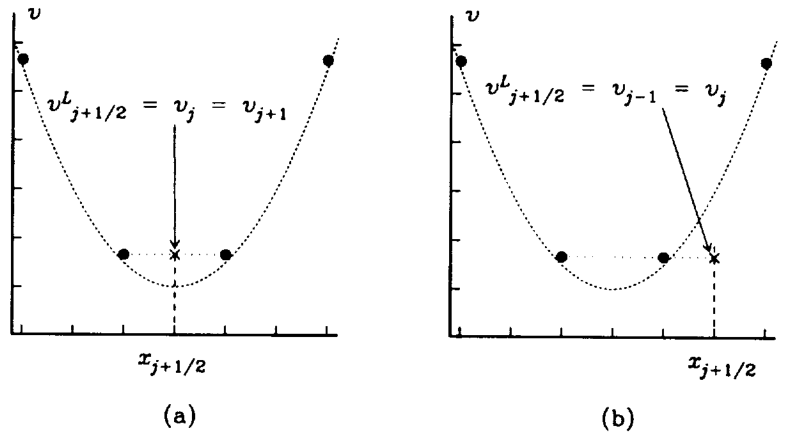

Figure 2.2: Loss of accuracy near extrema: (a) by (2.11) and (b) by (2.9).

With

$$v^{AV} = \tfrac{1}{2}(v_j + v_{j+1}), \tag{2.16}$$

where AV stands for average, set

$$v^{MD} = \text{median}\,(v^{AV}, v^{FL}, v^{FR}). \tag{2.17}$$

Constraint (2.11) is relaxed to

$$v^L_{j+1/2} \in I\,[v_j, v_{j+1}, v^{MD}]. \tag{2.18}$$

One can verify that if the four pieces of data $\{v_{j-1}, v_j, v_{j+1}, v_{j+2}\}$ are monotone, then, at the interface $j + 1/2$, v^{MD} lies between v_j and v_{j+1}, and the above constraint reduces to (2.11). Near an extremum as in the case of Fig. 2.3 (a), however, v^{MD} lies outside $I\,[v_j, v_{j+1}]$ and provides room so that the interval $I\,[v_j, v_{j+1}, v^{MD}]$ contains the original $v^L_{j+1/2}$.

The argument (2.15)-(2.18) conveys the idea. For the purpose of coding, it is more efficient if we employ the second differences. Set

$$d_j = v_{j-1} + v_{j+1} - 2v_j, \tag{2.19}$$

and

$$d^{MM}_{j+1/2} = \text{minmod}\,(d_j, d_{j+1}), \tag{2.20}$$

where MM stands for minmod. Then, since $v^{FL} = v^{AV} - \frac{1}{2}d_j$ and similarly for v^{FR}, it follows that

$$v^{MD} = v^{AV} - \tfrac{1}{2}d^{MM}_{j+1/2}. \tag{2.21}$$

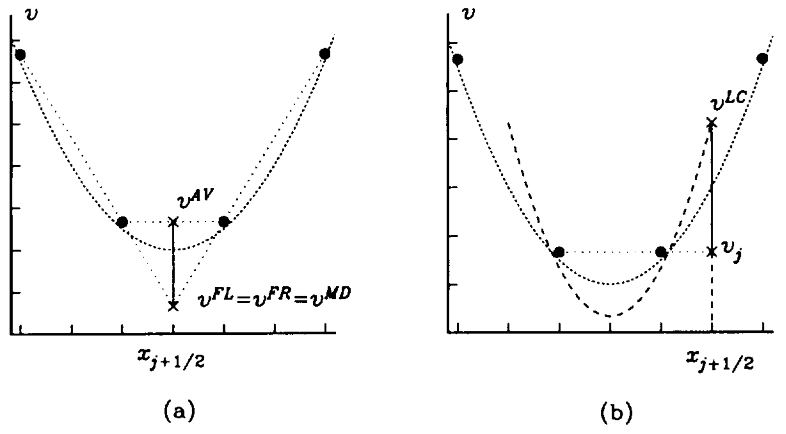

Figure 2.3: Enlarged monotonicity interval (a) by (2.18) and (b) by (2.23).

Next, the interval in (2.9) is enlarged by adjoining the value v^{LC} defined below (LC stands for large curvature). Consider the parabola p determined by the cell averages v_{j-1}, v_j, and the second difference d (a quantity similar to d_j). A straightforward calculation gives the value at $x_{j+1/2}$,

$$p(x_{j+1/2}) = v_j + \tfrac{1}{2}(v_j - v_{j-1}) + \tfrac{2}{3}d.$$

The parabola with $d = 2d^{MM}_{j-1/2}$ gives

$$v^{LC} = v_j + \tfrac{1}{2}(v_j - v_{j-1}) + \tfrac{4}{3}d^{MM}_{j-1/2}. \tag{2.22}$$

Constraint (2.9) is relaxed to

$$v^L_{j+1/2} \in I\,[v_j, v^{UL}, v^{LC}]. \tag{2.23}$$

Using the fact that $\alpha \geq 2$, one can verify that if the four pieces of data $\{v_{j-2}, v_{j-1}, v_j, v_{j+1}\}$ are monotone, then, at the interface $j + 1/2$, v^{LC} lies between v_j and v^{UL}. The above constraint therefore reduces to (2.9). Near an extremum as in the case of Fig. 2.3 (b), however, v^{LC} lies outside $I\,[v_j, v^{UL}]$ and provides room so that the interval $I\,[v_j, v^{UL}, v^{LC}]$ contains the original $v^L_{j+1/2}$.

In practice, we reduce the amount of room so that near a non-monotone discontinuity (such as a sawtooth profile), constraints (2.18) and (2.23) reduce to (2.11) and (2.9) respectively. This reduction can be accomplished by replacing $d^{MM}_{j+1/2}$ by

$$d^{M4}_{j+1/2} = \text{minmod}\,(4d_j - d_{j+1}, 4d_{j+1} - d_j, d_j, d_{j+1}). \tag{2.24}$$

To clarify the role of d^{M4}, assume that d_j and d_{j+1} are of the same sign. Then if $d_j/d_{j+1} < 1/4$ or $d_j/d_{j+1} > 4$, the above minmod of four arguments returns 0. Loosely put, when the second differences change substantially, then $d^{M4}_{j+1/2} = 0$, and the extended intervals reduce to the simple ones in (2.9) and (2.11). We thus replace expressions (2.21) and (2.22) for each interface $j + 1/2$ by

$$v^{MD} = v^{AV} - \tfrac{1}{2}d^{M4}_{j+1/2}, \tag{2.25}$$

$$v^{LC} = v_j + \tfrac{1}{2}(v_j - v_{j-1}) + \tfrac{4}{3}d^{M4}_{j-1/2}. \tag{2.26}$$

The intersection $[v^{\min}, v^{\max}]$ of the two intervals $I\,[v_j, v_{j+1}, v^{MD}]$ and $I\,[v_j, v^{UL}, v^{LC}]$ can be calculated by

$$v^{\min} = \max[\,\min(v_j, v_{j+1}, v^{MD}), \min(v_j, v^{UL}, v^{LC})\,], \tag{2.27a}$$

$$v^{\max} = \min[\,\max(v_j, v_{j+1}, v^{MD}), \max(v_j, v^{UL}, v^{LC})\,]. \tag{2.27b}$$

Finally, to ensure that $v^L_{j+1/2}$ lies in $[v^{\min}, v^{\max}]$, we replace $v^L_{j+1/2}$ by the median of $v^L_{j+1/2}$, $v^{\min}$, and $v^{\max}$:

$$v^L_{j+1/2} \leftarrow \text{median}\,(v^L_{j+1/2}, v^{\min}, v^{\max}). \tag{2.28}$$

The above limiting procedure preserves monotonicity and accuracy. In addition, for most cells in smooth regions, the original $v^L_{j+1/2}$ satisfies constraint (2.13) a priori. In this case, the limiting procedure (2.24)-(2.28) does not alter the original $v^L_{j+1/2}$. As a result, we can use (2.13) to detect such cells and bypass the limiting procedure altogether. The condition that the original $v^L_{j+1/2}$ lies in the interval $I\,[v_j, v^{MP}]$ is equivalent to $(v^L_{j+1/2} - v_j)(v^L_{j+1/2} - v^{MP}) \leq 0$. In practice, this condition is coded with a tolerance value of $\epsilon = 10^{-10}$:

$$(v^L_{j+1/2} - v_j)(v^L_{j+1/2} - v^{MP}) \leq \epsilon. \tag{2.29}$$

We summarize the computation of the interface value below.

Algorithm for the interface value. *Suppose the cell averages $\{v_j\}$ are given, and $\alpha \geq 0$. For each interface $j + 1/2$, calculate the original value $v^L_{j+1/2}$ from (2.1), and v^{MP} from (2.12). If (2.29) holds, then $v_{j+1/2} = v^L_{j+1/2}$, and we move on to the next interface. Otherwise, calculate d_{j-1}, d_j, d_{j+1} from (2.19), $d^{M4}_{j+1/2}$ and $d^{M4}_{j-1/2}$ from (2.24), v^{UL} from (2.8), v^{AV} from (2.16), v^{MD} and v^{LC} from (2.25) and (2.26), and $v^{\min}, v^{\max}$ from (2.27). Finally, calculate $v^L_{j+1/2}$ from (2.28), and the interface value is then $v_{j+1/2} = v^L_{j+1/2}$.*

2.3 Remarks

Note that constraint (2.13) on the interface value is a sufficient condition for monotone data to remain monotone under Runge-Kutta time stepping. It may be viewed as the analogue of Van Leer's constraint [19] which provides the same type of condition for monotonicity under exact time evolution. Also note that the geometric framework, the use of the median function, and v^{MD} were introduced by Huynh in [5].

For advection with $a < 0$, the interface value $v^R_{j+1/2}$ is obtained by reflecting the above expressions about $x_{j+1/2}$. To be specific, the reconstruction algorithm is the same with $\{v_{j-2}, v_{j-1}, v_j, v_{j+1}, v_{j+2}\}$ replaced by $\{v_{j+3}, v_{j+2}, v_{j+1}, v_j, v_{j-1}\}$ respectively. Next, the stencil for computing both $v^L_{j+1/2}$ and $v^R_{j+1/2}$ consists of the six points $\{v_{j-2}, ... v_{j+3}\}$. Therefore, we could define both $v^L_{j+1/2}$ and $v^R_{j+1/2}$ by the quintic fit of all six cell averages without enlarging the stencil (in the case of Euler equations). The corresponding limited scheme, however, is prone to stair-casing.

Higher-order schemes can be derived using larger stencils. With the same stencil as the mth-order ENO scheme, a $(2m-1)$th-order scheme can be obtained. For example, for $m = 4$, we have the seven-point formula

$$\begin{aligned} v^L_{j+1/2} &= (-3v_{j-3} + 25v_{j-2} - 101v_{j-1} + 319v_j \\ &+214v_{j-1} - 38v_{j+2} + 4v_{j+3})/420, \end{aligned} \tag{2.30a}$$

and, for $m = 5$, the nine-point formula

$$\begin{aligned} v^L_{j+1/2} &= (4v_{j-4} - 41v_{j-3} + 199v_{j-2} - 641v_{j-1} + 1879v_j \\ &+1375v_{j+1} - 305v_{j+2} + 55v_{j+3} - 5v_{j+4})/2520. \end{aligned} \tag{2.30b}$$

The same limiting can be employed for these original interface values. The resulting schemes achieve high spatial accuracy but remain third-order in time. For the fourth- and fifth-order Runge-Kutta methods, in order to preserve monotonicity we need the calculation of the time-reversed operator $\tilde{L}$ [15], which is beyond the scope of this paper.

The above reconstruction depends continuously on the data in the sense that a small change in the data causes a small change in the interface value. This property is shared by WENO (but not by ENO) reconstruction.

3. Extensions

In this section, we describe the extensions of the above schemes to the Euler equations. While these extensions are standard, the monotonicity-preserving property may not hold because the equations are nonlinear. Nevertheless, the numerical solutions obtained below are generally nonoscillatory.

3.1 Euler system in one dimension

The Euler equations of gas dynamics for a polytropic gas can be written as

$$\mathbf{u}_t + \mathbf{f}(\mathbf{u})_x = 0 \tag{3.1}$$

where

$$\begin{aligned} \mathbf{u} &= (\rho, \rho u, E)^T, \\ \mathbf{f}(\mathbf{u}) &= u\mathbf{u} + (0, p, up)^T, \\ p &= (\gamma - 1)(E - \tfrac{1}{2}\rho u^2). \end{aligned} \tag{3.2}$$

Here, T represents the transpose; ρ, u, p, and E are the density, velocity, pressure, and total energy respectively; and $\gamma = 1.4$, is the ratio of specific heats. The speed of sound c is given by $(\gamma p/\rho)^{1/2}$.

The eigenvalues of the Jacobian matrix $\mathbf{A}(\mathbf{u}) = \partial \mathbf{f}/\partial \mathbf{u}$ are $u - c$, u and $u + c$. The matrices of left and right eigenvectors of $\mathbf{A}$ are needed in the reconstruction. These are given by ([3])

$$\mathbf{L} = \begin{pmatrix} b_2/2 + u/2c & -b_1 u/2 - 1/2c & b_1/2 \\ 1 - b_2 & b_1 u & -b_1 \\ b_2/2 - u/2c & -b_1 u/2 + 1/2c & b_1/2 \end{pmatrix} \tag{3.3}$$

and

$$\mathbf{R} = \begin{pmatrix} 1 & 1 & 1 \\ u - c & u & u + c \\ H - uc & \tfrac{1}{2}u^2 & H + uc \end{pmatrix} \tag{3.4}$$

where $b_1 = (\gamma - 1)/c^2$ and $b_2 = u^2 b_1/2$, $H = c^2/(\gamma - 1) + \frac{1}{2}u^2$.

Integrating (3.1) over the cell $[x_{j-1/2}, x_{j+1/2}]$ yields

$$\frac{d\bar{\mathbf{u}}_j}{dt} + \frac{1}{h}\left[\mathbf{f}(\mathbf{u}(x_{j+1/2}, t)) - \mathbf{f}(\mathbf{u}(x_{j-1/2}, t))\right] = 0, \tag{3.5}$$

where $\bar{\mathbf{u}}_j(t)$ are the cell averages. The first step in calculating $\mathbf{f}_{j+1/2} \approx \mathbf{f}(\mathbf{u}(x_{j+1/2}, t^n))$ is to reconstruct $\mathbf{u}$ on both sides of the interface $x_{j+1/2}$.

It is well known that the reconstruction is best carried out in local characteristic variables [3]. If $\{\mathbf{v}_j\}$ are the approximations to the cell averages at time level n, these local characteristic variables for the cell $[x_{j-1/2}, x_{j+1/2}]$ are given by

$$\mathbf{w}_k = \mathbf{L}(\mathbf{v}_j)\mathbf{v}_{j+k}, \quad \text{for} \quad k = -2, 2. \tag{3.6}$$

The scalar reconstruction algorithm is now applied to obtain point values $\mathbf{w}^R_{-1/2}$ and $\mathbf{w}^L_{1/2}$ at $x_{j+1/2}$ and $x_{j-1/2}$, respectively. The corresponding conservative variables $\mathbf{v}^L_{j+1/2}$ and $\mathbf{v}^R_{j-1/2}$ are obtained by the inverse of (3.6)

$$\mathbf{v}^L_{j+1/2} = \mathbf{R}(\mathbf{v}_j)\mathbf{w}^L_{1/2}, \qquad \mathbf{v}^R_{j-1/2} = \mathbf{R}(\mathbf{v}_j)\mathbf{w}^R_{-1/2}. \tag{3.7}$$

At each interface $j + 1/2$, the two values $\mathbf{v}^L_{j+1/2}$ and $\mathbf{v}^R_{j+1/2}$ are used to calculate $\mathbf{f}_{j+1/2}$ via Roe's flux-difference splitting [12]. This splitting is implemented here with Huynh's entropy fix [6].

Equation(3.5) is then integrated by the Runge-Kutta scheme (1.5). The time step is given in terms of the CFL number σ by

$$\Delta t = \frac{\sigma \quad h}{\mathrm{Max}_j(|u_j| + c_j)}. \tag{3.8}$$

Note that the extension described above is standard, but it does not take advantage of the fact that our reconstruction algorithm leaves the left and right interface values unchanged in smooth regions away from extrema. In these regions, the reconstruction applied to the local characteristic variables yields a result identical to formula (2.1) applied to the conserved variables $\{\mathbf{v}_j\}$. Thus, the expense of characteristic decomposition may be avoided for such regions if they could be detected in a simple manner as in [6]. However, we do not pursue this approach here.

3.2 Euler system in two dimensions

An immediate extension of the above scheme to multi-dimensions can be accomplished in the same manner as the finite difference ENO schemes of Shu and Osher [15] [16]. The idea is to avoid calculating the mixed derivatives of the reconstruction from cell averages by applying the reconstruction directly on point values of the fluxes. The reconstruction then reduces to two one-dimensional reconstructions along the coordinate lines. The same Runge-Kutta scheme (1.5) is used to integrate the equations in time. Here we have chosen the Lax-Friedrichs version (ENO-LLF) in our numerical experiments. The 2D extension of the schemes derived here are obtained by substituting our algorithms for reconstruction in place of the scalar one-dimensional ENO reconstruction therein. Coding aspects of these schemes can be found in [17].

4. Numerical experiments

For simplicity, we present numerical results only for the scheme combining the quartic fit (2.1) and the accuracy-preserving constraint (2.28). We refer to this scheme as the MP5 scheme (MP for monotonicity preserving). Some comparisons with ENO3 and WENO5 schemes are provided. The three schemes MP5, WENO5 and ENO3 have the same stencil, and the first two are spatially fifth-order accurate. Listings of the these three reconstruction procedures in Fortran are given in Appendix A. Also note that we employ only uniform meshes and, unless otherwise stated, the CFL number is 0.4.

All computations are carried out on a 100 MHz R4000 SGI Indigo Workstation, with $\epsilon = 10^{-10}$ and $\alpha = 4$. In all cases, the compiler options -r8 -O3 were used. We have observed that computing times vary widely depending on the hardware and compiler options used. Therefore, computing times are to be viewed only as an approximate measure of the efficiency of the various schemes. The computing time for the scheme with constant reconstruction and the three-stage Runge-Kutta time stepping is also provided as a reference. Since the reconstruction is trivial for this scheme, this computing time reflects the cost of all other calculations except reconstruction.

4.1 Advection of a smooth function

We solve (1.1) with $u(x,0) = \sin(\pi x)^4$ with periodic boundaries. We are particularly interested in the behavior of the errors of the cell averages under mesh refinement. Since the function is smooth, the most accurate and efficient scheme with the given stencil of five cell averages is the unlimited scheme (2.1). We compare the results of the WENO5 and MP5 schemes to this unlimited scheme for $\sigma = 0.05$ in Table 2 (a) and $\sigma = 0.4$ in Table 2 (b). The results from ENO3 are less accurate and are not shown.

Note that the errors obtained by the unlimited scheme and those by the MP5 scheme are essentially identical. This confirms that the limiting procedure leaves the quartic fit unchanged at smooth extrema. At low CFL numbers, the MP5 scheme approaches the theoretical order of accuracy of five as can be seen in Table 2(a). In both cases, the MP5 scheme compares favorably with the WENO5 scheme in both accuracy and efficiency.

Table - 2: Advection of $\sin(\pi x)^4$ by several schemes. $t = 2$, $\Delta x = 2/N$, CPU time quoted is for all grids.

Table 2(a): $\Delta t/\Delta x = 0.05$, CPU time for constant reconstruction: 4.30 sec.						
Scheme	N	L_∞ error	L_∞ order	L_1 error	L_1 order	CPU time - sec.
WENO5	16	2.39(-1)	-	1.07(-1)	-	22.33
	32	3.45(-2)	2.79	1.73(-2)	2.62	
	64	3.51(-3)	3.29	1.75(-3)	3.31	
	128	3.44(-4)	3.35	8.88(-5)	4.30	
	256	1.15(-5)	4.90	2.54(-6)	5.13	
MP5	16	1.17(-1)	-	8.05(-2)	-	11.86
	32	1.40(-2)	3.06	8.14(-3)	3.31	
	64	5.05(-4)	4.80	3.01(-4)	4.76	
	128	1.63(-5)	4.96	9.74(-6)	4.95	
	256	5.25(-7)	4.95	3.14(-7)	4.96	
Unlim.	16	1.17(-1)	-	8.05(-2)	-	6.18
	32	1.40(-2)	3.06	8.14(-3)	3.30	
	64	5.05(-4)	4.80	3.01(-4)	4.76	
	128	1.63(-5)	4.96	9.74(-6)	4.95	
	256	5.25(-7)	4.95	3.14(-7)	4.96	

Table 2(b): $\Delta t/\Delta x = 0.4$, CPU time for constant reconstruction: 1.05 sec.						
Scheme	N	L_∞ error	L_∞ order	L_1 error	L_1 order	CPU time - sec.
WENO5	16	2.39(-1)	-	1.07(-1)	-	3.30
	32	3.74(-2)	2.68	1.87(-2)	2.52	
	64	3.26(-3)	3.52	1.79(-3)	3.39	
	128	3.00(-4)	3.44	1.11(-4)	4.01	
	256	1.25(-5)	4.58	6.17(-6)	4.17	
MP5	16	1.21(-1)	-	8.01(-2)	-	2.02
	32	1.77(-2)	2.77	1.03(-2)	2.96	
	64	1.10(-3)	4.01	6.15(-4)	4.06	
	128	9.50(-5)	3.54	5.05(-5)	3.61	
	256	1.04(-5)	3.19	5.42(-6)	3.22	
Unlim.	16	1.21(-1)	-	8.01(-2)	-	1.29
	32	1.77(-2)	2.77	1.03(-2)	2.96	
	64	1.10(-3)	4.01	6.17(-4)	4.06	
	128	9.50(-5)	3.54	5.04(-5)	3.61	
	256	1.04(-5)	3.19	5.42(-6)	3.22	

4.2 Advection of a piecewise continuous function

Next, the initial condition is given by

$$\begin{aligned}
u_0(x) &= \exp(-\log(2)(x+0.7)^2/0.0009) && \text{if } -0.8 \le x \le -0.6,\\
u_0(x) &= 1 && \text{if } -0.4 \le x \le -0.2,\\
u_0(x) &= 1 - |10(x-0.1)| && \text{if } 0 \le x \le 0.2,\\
u_0(x) &= [1 - 100(x-0.5)^2]^{1/2} && \text{if } 0.4 \le x \le 0.6,\\
u_0(x) &= 0 && \text{otherwise.}
\end{aligned} \tag{4.1}$$

This initial condition includes a Gaussian wave, a square wave, a triangular wave, and a semi-ellipse. We use 200 cells with $\sigma = 0.4$. The solutions at $t = 2$ (after one period or 200 cells) and $t = 20$ (ten periods) are shown in Figs. 4.1 and 4.2 respectively. The solid line represents the exact solution. Also shown are the computing times of the various schemes. Again, note that the MP5 solutions compare well with those by ENO3 and WENO5 schemes.

Resolution at discontinuities can be enhanced by using steepening techniques as in [4], [21], and [6]. These techniques are expensive and, while they are effective in one dimension, it is still not clear how well they perform in multi-dimensions. Here, we will limit our study to the base schemes only.

4.3 Euler system in one dimension

In the following three problems, the CFL number is 0.4, and the spatial domain is $[-1, 1]$. For the initial conditions, unless otherwise stated, the subscript L denotes $-1 \le x \le 0$, and R, $0 < x \le 1$. The final time is denoted by t_f, and the total number of cells, by N.

1. *Sod's problem [13]*

$$\begin{aligned}
(\rho_L, u_L, p_L) &= (1, 0, 1),\\
(\rho_R, u_R, p_R) &= (0.125, 0, 0.1),\\
t_f &= 0.4,\\
N &= 100.
\end{aligned}$$

Since this problem starts from a singularity, smaller time steps are used initially as described in [6]. The density field from the MP5 scheme is shown in Fig. 4.3. Note that the contact discontinuity and the shock are resolved with high resolution.

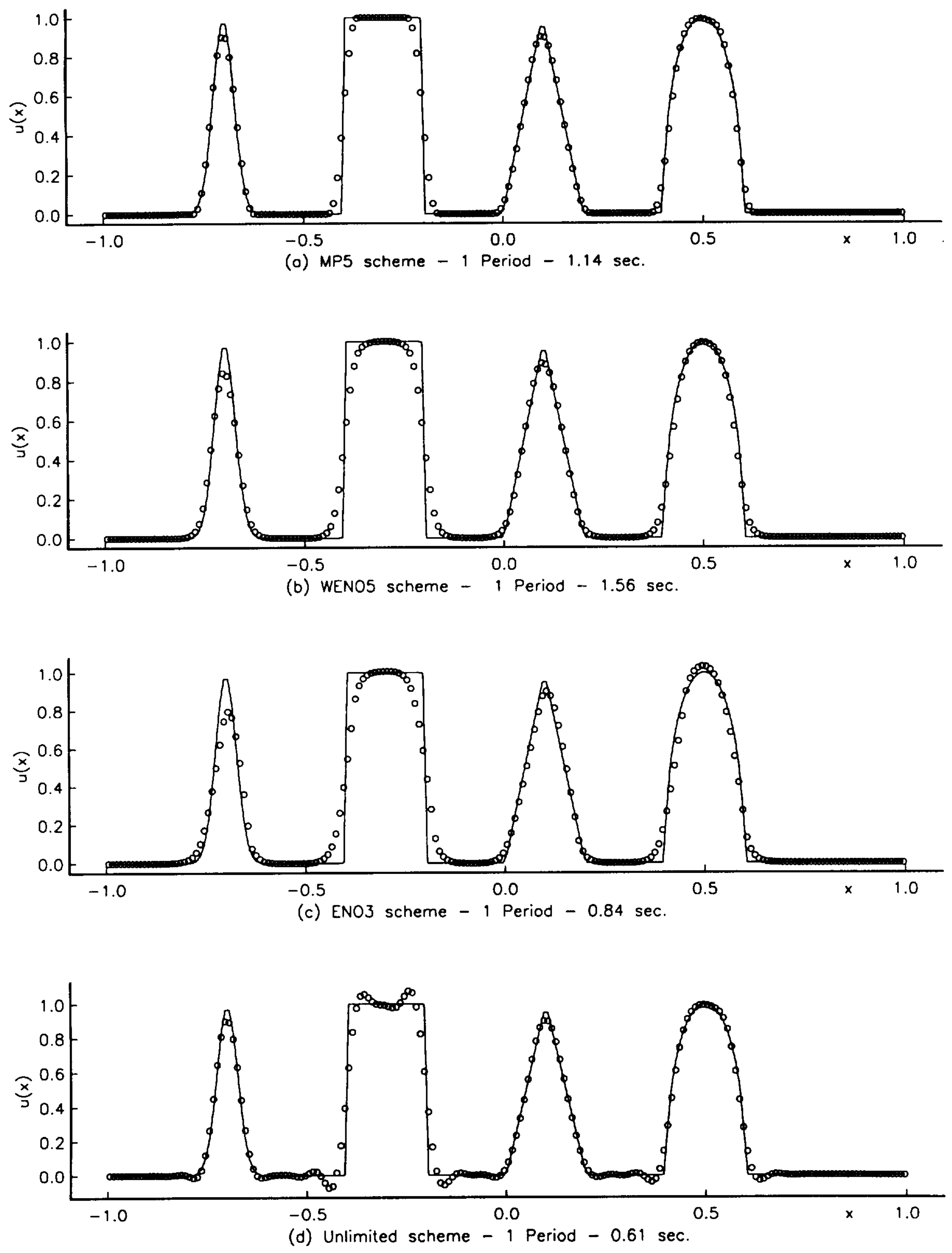

Figure 4.1: Advection over a period by (a) MP5, (b) WENO5, (c) ENO3, (d) Unlimited scheme $\Delta t/\Delta x = 0.4$, $\Delta x = 2/200$, $t = 2$, CPU time for constant reconstruction = 0.63 sec.

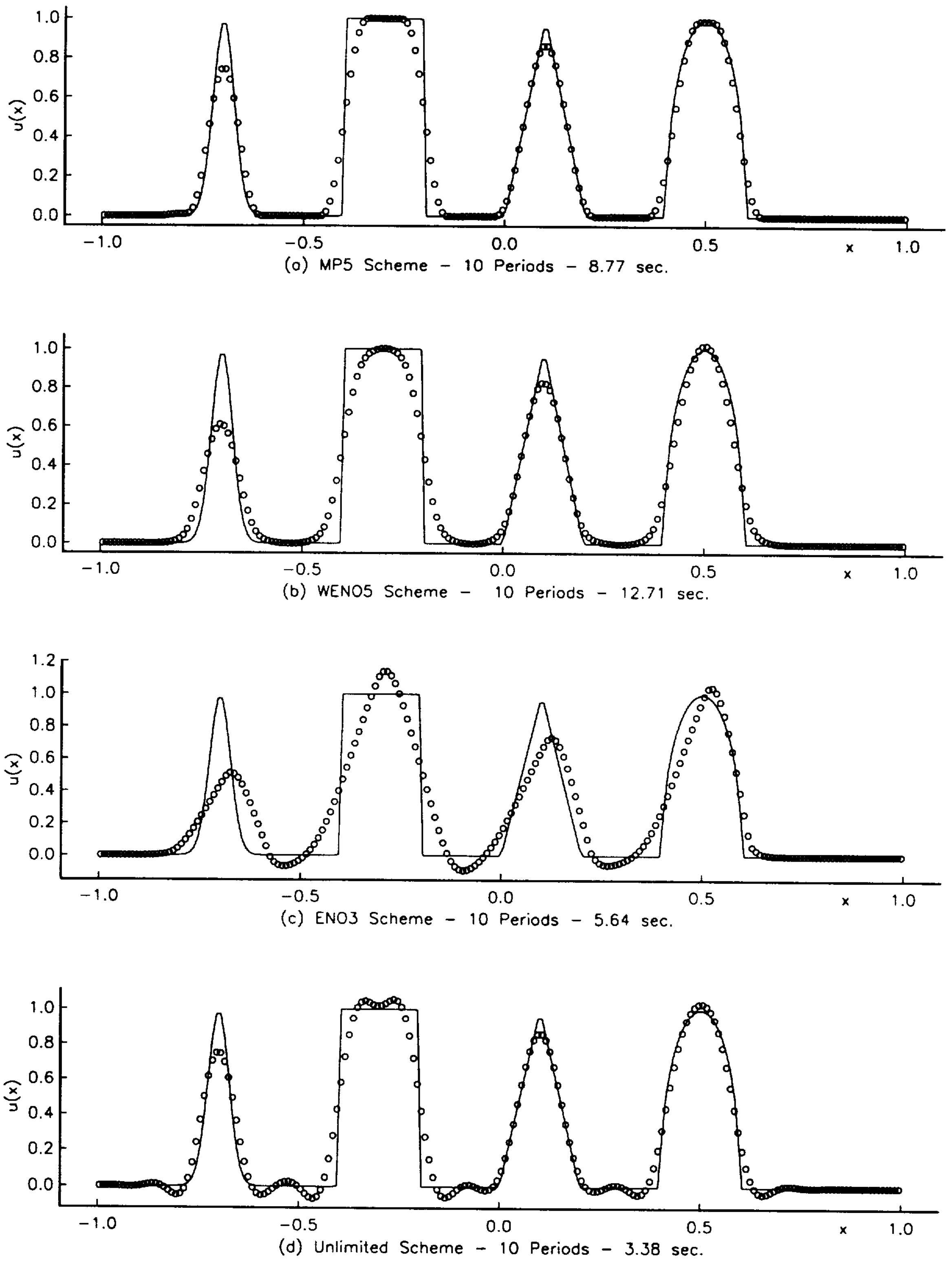

Figure 4.2: Advection over 10 periods by (a) MP5, (b) WENO5, (c) ENO3, (d) Unlimited scheme $\Delta t/\Delta x = 0.4$, $\Delta x = 2/200$, $t = 20$, CPU time for constant reconstruction = 2.69 sec.

2. *Lax's problem [10]*

$$\begin{aligned}(\rho_L, u_L, p_L) &= (0.445, 0.698, 3.528),\\ (\rho_R, u_R, p_R) &= (0.5, 0, 0.571),\\ t_f &= 0.32,\\ N &= 100.\end{aligned}$$

The density field from the MP5 scheme is shown in Fig. 4.4. Again, the contact discontinuity and the shock are well-resolved.

3. *Shu's problem [14]*

In this problem, a moving shock wave interacts with a density disturbance and generates a flow field with both smooth structure and discontinuities. Here, L stands for $-1 \le x \le -0.8$, and R, $-0.8 \le x \le 1$. The initial conditions, final time, and number of mesh points are

$$\begin{aligned}(\rho_L, u_L, p_L) &= (3.857143, 2.629369, 10.3333),\\ (\rho_R, u_R, p_R) &= (1 + 0.2\sin(5\pi x), 0, 1),\\ t_f &= 0.36,\\ N &= 300.\end{aligned}$$

Since the exact solution is not known, the solution by ENO3 with 800 cells is used in its place.

The results of MP5 and WENO5 are shown in Fig. 4.5. The MP5 scheme captures the shock with high resolution and resolves all local extrema accurately.

4.1 Euler system in two dimensions

We present results for two well known problems.

1. *Oblique shock reflection [1]*

The domain $[0, 4] \times [0, 1]$ is covered by a uniform mesh of 60×20 cells. The boundary conditions are: at the bottom, solid boundary; at the right, supersonic outflow; at the left, the conditions are fixed with

$$(\rho, u, v, p) = (1, 2.9, 0, 1/\gamma);$$

and at the top,

$$(\rho, u, v, p) = (1.69997, 2.61934, -0.50632, 1.52819).$$

Under these conditions, an oblique shock forms from the top left corner and is reflected by the bottom boundary. Initially, flow conditions at the left boundary are set throughout the whole domain. After 10,000 iterations, the solution essentially reaches steady state. The residual drops roughly 2 orders of magnitude for MP5, WENO5, and ENO3, while for the minmod and first-order upwind schemes the residual drops to machine zero. (Note that the minmod scheme is defined by $\mathbf{w}^L_{1/2} = \mathbf{w}_0 + \frac{1}{2}\text{minmod}\,(\mathbf{w}_1 - \mathbf{w}_0, \mathbf{w}_0 - \mathbf{w}_{-1})$, where $\mathbf{w}$ is the characteristic flux in this case.)

The pressure along the line $y = 0.55$ (j=11) is shown in Fig. 4.6. Concerning accuracy, it can be seen that the higher-order schemes have small oscillations about the exact solution. These oscillations are reduced on finer grids for all three schemes. Note that the MP5 scheme yields a highly accurate solution.

The computing times of the various schemes are also shown in Fig. 4.6. The first-order and minmod schemes are coded here with the local characteristic decompositions over the full five-point stencil. This first-order scheme represents the most efficient reconstruction in this framework, and its CPU time reflects the overhead of the characteristic decomposition. It can be seen from Fig. 4.6 that the overhead is more than half of the total computing time. In other words, unlike the case of advection, the computing time of the reconstruction step for the Euler equations is less than one third of the total time.

2. *Double Mach reflection [20]*

The computational domain is $[0, 4] \times [0, 1]$. The reflecting wall is from $(1/6, 0)$ to $(4, 0)$. Initially, a Mach 10 shock is incident on this wall at $(1/6, 0)$ making an angle of sixty degrees with the x-axis. To the right of the shock is undisturbed fluid of uniform pressure 1 and density 1.4. To the left of the shock, the conditions are

$$(\rho, u, v, p) = (8.0, 7.1447, -4.125, 116.5).$$

As the shock reflects off the wall, a diffraction pattern is formed. The final time is $t_f = 0.2$. A detailed description of the problem and various solutions can be found in [20].

The boundary conditions are: at the bottom, from $(0,0)$ to $(1/6,0)$, linear extrapolation; from $(1/6,0)$ to $(4,0)$, solid boundary; at the right, linear extrapolation; at the left, supersonic inflow; at the top, time-dependent conditions determined by the exact motion of the Mach 10 shock.

The MP5 and WENO5 solutions, obtained using a 240×60 mesh, are shown in Fig. 4.7. It can be seen that both schemes capture all the significant features of the solution such as the two Mach stems and the wall jet.

5. Conclusions

A new class of high-order schemes for the numerical solution of hyperbolic conservation laws was introduced. The key feature of these schemes is the reconstruction procedure which combines an accurate interface formula with a monotonicity-preserving constraint. The constraint is designed to preserve accuracy and to work well with Runge-Kutta time stepping. It is shown that, for advection, if the data are monotone, then the solution is also monotone under a time step restriction. Numerical experiments confirm that the resulting scheme is accurate in smooth regions, resolves discontinuities with high resolution, and is also efficient. The new scheme compares favorably with state-of-the-art schemes such as ENO3 and WENO5.

Appendix-A

In this appendix, we give the listings of the three higher-order reconstruction algorithms in Fortran. V(J) are the cell averages v_j and VL(J) are the computed interface values $v^L_{j+1/2}$. DMM(X,Y) is the minmod function of two arguments while DM4(W,X,Y,Z) is the minmod function of four arguments.

```
c-MP5 RECONSTRUCTION
c---DMM(X,Y) = 0.5*(SIGN(1.,X) + SIGN(1.,Y))*
   &        MIN(ABS(X),ABS(Y))
c---DM4(W,X,Y,Z) = 0.125*(SIGN(1.,W) + SIGN(1.,X))*
   &        ABS( (SIGN(1.,W) + SIGN(1.,Y))*
   &             (SIGN(1.,W) + SIGN(1.,Z)) )
   &        *MIN(ABS(W),ABS(X),ABS(Y),ABS(Z))
c
      B1 = 0.016666666667
      B2 = 1.333333333333
      ALPHA = 4.
      EPSM = 1.E-10
c
     VOR = B1*(2.*V(J-2)-13.*V(J-1)
   & + 47.*V(J) + 27.*V(J+1)
   & - 3.*V(J+2) )
     VMP= V(J) +  DMM(V(J+1)-V(J),ALPHA*(V(J)-V(J-1)))
     IF((VOR-V(J))*(VOR-VMP).LE. EPSM) THEN
     VL(J) = VOR
     ELSE
     ELSE
C
     DJM1 = V(J-2)-2.*V(J-1)+ V(J  )
     DJ   = V(J-1)-2.*V(J  )+ V(J+1)
     DJP1  = V(J  )-2.*V(J+1)+ V(J+2)
     DM4JPH= DM4(4.*DJ-DJP1,4.*DJP1-DJ,DJ,DJP1)
     DM4JMH= DM4(4.*DJ-DJM1,4.*DJM1-DJ,DJ,DJM1)
     VUL = V(J) + ALPHA*(V(J)-V(J-1))
     VAV = 0.5*(V(J) + V(J+1))
     VMD = VAV - 0.5*DM4JPH
     VLC = V(J) + 0.5*(V(J)-V(J-1)) + B2*DM4JMH
     VMIN = MAX(MIN(V(J),V(J+1),VMD),
   &          MIN(V(J),VUL,VLC))
     VMAX = MIN(MAX(V(J),V(J+1),VMD),
   &          MAX(V(J),VUL,VLC))
     VL(J) = VOR + DMM(VMIN-VOR,VMAX-VOR)
     ENDIF

C-WENO5 RECONSTRUCTION
```

```
      EPSW = 1.E-6
      B1 = 1.08333333333
      B2 = 0.166666666667
      DJM1 =  V(J-2)-2.*V(J-1)+     V(J  )
      EJM1 =  V(J-2)-4.*V(J-1)+3.*V(J  )
      DJ   =  V(J-1)-2.*V(J  )+     V(J+1)
      EJ   =  V(J-1)-     V(J+1)
      DJP1 =  V(J  )-2.*V(J+1)+     V(J+2)
      EJP1 = 3.*V(J)-4.*V(J+1)+V(J+2)
c
      DIS0 = B1*DJM1*DJM1 + 0.25*EJM1*EJM1 + EPSW
      DIS1 = B1*DJ*DJ     + 0.25*EJ*EJ + EPSW
      DIS2 = B1*DJP1*DJP1 + 0.25*EJP1*EJP1 + EPSW
C
     Q30 =  2.*V(J-2)-7.*V(J-1)+ 11.*V(J  )
     Q31 =       -V(J-1)+5.*V(J  )+  2.*V(J+1)
     Q32 =  2.*V(J  )+5.*V(J+1)       -V(J+2)
C
     D01 = DIS0/DIS1
     D02 = DIS0/DIS2
     A1BA0 = 6.*D01*D01
     A2BA0 = 3.*D02*D02
     W0 = 1./(1. + A1BA0 + A2BA0)
     W1 = A1BA0*W0
     W2 = 1. - W0 - W1
     VL(J) = B2*( W0*Q30 + W1*Q31 + W2*Q32 )
C
C-ENO3 RECONSTRUCTION
      DATA CM(1,1),CM(1,2),CM(1,3)/2.,-7.,11./
      DATA CM(2,1),CM(2,2),CM(2,3)/-1.,5.,2./
      DATA CM(3,1),CM(3,2),CM(3,3)/2.,5.,-1./
          B1 = 0.166666666667
          SP = ABS( V(J+1) - V(J  ) )
          SM = ABS( V(J  ) - V(J-1) )
          DJ   = ABS( V(J+1) - 2.*V(J  ) + V(J-1) )
C
          IF(2.*SP .GT. SM) THEN
              DJM1 = ABS( V(J  ) - 2.*V(J-1) + V(J-2) )
              IF(DJ.GT.2.*DJM1) THEN
              ID = 1
              ELSE
              ID = 2
              ENDIF
          ELSE
              DJP1 = ABS( V(J+2)-2.*V(J+1) + V(J  ) )
              IF(2.*DJP1 .GT. DJ) THEN
              ID = 2
              ELSE
              ENDIF
          ENDIF
C
          VL(J)= ( CM(ID,1)*V(J-3+ID) + CM(ID,2)*V(J-2+ID) +
    &      CM(ID,3)*V(J-1+ID) )*B1
```

Acknowledgments

The first author was supported by NASA Lewis Research Center under Contract NAS3-27186 with Dr. I. Lopez as monitor.

References

[1] P. Colella, *J. Comput. Phys.*, **87**, 171 (1990).

[2] P. Colella and P. Woodward, *J. Comput. Phys.*, **54**, 174, (1984).

[3] A. Harten, B. Engquist, S. Osher, and S. R. Chakravarthy, *J. Comput. Phys.*, **71**, 231 (1987).

[4] A. Harten, *J. Comput. Phys.*, **83**, 148 (1989).

[5] H. T. Huynh, *SIAM J. Numer. Anal.*, **30**, 57 (1993).

[6] H. T. Huynh, *SIAM J. Numer. Anal.*, **32**, 1565 (1995).

[7] H. T. Huynh, AIAA 95-1739, *AIAA 12th. Computational Fluid Dynamics Conference, San Diego, CA, (1995).*

[8] H. T. Huynh, *International Conference on Numerical Methods in Fluid Dynamics, Monterey, CA, (1996).*

[9] G-S. Jiang and C-W. Shu, *J. Comput. Phys.*, **126**, 202 (1996).

[10] P. D. Lax, *Commun. Pure Appl. Math.*, **7**, 159 (1954).

[11] X. Liu, S. Osher, and T. Chan, *J. Comput. Phys.*, **115**, 200 (1994).

[12] P. L. Roe, *J. Comput. Phys.*, **43**, 357 (1981).

[13] G. A. Sod, *J. Comput. Phys.*, **27**, 1 (1978).

[14] C.-W. Shu, *J. Sci. Comput.*, **5**, 127 (1990).

[15] C.-W. Shu and S. Osher, *J. Comput. Phys.*, **77**, 439 (1988).

[16] C.-W. Shu and S. Osher, *J. Comput. Phys.*, **83**, 32 (1989).

[17] C.-W. Shu, G. Erlebacher, T. A. Zang, D. Whitaker, and S. Osher, *ICASE Report 91-38, Hampton, Virginia (1991).*

[18] A. Suresh, AIAA 95-1755, *AIAA 12th. Computational Fluid Dynamics Conference, San Diego, CA, (1995).*

[19] B. van Leer, *J. Comput. Phys.*, **14**, 361 (1974).

[20] P. Woodward and P. Colella, *J. Comput. Phys.*, **54**, 115 (1984).

[21] H. Yang, *J. Comput. Phys.*, **89**, 125 (1990).

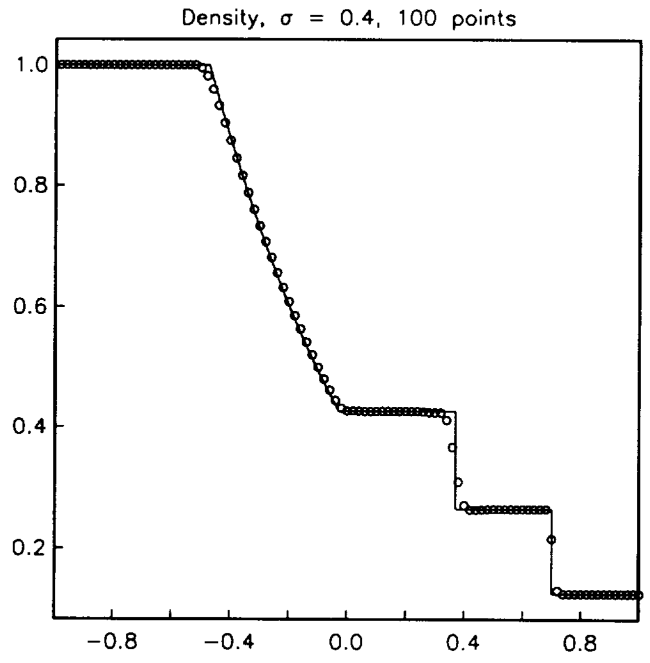

Figure 4.3: Sod's problem with the MP5 scheme.

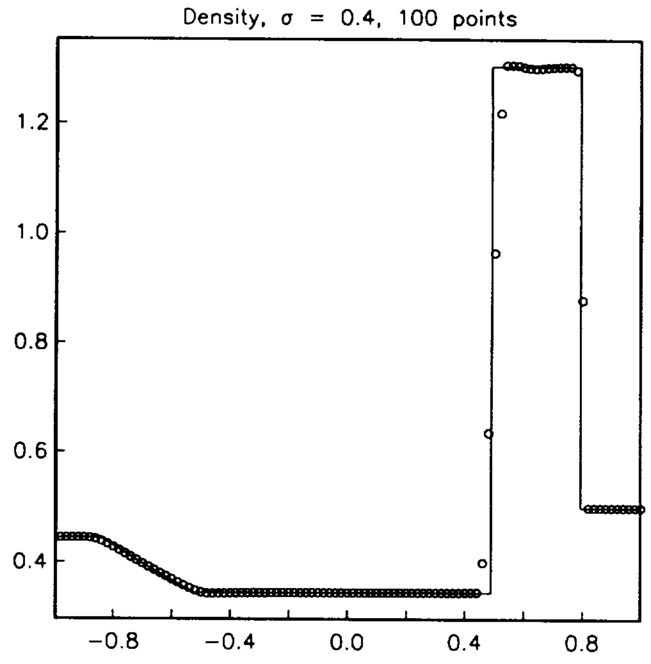

Figure 4.4: Lax's problem with the MP5 scheme.

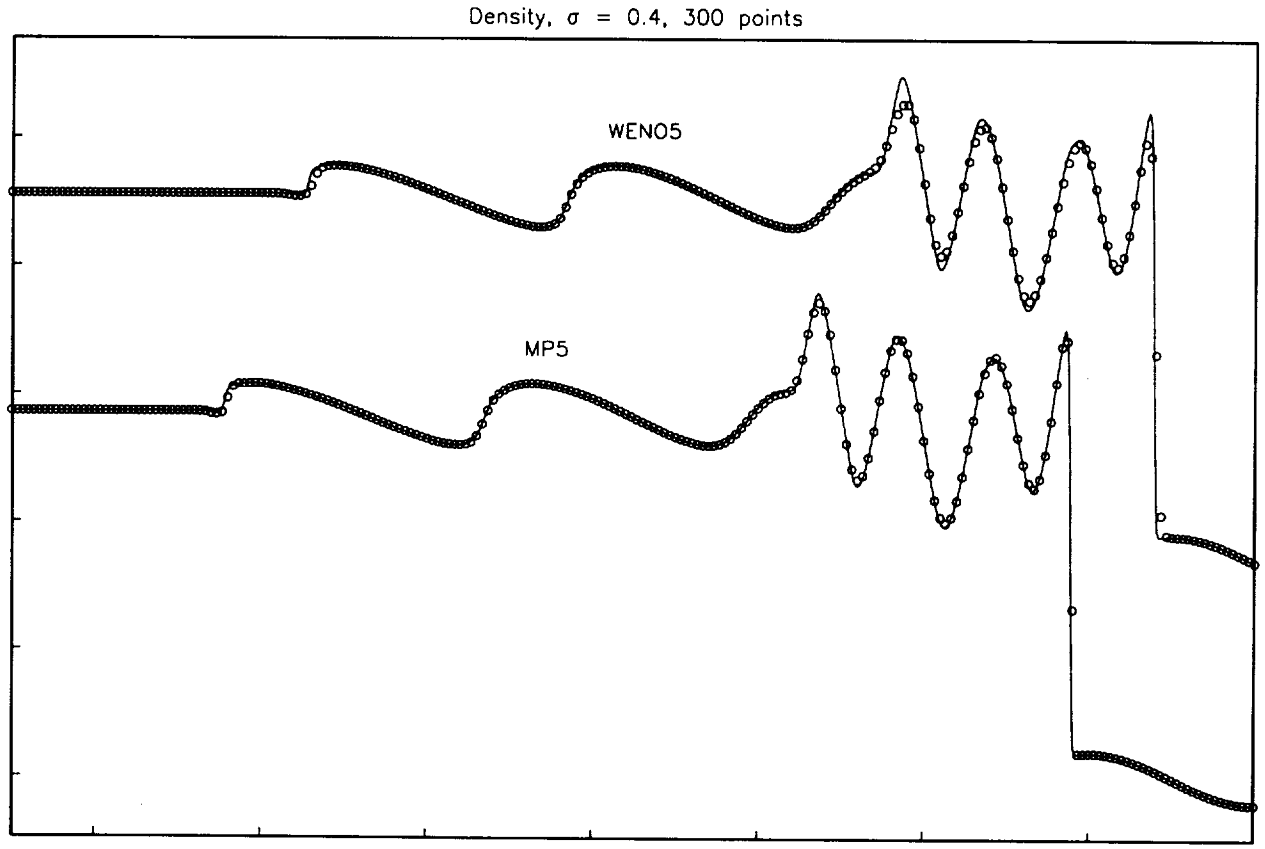

Figure 4.5: Shu's problem with MP5 and WENO5 schemes.

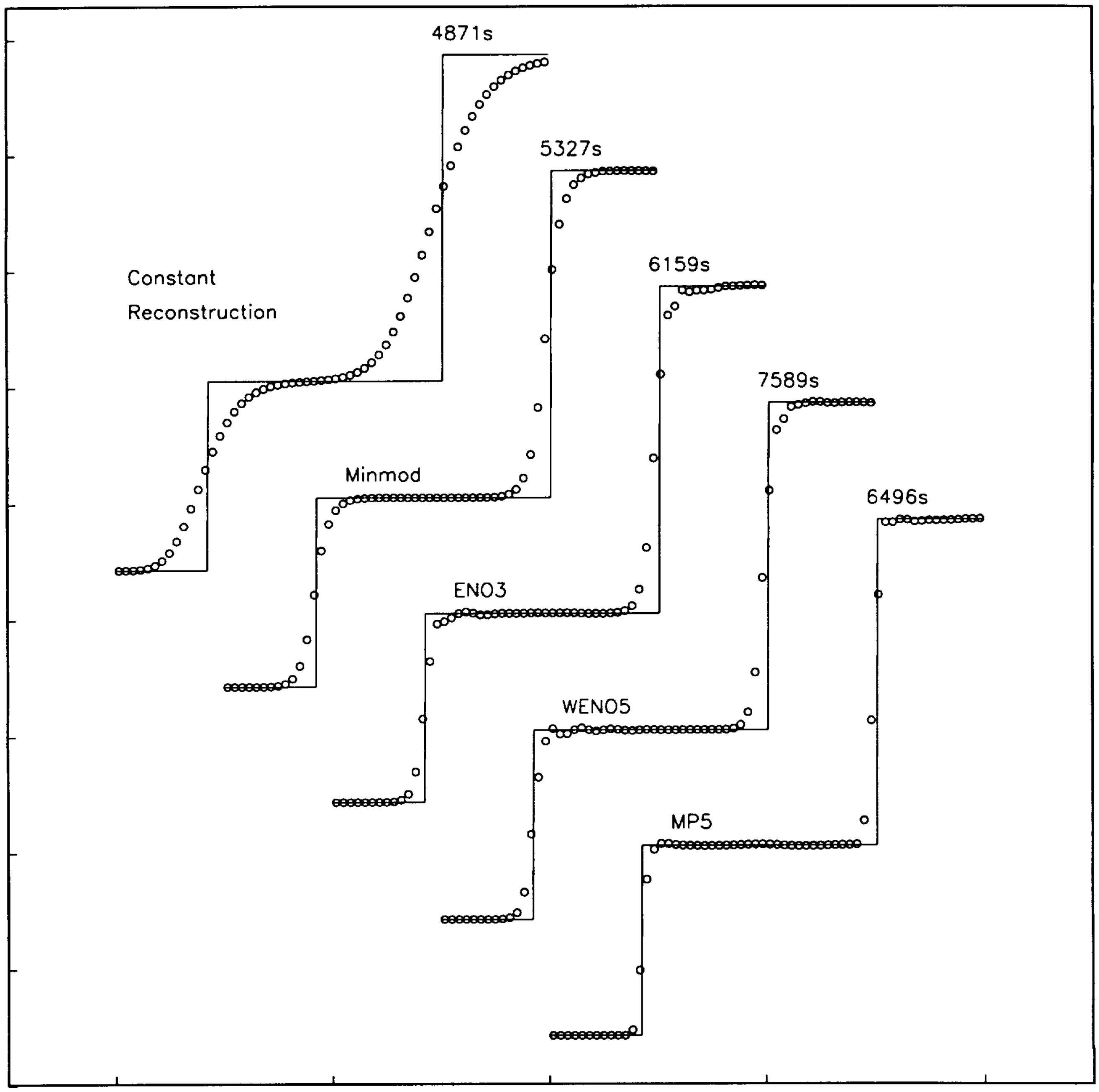

Figure 4.6: Oblique shock reflection problem with various schemes. $\sigma = 0.4$, 60 X 40 grid. CPU time in seconds is shown above each plot.

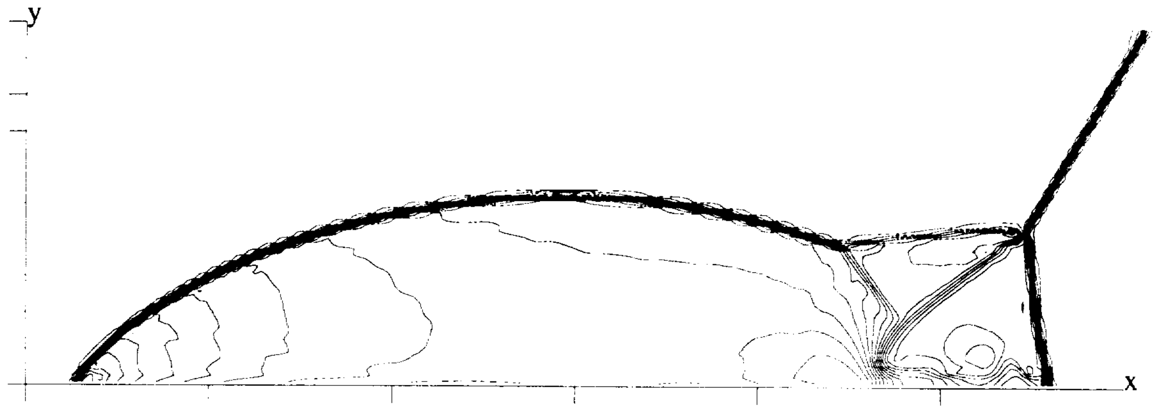

Figure 4.7(a): Double Mach reflection problem with MP5, CFL = 0.4, 240 X 60 grid. 30 density contours from 1.73 to 21. CPU time = 5553 seconds. CPU time for constant reconstruction= 4122 seconds.

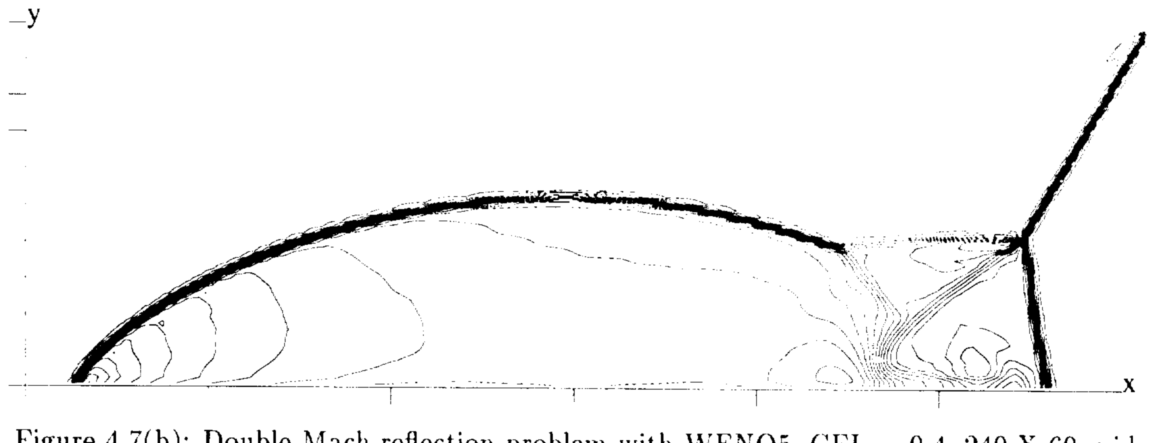

Figure 4.7(b): Double Mach reflection problem with WENO5, CFL = 0.4, 240 X 60 grid. 30 density contours from 1.73 to 21. CPU time = 6843 seconds. CPU time for constant reconstruction= 4122 seconds.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave blank*)	2. REPORT DATE January 1997	3. REPORT TYPE AND DATES COVERED Technical Memorandum

4. TITLE AND SUBTITLE Accurate Monotonicity-Preserving Schemes With Runge-Kutta Time Stepping	5. FUNDING NUMBERS WU-505-62-52
6. AUTHOR(S) A. Suresh and H.T. Huynh	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) National Aeronautics and Space Administration Lewis Research Center Cleveland, Ohio 44135-3191	8. PERFORMING ORGANIZATION REPORT NUMBER E-10532
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) National Aeronautics and Space Administration Washington, DC 20546-0001	10. SPONSORING/MONITORING AGENCY REPORT NUMBER NASA TM-107367 AIAA-97-2037

11. SUPPLEMENTARY NOTES
Prepared for the 13th CFD Conference sponsored by the American Institute of Aeronautics and Astronautics, Snowmass, Colorado, June 29—July 2, 1997. A. Suresh, NYMA, Inc., 2001 Aerospace Parkway, Brook Park, Ohio 44142 (work funded by NASA Contract NAS3-27186) and H.T. Huynh, NASA Lewis Research Center. Responsible person, H.T. Huynh, organization code 5880, (216) 433-5852.

12a. DISTRIBUTION/AVAILABILITY STATEMENT Unclassified - Unlimited Subject Categories 02 and 34 This publication is available from the NASA Center for AeroSpace Information, (301) 621-0390.	12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

A new class of high-order monotonicity-preserving schemes for the numerical solution of conservation laws is presented. The interface value in these schemes is obtained by limiting a higher-order polynominal reconstruction. The limiting is designed to preserve accuracy near extrema and to work well with Runge-Kutta time stepping. Computational efficiency is enhanced by a simple test that determines whether the limiting procedure is needed. For linear advection in one dimension, these schemes are shown to be monotonicity-preserving and uniformly high-order accurate. Numerical experiments for advection as well as the Euler equations also confirm their high accuracy, good shock resolution, and computational efficiency.

14. SUBJECT TERMS Monotonicity-preserving; Shock-capturing	15. NUMBER OF PAGES 28
	16. PRICE CODE A03

17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT