JOURNAL OF COMPUTATIONAL PHYSICS **136,** 83–99 (1997)
ARTICLE NO. CP975745

Accurate Monotonicity-Preserving Schemes with Runge–Kutta Time Stepping

A. Suresh*,† and H. T. Huynh†

*NYMA, Inc., Brook Park, Ohio 44142; and †NASA Lewis Research Center, Cleveland, Ohio 44135

Received September 24, 1996; revised March 12, 1997; accepted May 16, 1997

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. Leonard combined the limiting approach with a high-order (up to ninth-order) interface value in [12]. These limiting procedures, however, cause accuracy to degenerate to first-order near extrema. Note that PPM can be considered to be an extension of Van Leer's (piecewise linear) MUSCL scheme [23], and MUSCL in turn is an extension of Godunov's scheme [3].

The essentially nonoscillatory (ENO) schemes of Harten *et al.* [4] 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.* [14] and Jiang and Shu [10] 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—the same stencil as the third-order ENO and fifth-order WENO schemes. Compared to PPM, the five-point interface value has the disadvantage of having a slightly larger stencil, but it has the advantage of being less prone to staircasing because the leading error is dissipative. Similar to PPM, oscillations are controlled by limiting. The key differences between our limiting procedure and those in the literature are that our limiting (a) preserves both monotonicity and accuracy, (b) is designed for Runge–Kutta time stepping; (c) is economical due to a simple test that determines whether limiting is needed; and (d) is presented in a geometric framework which simplifies the concepts and facilitates the proofs. 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 introduced by Huynh [7]. Extensions to piecewise parabolic schemes were presented by Suresh [21] and Huynh [8, 9]. The present scheme incorporates these ideas within a Runge–Kutta time integration framework.

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

0021-9991/97 $25.00

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, $u_0(x)$ is assumed to be periodic or of compact support so that boundary conditions are straightforward.

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

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

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

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

At time $t^n = n\tau$, where τ is the time step, assume that we know v_j^n which approximates the cell average $\overline{u}_j(t^n)$. 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 the point value $u(x_{j+1/2}, t^n)$ is called an interface value and is denoted by $v_{j+1/2}$. Since our schemes are third-order or higher, it is essential to bear in mind that v_j represents a cell average quantity while $v_{j+1/2}$ represents a point value. The calculation of the interface value (or interface flux in the case of systems of equations) from the known cell averages is accomplished in two steps. In the first or *reconstruction* step, the values $v_{j+1/2}^L$ and $v_{j+1/2}^R$ to the left and right of $x_{j+1/2}$ are defined. 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 [18]. 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

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 modified or *limited* to obtain a *final* interface value. The key idea of our limiting procedure is that it should not alter the original interface value in smooth regions including extrema—thus, accuracy is preserved. Near a discontinuity, limiting takes effect and pulls the original interface value into a certain interval; as a result, monotonicity is preserved.

Without loss of generality, we discuss the reconstruction only for $v_{j+1/2}^L$, i.e., we assume $a > 0$. We employ the five cell averages v_{j-2}, ..., v_{j+2} (the same stencil as the third-order ENO scheme) to define $v_{j+1/2}^L$. The reason for this choice is that we need at least five points to be able to distinguish between an extremum and a discontinuity. A three-point stencil used by the popular total variation diminishing (TVD) schemes cannot make this distinction. Figure 2.1a shows the data near a minimum of a parabola, and Fig. 2.1b shows the data at a discontinuity. The three values v_{j-1}, v_j, and v_{j+1} in Fig. 2.1a are identical to the corresponding ones in Fig. 2.1b. In the rest of this paper, $v_{j+1/2}^L$ denotes the original, the final, and also any generic interface value. Its exact meaning will be clarified in the text.

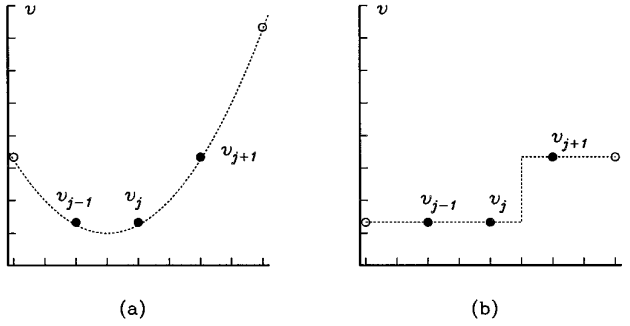

FIG. 2.1. An extrema and a discontinuity look the same over a stencil of three points. The values v_{j-1}, v_j and v_{j+1} in (a) are identical to the corresponding ones in (b).

2.1. *Original Interface Value*

The five cell average quantities v_{j-2}, ..., v_{j+2} determine a fourth degree polynomial (quartic) whose value at $x_{j+1/2}$ is

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

The above choice of the original interface value results in a spatially fifth-order accurate scheme. Other choices include a low phase error fourth-order formula [9]

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

or a fifth-order accurate implicit formula [9]

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

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

The original interface value defined by one of the formulas (2.1)–(2.3) creates oscillations near a discontinuity. To suppress these oscillations, we require the interface value to lie inside a certain interval; that is, it must satisfy a certain constraint. The final interface value is obtained by enforcing this constraint on the original interface value.

2.2. *First-Order Accurate Constraint*

The constraint in this subsection is designed specifically for Runge–Kutta time stepping. It has the drawback of being only first-order accurate near extrema. The geometric framework presented here, however, will facilitate the accuracy-preserving extension in the next subsection.

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,

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

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

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

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

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

where

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

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

We now derive constraints for the interface value so that monotonicity is preserved by (1.6). At interface $j - 1/2$, suppose the value $v^L_{j-1/2}$ lies between v_{j-1} and v_j,

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

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

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

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},

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

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 restriction

$$\sigma \leq 1/(1 + \alpha). \quad (2.10)$$

Indeed, for increasing data, (2.7) and (2.9) imply that the steepest slope $v^{UL} - v_{j-1}$ satisfies $v^{UL} - v_{j-1} \leq$

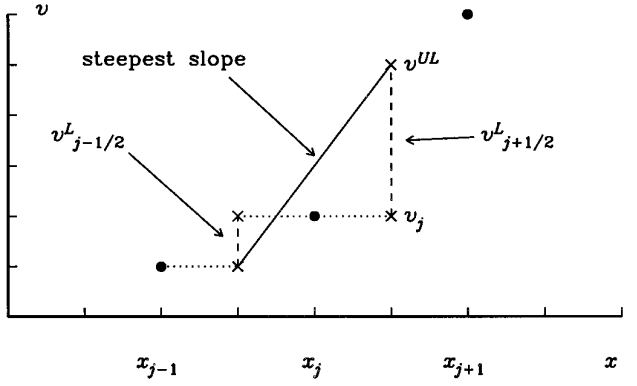

FIG. 2.2. Monotonicity-preserving constraint (2.7) and (2.9).

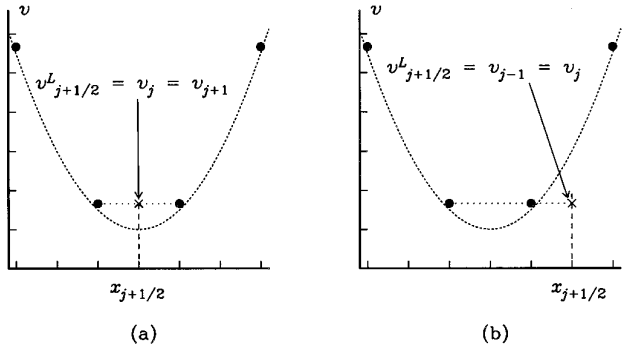

FIG. 2.4. Loss of accuracy near extrema: the original $v^L_{j+1/2}$ is exact (on the dotted line), but the final $v^L_{j+1/2}$ is identical to v_j in (a) due to (2.11) and in (b) due to (2.9).

$(\alpha + 1)(v_j - v_{j-1})$; therefore, (2.10) implies $v_{j-1} \le w_j^{(1)} \le v_j$. See Fig. 2.2.

Note that for parabolic reconstruction schemes, α is typically 2 (see [2] or [21]). For Runge–Kutta time stepping, numerical experiments indicate that $\alpha = 4$ works well, while $\alpha = 2$ may cause staircasing. With $\alpha = 4$, expression (2.10) leads to a CFL number restriction $\sigma \le 0.2$; 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} where MP stands for monotonicity-preserving. Using v_j as the pivot, v^{MP} can be expressed by the minmod function,

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

Expressions (2.9) and (2.11) therefore imply

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

The above constraint can be enforced by using the median function. Indeed, to bring a quantity y into an interval $I[a, b]$, we simply replace y by the median of y, a, and b (see Fig. 2.3). Thus, to bring the original $v^L_{j+1/2}$ into the interval $I[v_j, v^{MP}]$, we replace the original $v^L_{j+1/2}$ by the median of the three quantities: the original $v^L_{j+1/2}$, v_j, and v^{MP},

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

Note that the above technique will be used repeatedly to bring an interface value into a specific interval. Also note that the median function yields a result that depends continuously on the data.

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 CFL restriction (2.10) 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 $\{v_j^{n+1}\} = \{w_j^{(3)}\}$ are also monotone.

The drawback of (2.14) is that near an extremum, it causes accuracy to degenerate to first-order as shown in Fig. 2.4. In this figure, the data are on a parabola, and the original interface value is exact (on the dotted curves). For Fig. 2.4a, due to (2.11), the interval $I[v_j, v^{MP}]$ reduces to the point v_j; as a result, the final $v^L_{j+1/2}$ is identical to v_j. Similarly, for Fig. 2.4b due to (2.9), $I[v_j, v^{MP}]$ also reduces to v_j and the final $v^L_{j+1/2}$ is again identical to v_j.

This loss of accuracy can be avoided by a technique that senses an extremum and turns off the constraint there [13].

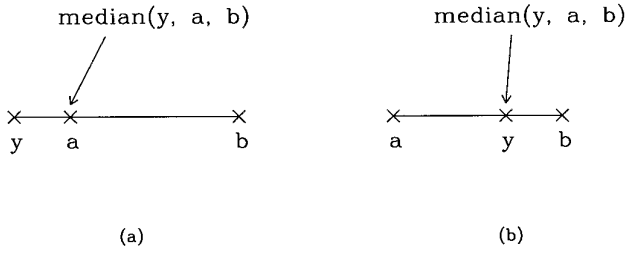

FIG. 2.3. The median function is employed to bring y into the interval $[a, b]$.

While such a technique can be effective, its drawbacks are: (a) it is sometimes difficult to distinguish between smooth and nonsmooth extrema and (b) the solution does not depend continuously on the data. Our approach below is free from these problems; in addition, it facilitates the proofs.

2.3. *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 $v^L_{j+1/2}$.

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,

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

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

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

The above constraint preserves accuracy near extrema. Indeed, consider a smooth minimum as in Fig. 2.5a. Recall that the interval $I[v_j, v_{j+1}]$ reduces to the point v_j in this case, and constraint (2.11) causes a loss of accuracy. The value v^{MD}, however, 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 $v^L_{j+1/2}$. In fact, if the second derivative is not zero and the original $v^L_{j+1/2}$ is highly accurate, then $v^{MD} - v^L_{j+1/2}$ is of order $O(h)$, i.e., constraint (2.18) provides plenty of room so that an accurate original interface value is not altered.

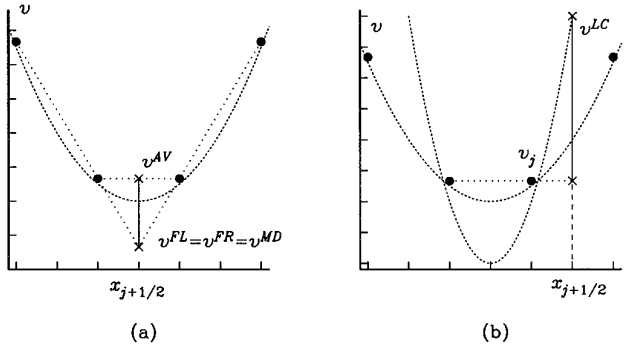

FIG. 2.5. Enlarged monotonicity intervals: (a) by (2.18) and (b) by (2.23).

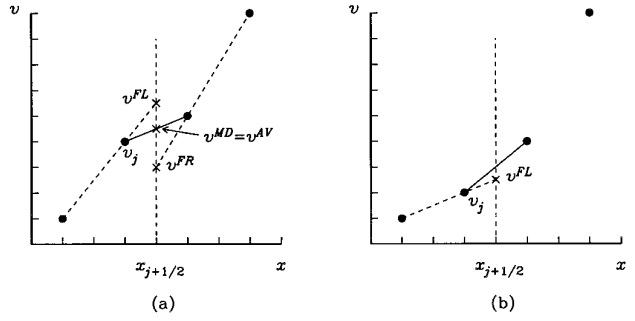

FIG. 2.6. Enlarged interval (2.18) reduces to first-order accurate interval (2.11).

If, however, 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}—as a result, the above constraint reduces to (2.11). To show this fact, set

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

Without loss of generality, assume that the data are increasing. If d_j and d_{j+1} are of opposite sign as in Fig. 2.6a, then $v^{MD} = v^{AV}$, and the above claim follows. If d_j and d_{j+1} are of the same sign, say, both positive as in Fig. 2.6b, then v^{FL} lies between v_j and v_{j+1} (if d_j and d_{j+1} are both negative, then v^{FR} lies between v_j and v_{j+1}). Consequently, v^{MD} also lies between v_j and v_{j+1}, and the claim again holds true.

The argument (2.15)–(2.18) conveys the idea. For efficient coding and for later use, set

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

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

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

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 $x_{j+1/2}$,

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

The parabola with $d = 4d^{MM}_{j-1/2}$ gives (the factor 4 will be explained momentarily)

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

Constraint (2.9) is relaxed to

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

The above constraint preserves accuracy near extrema. Indeed, consider an interface near a smooth minimum as in Fig. 2.5(b). Recall that the interval $I[v_j, v^{UL}]$ in this case reduces to the point v_j and constraint (2.9) causes a loss of accuracy. The value v^{LC}, however, 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 fact, if the second derivative is not zero and the original $v^L_{j+1/2}$ is highly accurate, then $v^{LC} - v^L_{j+1/2}$ is of order $O(h)$, i.e., constraint (2.23) provides plenty of room so that an accurate original interface value is not altered. As for the proof of monotonicity, we will need the intersection of the intervals in (2.18) and (2.23).

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

$$v^{\min} = \max[\min(v_j, v_{j+1}, v^{MD}), \min(v_j, v^{UL}, v^{LC})], \tag{2.24a}$$

$$v^{\max} = \min[\max(v_j, v_{j+1}, v^{MD}), \max(v_j, v^{UL}, v^{LC})]. \tag{2.24b}$$

The accuracy-preserving constraint takes the form

$$v^L_{j+1/2} \in [v^{\min}, v^{\max}]. \tag{2.25}$$

Finally, to bring the original $v^L_{j+1/2}$ into the interval $[v^{\min}, v^{\max}]$, we replace $v^L_{j+1/2}$ by the median of $v^L_{j+1/2}$, $v^{\min}$, and $v^{\max}$:

$$v^L_{j+1/2} \leftarrow \text{median}\,(v^L_{j+1/2}, v^{\min}, v^{\max}). \tag{2.26}$$

We turn now to the proof of monotonicity. We will show that if the five pieces of data v_{j-2}, ..., v_{j+2} are monotone, then at the interface $j + 1/2$, the interval $[v^{\min}, v^{\max}]$ reduces to the interval $I[v_j, v^{MP}]$ of the first-order accurate constraint; consequently, monotonicity is preserved. Indeed, without loss of generality, we may assume the data are increasing. Consider the following two cases determined by the sign of d_{j-1}.

In the first case, d_{j-1} is positive. If $v_{j-1} = v_j$, then since the data are increasing and d_{j-1} is positive, $v_{j-2} = v_{j-1} = v_j$, and the claim follows. If $v_{j-1} \neq v_j$, after a normalization, we may assume $v_{j-1} = 0$ and $v_j = 1$. Since $d_{j-1} \geq 0$ and the data are increasing, it follows that $-1 \leq v_{j-2} \leq 0$. The worst case scenario is therefore $v_{j-2} = v_{j-1} = 0$, and $v_j = 1$ as shown in Fig. 2.7a. We will show that $v^{LC} \leq v^{UL}$. To this end, since $\alpha \geq 2$ (here the assumption $\alpha \geq 2$ introduced after (2.8) is employed), the definition of v^{UL} given by (2.8) implies $v^{UL} \geq 3$. Since $d_{j-1} = 1$, expression (2.22) implies $v^{LC} \leq 17/6$. This is where the factor 4 above expression (2.22) is needed. (If we use the factor of 9/2, then the upper bound for v^{LC} is 3; we chose the factor 4 because it already provides plenty of room.) Thus $v^{LC} \leq v^{UL}$, and the enlarged interval $I[v_j, v^{UL}, v^{LC}]$ in (2.23) reduces to the interval $I[v_j, v^{UL}]$ in (2.9). The argument of the paragraph containing (2.19) shows that the interval $I[v_j, v_{j+1}, v^{MD}]$ also reduces to $I[v_j, v_{j+1}]$. Consequently, $[v^{\min}, v^{\max}]$ reduces to $I[v_j, v^{MP}]$.

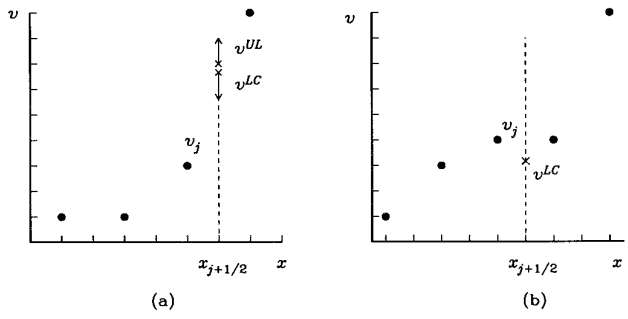

FIG. 2.7. Enlarged monotonicity interval (2.23).

In the second case, d_{j-1} is negative. Again $v^{LC} \leq v^{UL}$, but what can cause trouble is that v^{LC} can be smaller than v_j as shown in Fig. 2.7b. Here, the other interval comes to the rescue: since $I[v_j, v_{j+1}, v^{MD}]$ reduces to $I[v_j, v_{j+1}]$, the interval $[v^{\min}, v^{\max}]$ is again identical to $I[v_j, v^{MP}]$. This completes the proof of monotonicity.

The above accuracy-preserving constraints work well in most cases. In practice, however, we reduce the amount of room so that near a non-monotone discontinuity, 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

$$d^{M4}_{j+1/2} = \text{minmod}(4d_j - d_{j+1}, 4d_{j+1} - d_j, d_j, d_{j+1}). \tag{2.27}$$

To clarify the role of d^{M4}, assume that d_j and d_{j+1} are of the same sign. Then if $d_{j+1}/d_j < 1/4$ or $d_{j+1}/d_j > 4$, the above minmod of four arguments returns 0; in this case, the extended intervals reduce to the simple ones in (2.9) and (2.11). If $1/3 \leq d_{j+1}/d_j \leq 3$ then (2.27) reduces to $\text{minmod}(d_j, d_{j+1})$. In terms of limiter functions, with $r = d_{j+1}/d_j$, the right-hand side of (2.27) takes the form $d_j f(r)$, where

$$f(r) = \text{minmod}(4 - r, 4r - 1, 1, r).$$

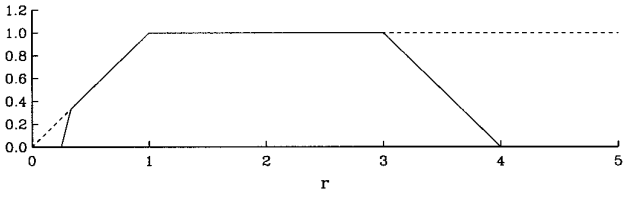

FIG. 2.8. Limiter functions: minmod $(4 - r, 4r - 1, 1, r)$ (solid line) and minmod $(1, r)$ (dashed line).

The above limiter function is plotted in Fig. 2.8 (solid line) together with the function minmod $(1, r)$ (dashed line) for comparison.

As a result of the above argument, for each interface $j + 1/2$, expressions (2.21) and (2.22) are replaced by

$$v^{MD} = v^{AV} - \tfrac{1}{2} d^{M4}_{j+1/2}, \tag{2.28}$$

$$v^{LC} = v_j + \tfrac{1}{2}(v_j - v_{j-1}) + \tfrac{4}{3} d^{M4}_{j-1/2}. \tag{2.29}$$

It can be verified that the proofs of monotonicity and accuracy still hold for constraints (2.23) and (2.18) with v^{MD} and v^{LC} defined as above.

Note that for most cells in smooth regions, the original $v^L_{j+1/2}$ satisfies the first-order accurate constraint (2.13) *a priori.* In this case, the limiting procedure (2.26) 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 (2.26) altogether. The condition that the original $v^L_{j+1/2}$ lies in the interval $I[v_j, v^{MP}]$ is equivalent to $(v^L_{j+1/2} - v_j)(v^L_{j+1/2} - v^{MP}) \le 0$. In practice, this condition is coded with a tolerance value of $\varepsilon = 10^{-10}$:

$$(v^L_{j+1/2} - v_j)(v^L_{j+1/2} - v^{MP}) \le \varepsilon. \tag{2.30}$$

We summarize the computation of the interface value below.

ALGORITHM FOR THE INTERFACE VALUE. *Suppose the cell averages $\{v_j\}$ are given, and $a \ge 0$. For each interface $j + 1/2$, calculate the original value $v^L_{j+1/2}$ by* (2.1), *and v^{MP} by* (2.12). *If* (2.30) *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} by* (2.19), *$d^{M4}_{j+1/2}$ and $d^{M4}_{j-1/2}$ by* (2.27), *v^{UL} by* (2.8), *v^{AV} by* (2.16), *v^{MD} by* (2.28), *v^{LC} by* (2.29), *and $v^{\min}, v^{\max}$ by* (2.24). *Finally, calculate $v^L_{j+1/2}$ by* (2.26), *and the interface value is then $v_{j+1/2} = v^L_{j+1/2}$.*

2.4. *Remarks*

The first-order accurate constraint (2.13) is a sufficient condition for monotone data to remain monotone under Runge–Kutta time stepping. It may be viewed as an analogue of Van Leer's constraint [22] 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 [6].

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. The corresponding limited scheme is sixth-order accurate but it is prone to staircasing [12].

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

$$\begin{aligned} v^L_{j+1/2} = (&-3v_{j-3} + 25v_{j-2} - 101v_{j-1} + 319v_j \\ &+ 214v_{j+1} - 38v_{j+2} + 4v_{j+3})/420, \end{aligned} \tag{2.31a}$$

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

$$\begin{aligned} v^L_{j+1/2} = (&4v_{j-4} - 41v_{j-3} + 199v_{j-2} - 641v_{j-1} \\ &+ 1879v_j + 1375v_{j+1} - 305v_{j+2} \\ &+ 55v_{j+3} - 5v_{j+4})/2520. \end{aligned} \tag{2.31b}$$

The same limiting can be employed for these original interface values. The resulting schemes achieve high spatial accuracy but remain third-order in time. To achieve high-order accuracy in time, one can employ the fourth- and fifth-order Runge–Kutta methods. These methods, however, require the calculation of the time-reversed operator $\tilde{L}$ as described in [18], 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 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 an ideal gas can be written as

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

where

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

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 and are given by

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

and

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

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

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

where $\overline{\mathbf{u}}_j(t)$ are the cell averages. The task reduces to calculating $\mathbf{f}_{j+1/2}$ which approximates the exact flux $\mathbf{f}(\mathbf{u}(x_{j+1/2}, t^n))$. To this end, we first reconstruct $\mathbf{u}$ on both sides of the interface $x_{j+1/2}$.

It is well known that the reconstruction of $\mathbf{u}$ is best carried out in local characteristic variables to avoid oscillations [4]. Let $\{\mathbf{v}_j\}$ be the approximations to the cell averages of $\mathbf{u}$ at time level n. The local characteristic variables for the reconstruction in each cell j are

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

Note that in the above expression, $\mathbf{L}$ is frozen at $\mathbf{v}_j$, and k varies from -2 to 2 because our reconstruction has a five-point stencil. The scalar reconstruction algorithm is now applied to each component of $\mathbf{w}$ to obtain the interface 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 calculated by

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

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 [15]. This splitting is implemented here with Huynh's entropy fix [7].

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

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

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 [7]. However, we do not pursue this approach here.

3.2 *Euler System in Two Dimensions*

An immediate extension of the above scheme to multidimensions can be accomplished in the same manner as the finite difference ENO schemes of Shu and Osher [18, 19]. 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. The computed values are the point values at the cell centers. From among the different versions of these finite difference ENO schemes, we have chosen Lax–Friedrichs (ENO-LLF). Our two-dimensional scheme is obtained by substituting the one-dimensional reconstruction summarized at the end of Section 2 for the ENO reconstruction

in ENO-LLF. Coding aspects of these ENO schemes can be found in [20].

4. NUMERICAL EXPERIMENTS

For simplicity, we present numerical results only for the scheme combining the quartic fit (2.1) and the accuracy-preserving constraint (2.26). 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 these three reconstruction procedures in FORTRAN are given in the Appendix. Note that we employ only uniform meshes.

The MP5 scheme is fifth-order in space and third-order in time. Consequently, one may wish to employ a small CFL number so that the temporal error is comparable to the spatial error. For nonsmooth data, however, a small CFL number does not improve the results. As a compromise between accuracy, monotonicity, and economy, we use the CFL number 0.4 unless otherwise stated.

All computations are carried out on a 100-MHz R4000 SGI Indigo Workstation, with $\varepsilon = 10^{-10}$ and $\alpha = 4$. In all cases, the compiler options -r8 -O3 are employed (r8 for double precision and O3 for optimization). 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 three-stage Runge–Kutta time stepping is also provided as a reference. Since the reconstruction is trivial, this computing time reflects the cost of all other steps except reconstruction. It does not reflect the true cost of the first-order scheme, which can be coded without the characteristic decomposition (3.6) or Runge–Kutta time stepping.

4.1. *One-Dimensional Advection*

EXAMPLE 1 (Smooth Initial Condition). On the domain $[-1, 1]$, we solve (1.1) with $u_0(x) = \sin^4(\pi x)$ and periodic boundary conditions. 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 I and $\sigma = 0.4$ in Table II. To get an idea of the efficiency of each scheme, the computing times on the finest grid are also provided. The results from ENO3 are less accurate and are not shown.

TABLE I

Advection of $\sin^4(\pi x)$ for One Period

Scheme	N	L_∞ error	L_∞ order	L_1 error	L_1 order	CPU time (s)
WENO5	16	2.39(−1)	—	1.07(−1)	—	
	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	16.60
MP5	16	1.17(−1)	—	8.05(−2)	—	
	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	8.64
Unlim.	16	1.17(−1)	—	8.05(−2)	—	
	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	4.51
Constant reconstruction						3.13

Note. CFL number = 0.05.

Note that the errors obtained by the unlimited scheme and those by the MP5 scheme are basically identical. This confirms that the limiting procedure leaves the quartic fit essentially unchanged at smooth extrema. At CFL number 0.05, the MP5 scheme approaches the theoretical order of accuracy of five as can be seen in Table I. In both cases, the MP5 scheme compares favorably with the WENO5 scheme in both accuracy and efficiency.

TABLE II

Advection of $\sin^4(\pi x)$ for One Period

Scheme	N	L_∞ error	L_∞ order	L_1 error	L_1 order	CPU time (s)
WENO5	16	2.39(−1)	—	1.07(−1)	—	
	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	2.35
MP5	16	1.21(−1)	—	8.01(−2)	—	
	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	1.36
Unlim.	16	1.21(−1)	—	8.01(−2)	—	
	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	0.83
Constant reconstruction						0.66

Note. CFL number = 0.4.

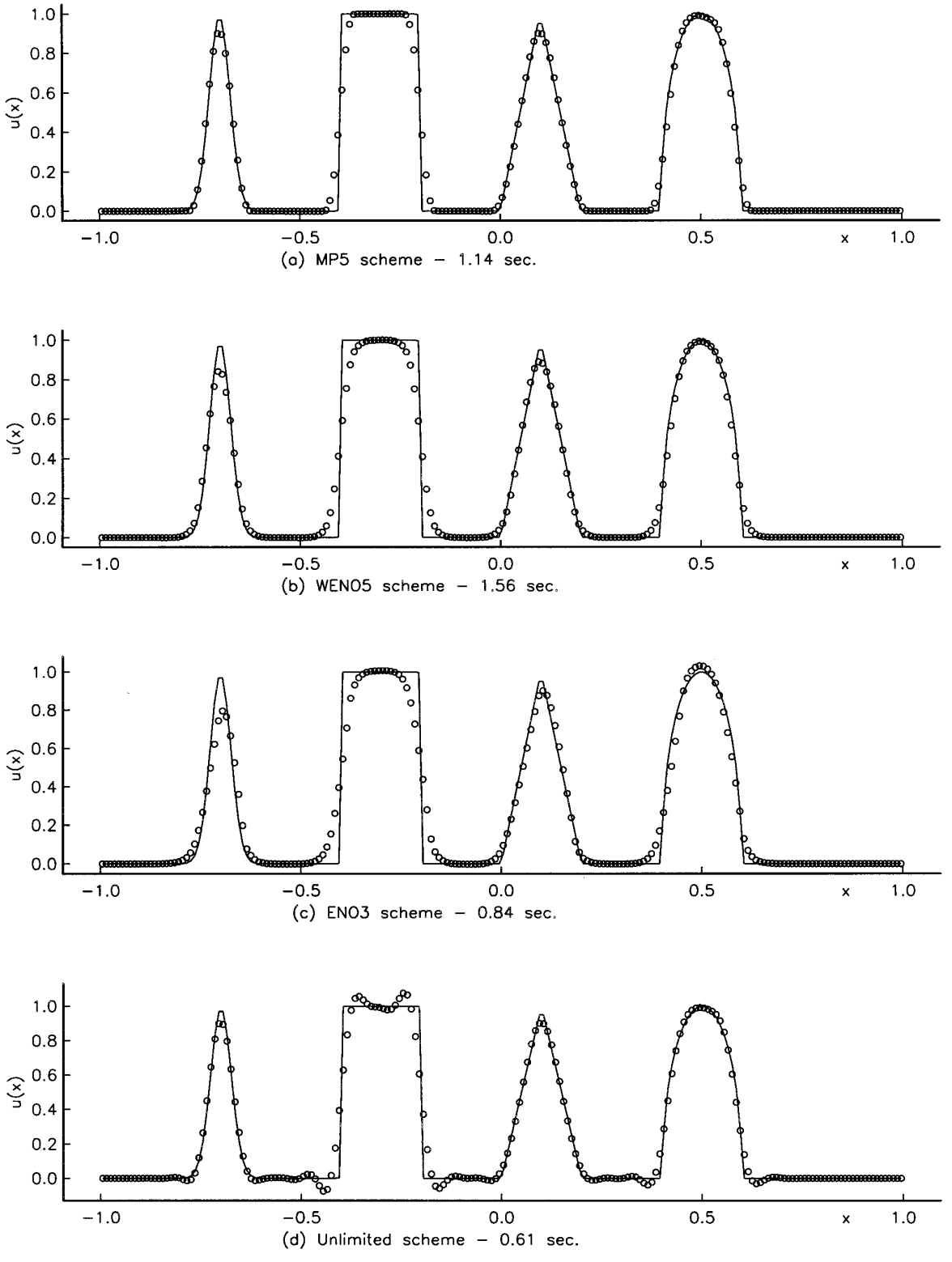

FIG. 4.1. Advection over one period by (a) MP5, (b) WENO5, (c) ENO3, and (d) Unlimited scheme. CFL number = 0.4, 200 points, CPU time for constant reconstruction = 0.63 sec.

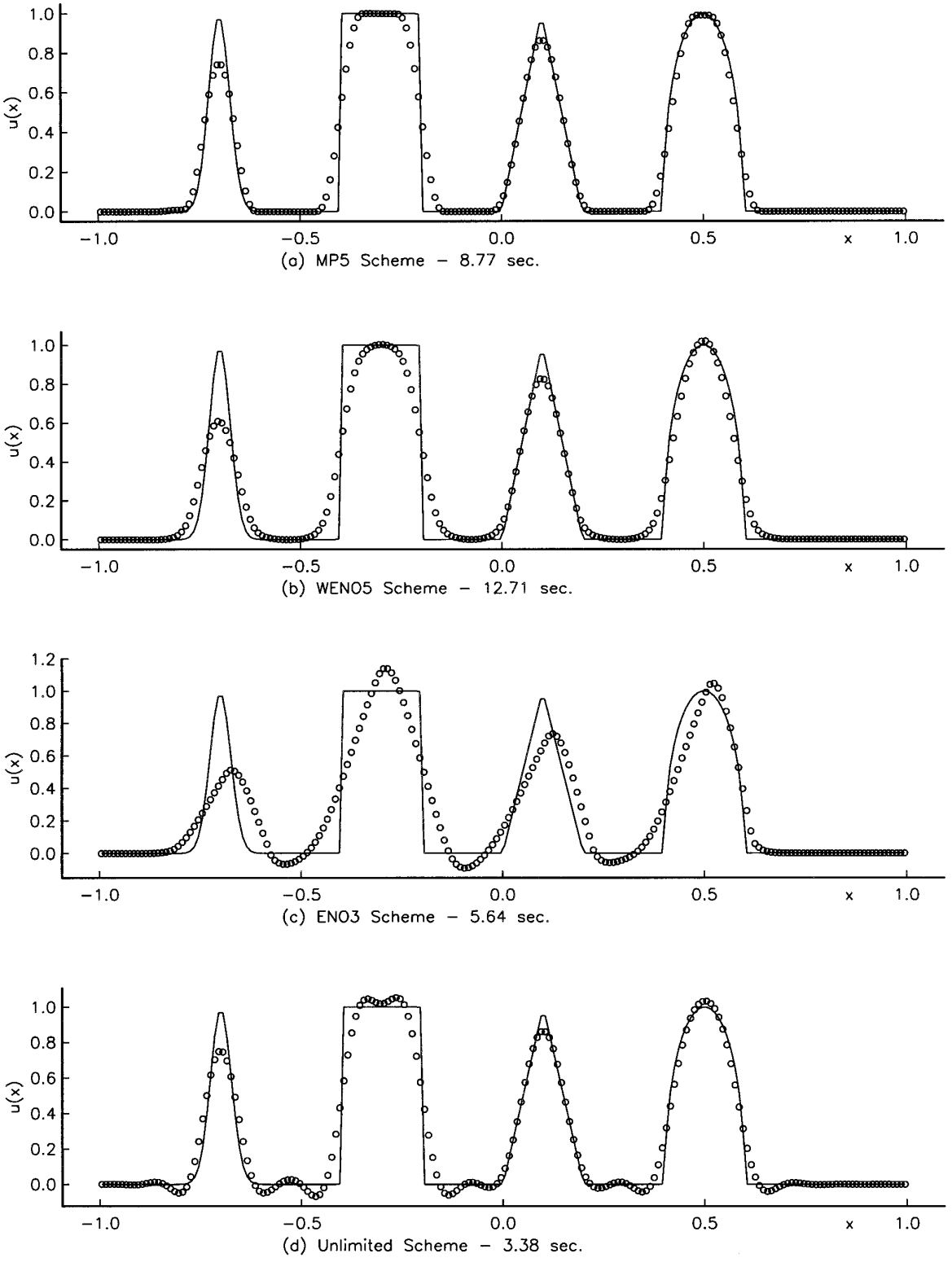

FIG. 4.2. Advection over 10 periods by (a) MP5, (b) WENO5, (c) ENO3, and (d) Unlimited scheme. CFL number = 0.4, 200 points, CPU time for constant reconstruction = 2.69 sec.

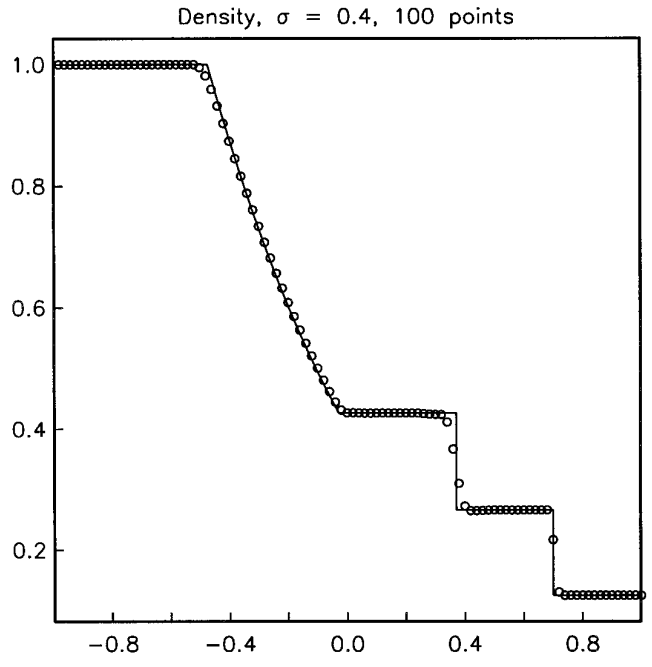

FIG. 4.3. Sod's problem with the MP5 scheme.

EXAMPLE 2 (Initial Condition with Discontinuities). Next, on the interval $[-1, 1]$, 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 [5, 7, 25]. 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 limit our study to the base schemes only.

4.2. *Euler System in One Dimension*

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

EXAMPLE 1 (Sod's Problem [16]). The initial conditions, t_f, and N are

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

Since this problem starts from a singularity, smaller time steps are used initially as described in [7]. 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.

EXAMPLE 2 (Lax's Problem [11]). The initial conditions, final time, and total number of mesh points are

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

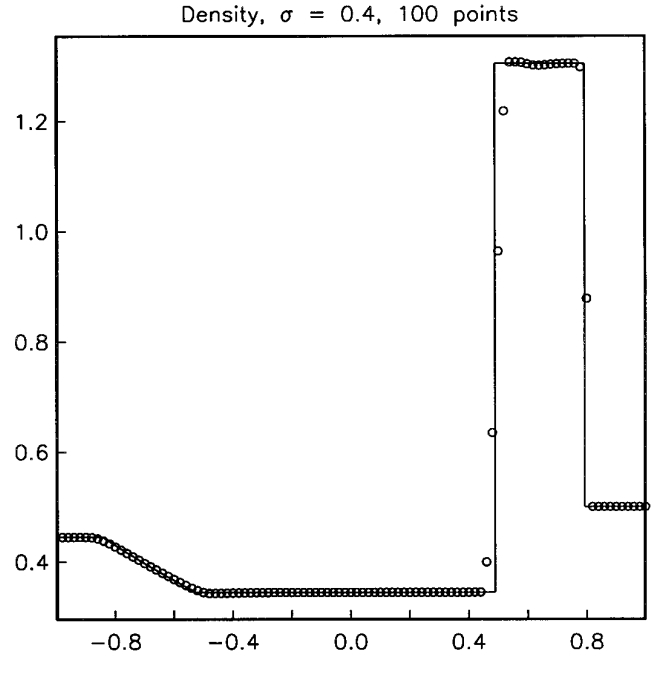

FIG. 4.4. Lax's problem with the MP5 scheme.

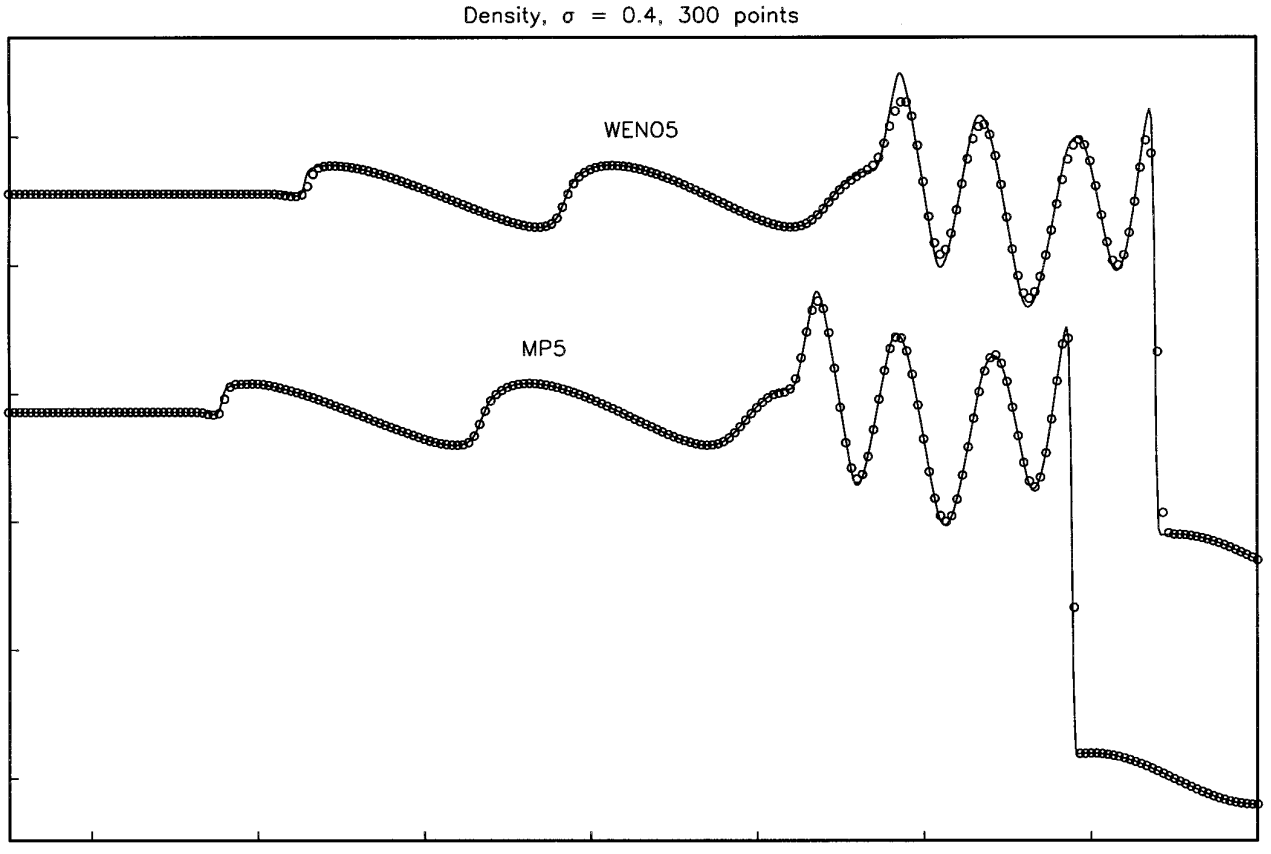

FIG. 4.5. Shu's problem with MP5 and WENO5 schemes.

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

EXAMPLE 3 (Shu's Problem [17]). 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

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

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.

4.3. *Euler System in Two Dimensions*

We present results for two well-known problems.

EXAMPLE 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 here 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 two orders of magnitude for MP5, WENO5, and

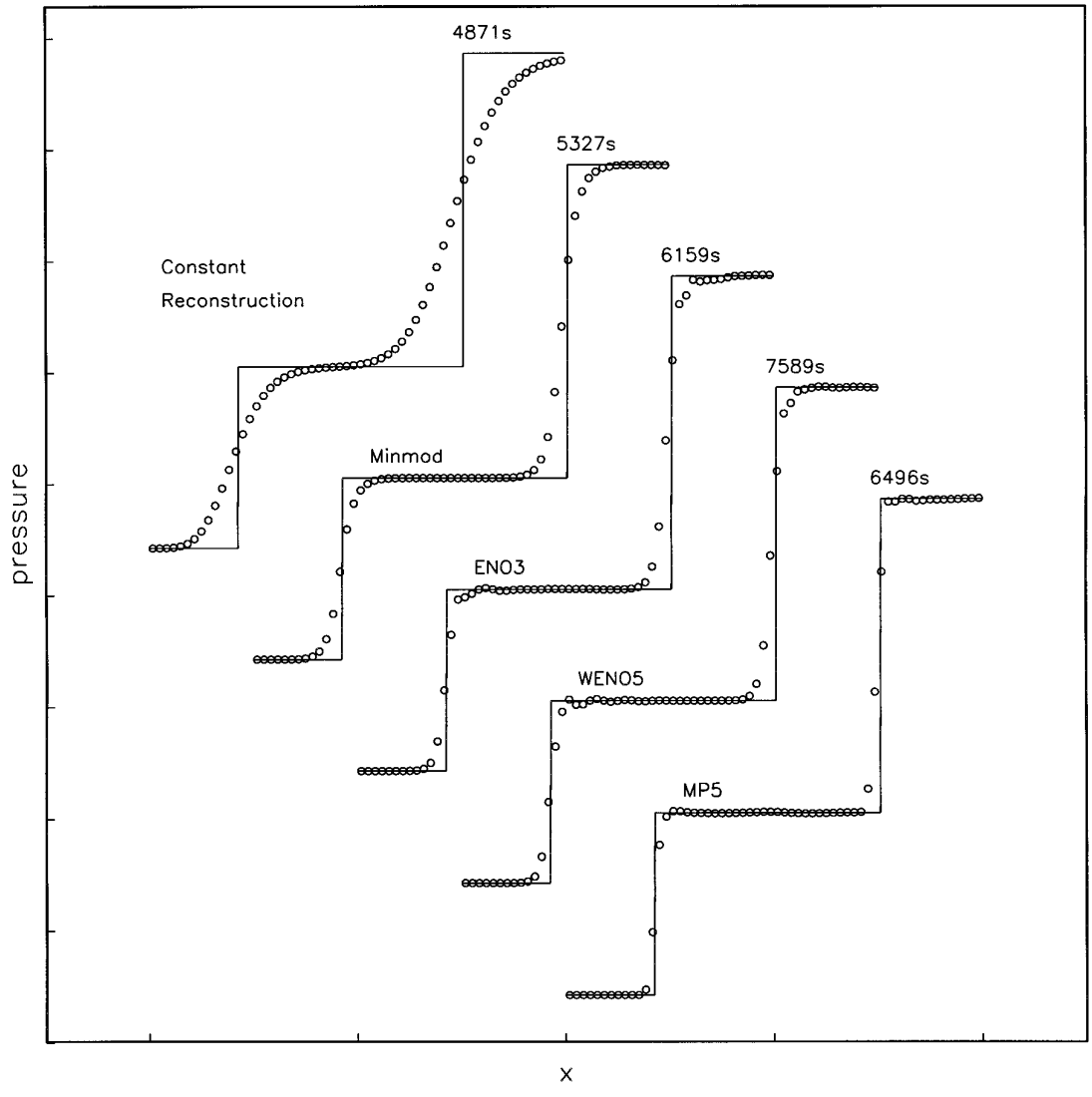

FIG. 4.6. Oblique shock reflection problem with various schemes; $\sigma = 0.4$, 60×40 grid. CPU time in seconds is shown above each plot.

ENO3, while for the minmod and first-order upwind schemes the residual drops to machine zero. (Note that the minmod scheme is defined by $\mathbf{w}_{1/2}^{L} = \mathbf{w}_0 + \frac{1}{2}$ 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. 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.

EXAMPLE 2 (Double Mach Reflection [24]). 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 60 degrees with the x-axis. To the right of the shock is undisturbed fluid

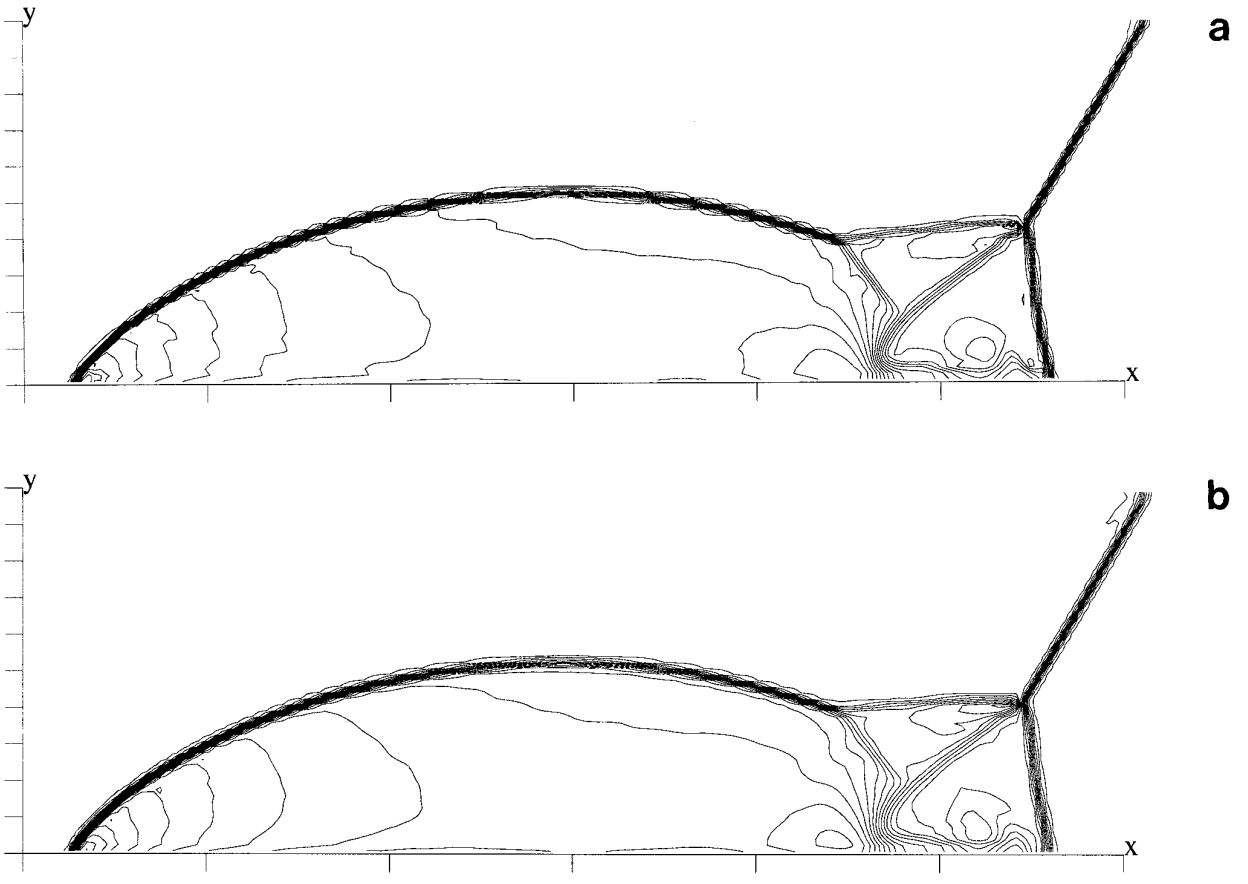

FIG. 4.7. Double Mach reflection problem by (a) MP5 scheme, CPU time = 5553 s and (b) WENO5 scheme, CPU time = 6843 s. (CFL number = 0.4, 240 × 60 grid, 30 density contours from 1.73 to 21, CPU time for constant reconstruction = 4122 s.)

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 [24].

The boundary conditions here are: at the bottom, from (0, 0) to (1/6, 0), linear extrapolation while 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. These schemes combine an accurate interface formula with a monotonicity-preserving constraint. The key differences between our constraint and those in the literature are that our constraint (a) preserves both monotonicity and accuracy; (b) is designed for Runge–Kutta time stepping; (c) is economical due to a simple test that determines whether limiting is needed; and (d) is presented in a geometric framework which simplifies the concepts and facilitates the proofs. For advection, if the data are monotone, then the solution was shown to be monotone under a time step restriction. Numerical experiments for advection and the Euler equations in one and two dimensions 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

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.
      DMM(X,Y) = 0.5*(SIGN(1.,X) + SIGN(1.,Y))*
     &       MIN(ABS(X),ABS(Y))
      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
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 = 1.083333333333
      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
            ID = 3
            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
```

ACKNOWLEDGMENT

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

REFERENCES

1. P. Colella, Multidimensional upwind methods for hyperbolic conservation laws, *J. Comput. Phys.* **87,** 171 (1990).
2. P. Colella and P. Woodward, The piecewise parabolic method for gas-dynamical simulations, *J. Comput. Phys.* **54,** 174 (1984).
3. S. K. Godunov, A finite difference method for the numerical computation of discontinuous solutions of the equations of fluid dynamics, *Mat. Sb.* **47,** 357 (1959).
4. A. Harten, B. Engquist, S. Osher, and S. R. Chakravarthy, Uniformly high-order accurate essentially nonoscillatory schemes, III, *J. Comput. Phys.* **71,** 231 (1987).
5. A. Harten, ENO Schemes with subcell resolution, *J. Comput. Phys.* **83,** 148 (1989).
6. H. T. Huynh, Accurate monotone cubic interpolation, *SIAM J. Numer. Anal.* **30,** 57 (1993).
7. H. T. Huynh, Accurate upwind methods for the Euler equations, *SIAM J. Numer. Anal.* **32,** 1565 (1995).
8. H. T. Huynh, A piecewise-parabolic dual mesh method for the Euler equations, AIAA 95-1739, in *AIAA 12th Computational Fluid Dynamics Conference, San Diego, CA, 1995.*
9. H. T. Huynh, Schemes and constraints for advection, in *International Conference on Numerical Methods in Fluid Dynamics, Monterey, CA, 1996.*
10. G-S. Jiang and C-W. Shu, Efficient implementation of weighted ENO schemes, *J. Comput. Phys.* **126,** 202 (1996).
11. P. D. Lax, Weak solutions of nonlinear hyperbolic equations and their numerical computation, *Commun. Pure Appl. Math.* **7,** 159 (1954).
12. B. P. Leonard, The ULTIMATE conservative difference scheme applied to unsteady one-dimensional advection, *Comput. Meth. Appl. Mech. Eng.* **88,** 17 (1991).
13. B. P. Leonard, A. P. Lock, and M. K. Macvean, The NIRVANA scheme applied to one-dimensional advection, *Int. J. Num. Meth. Heat Fluid Flow* **5,** 341 (1995).
14. X. Liu, S. Osher, and T. Chan, Weighted essentially non-oscillatory schemes, *J. Comput. Phys.* **115,** 200 (1994).
15. P. L. Roe, Approximate Riemann solvers, parameter vectors, and difference schemes, *J. Comput. Phys.* **43,** 357 (1981).
16. G. A. Sod, A survey of several finite difference methods for systems of non-linear hyperbolic conservation laws, *J. Comput. Phys.* **27,** 1 (1978).
17. C.-W. Shu, Numerical experiments on the accuracy of ENO and modified ENO schemes, *J. Sci. Comput.* **5,** 127 (1990).
18. C.-W. Shu and S. Osher, Efficient implementation of essentially non-oscillatory schemes, *J. Comput. Phys.* **77,** 439 (1988).
19. C.-W. Shu and S. Osher, Efficient implementation of essentially non-oscillatory schemes, II, *J. Comput. Phys.* **83,** 32 (1989).
20. C.-W. Shu, G. Erlebacher, T. A. Zang, D. Whitaker, and S. Osher, High-order ENO schemes applied to two and three-dimensional compressible flow, *ICASE Report 91-38,* Hampton, Virginia, 1991.
21. A. Suresh, Centered nonoscillatory schemes of third order, AIAA 95-1755, in *AIAA 12th Computational Fluid Dynamics Conference, San Diego, CA, 1995.*

22. B. van Leer, Towards the ultimate conservative difference scheme. II. Monotonicity and conservation combined in a second-order scheme. *J. Comput. Phys.* **14,** 361 (1974).
23. B. van Leer, Towards the ultimate conservative difference scheme. V. A. second-order sequel to Godunov's method, *J. Comput. Phys.* **32,** 101 (1979).
24. P. Woodward and P. Colella, The numerical simulation of two-dimensional fluid flow with strong shocks, *J. Comput. Phys.* **54,** 115 (1984).
25. H. Yang, An artificial compression method for ENO schemes: The slope modification method, *J. Comput. Phys.* **89,** 125 (1990).