JOURNAL OF COMPUTATIONAL PHYSICS 101, 314–329 (1992)

High Resolution Staggered Mesh Approach for Nonlinear Hyperbolic Systems of Conservation Laws

RICHARD SANDERS*

Department of Mathematics, University of Houston, Houston, Texas 77004

AND

ALAN WEISER

Exxon Production Research Company, Houston, Texas 77252-2189

Received May 8, 1989; revised June 28, 1991

A numerical technique is presented to approximate weak solutions of hyperbolic systems of conservation laws in one and two space dimensions. When strong shocks are present in the exact solution other techniques rely on approximate Riemann problem solvers to cope with this difficulty. The high-order approach here does not use approximate Riemann solvers as a building block and is therefore considerably easier to implement compared to the usual methods. Moreover, the approach here yields sharp nonoscillatory shocks, sharp corners at the base of rarefaction waves and has high order accuracy in regions where the solution is smooth. For the scalar one-dimensional problem the theoretical results confirm the reliability of this approach.

1. INTRODUCTION

In this paper, we present a shock-capturing finite difference method for systems of hyperbolic conservation laws

$$\frac{\partial}{\partial t}u+\frac{\partial}{\partial x}f(u)+\frac{\partial}{\partial y}g(u)=0$$

$$u(x, y, 0)=u_0(x, y),$$

which is competitive in performance with the best present-day methods while avoiding the need to solve Riemann problems either analytically or approximately. Moreover, the resulting computer program can be extremely short even while solving complicated multidimensional hyperbolic systems.

It is my sad duty to report the tragic death of my good friend and colleague Alan Weiser. He passed away on September 25, 1991, at the age of thirty six years. He was one of the kindest and brightest persons I have ever known. All who knew him are sure to agree. He will be missed. Richard Sanders

* Research supported in part by NSF Grant DMS 87-03383 and in part by EPR contract PR-10434.

The method we present here can be thought of as a conservative version of the modified method of characteristics [5]. A staggered spatial mesh is employed so that complicated nonlinear waves coming from nonconstant data separated by jump discontinuities can be neglected during the calculation of numerical flux functions. Other methods, using nonstaggered meshes, require a great deal of overhead to both develop the computer program and to compute the numerical fluxes during execution. One possible drawback of using a staggered mesh is the amount of numerical diffusion encountered when capturing slowly moving waves. To eliminate this possible drawback we consider a new type of reconstruction algorithm. Specifically, we reconstruct point values and cell averages into piecewise linear functions with hinges. By introducing a hinge in the reconstruction it is possible to resolve fixed discontinuities for all time with only a two-zone transition.

Some advantage of the method discussed here compared to conventional techniques include:

(1) No Riemann problems are solved. Analytic solutions to Riemann problems are available for special classes of problems, such as scalar equations [10] or gas dynamics [6], but they are not readily found, in general. Furthermore, plausible approximate Riemann solvers may not be adequately robust. For example, Roe's scheme leaves certain expansion shocks fixed [8].

(2) The method is easy to program and to vectorize.

(3) The method has a three-point stencil. This compares to a five-point stencil for second-order Godunov methods [1] and the PPM method [4] and a nine-point stencil for the fourth-order ENO method [7].

Some disadvantages of the method discussed here compared to conventional methods are:

0021-9991/92 $5.00

(1) The method requires half the usual CFL time step stability limit.

(2) The method uses a nonstandard staggered grid. This makes it more difficult to incorporate into existing finite difference codes.

(3) For two-dimensional problems, the method requires twice the usual number of one-dimensional sweeps per time step. However, this additional cost is offset by the cheaper cost per time step.

2

2.1. *The Evolution Algorithm*

We begin the development by discussing the staggered mesh time evolution scheme for the one-dimensional Cauchy problem

$$\begin{aligned} &\frac{\partial}{\partial t}u+\frac{\partial}{\partial x}f(u)=0,\\ &u(x,0)=u_0(x). \end{aligned} \tag{2.1}$$

Given initial data $u_0(x)$, we wish to construct an approximation $u^1(x)\approx u(x, \Delta t)$, using both point value information about $u(x, \Delta t)$ coming from the characteristics of (2.1) and cell average information coming from the divergence theorem applied to (2.1). Consider a set of grid points $\cdots < x_{j-1} < x_j < x_{j+1} \cdots$ and two partitions of the real line

$$\mathbf{R}=\bigcup_{j\in\mathscr{J}^0}\bar{I}_j, \tag{2.2a}$$

$$\mathbf{R}=\bigcup_{j\in\mathscr{J}^1}\bar{I}_j, \tag{2.2b}$$

where $\mathscr{J}^0$ is the set of even integers, $\mathscr{J}^1$ is the set of odd integers, and where I_j denotes the interval $[x_{j-1}, x_{j+1})$. Note that (2.2b) is a staggering of (2.2a). For ease of exposition we assume that each interval has equal length $\Delta x = x_{j+1} - x_{j-1}$. Functions of bounded variation are mapped onto intervals I_j by what we refer to as a local reconstruction operator. We denote the jth local reconstruction operator applied to a function $u(x)$ by $\mathscr{R}_j(u)(x)$. Throughout, $\mathscr{R}_j(u)$ will be required to satisfy to conservation property

$$\int_{I_j}\mathscr{R}_j(u)(x)\,dx=\int_{I_j}u(x)\,dx.$$

In this paper, $\mathscr{R}_j(u)(x)$ will always be completely determined by the values of $u(x_{j-1})$, $u(x_{j+1})$ and the mean value of $u(x)$ over the interval I_j. $\mathscr{R}_j(u)(x)$ should be thought of as a continuous, finite degree of freedom, cell average preserving approximation to $u(x)$ valid on the interval I_j. Particular choices for local reconstructions are discussed in Section 2.2. On partition (2.2a) the global reconstruction of a function $u(x)$ is given by

$$\mathscr{R}^0(u)(x)=\sum_{j\in\mathscr{J}^0}\mathscr{R}_j(u)(x)\,\chi_{I_j}(x),$$

and on partition (2.2b) the global reconstruction is given by

$$\mathscr{R}^1(u)(x)=\sum_{j\in\mathscr{J}^1}\mathscr{R}_j(u)(x)\,\chi_{I_j}(x),$$

where $\chi_{I_j}(x)$ is the characteristic function of the interval I_j.

We discretize time into strips with endpoints t^n, where $t^0=0$ and $t^{n+1}=t^n+\Delta t^n$. At time level t^0 the initial data are reconstructed on partition (2.2a) by $\mathscr{R}^0$ and we define $u^0(x)$ by

$$u^0(x)=\mathscr{R}^0(u_0)(x). \tag{2.3a}$$

Denote the exact solution of (2.1) with initial data $u(x,0)=v(x)$ by the solution operator $\mathscr{S}(t, v)(x)$. We wish to determine the reconstruction of $\mathscr{S}(\Delta t^0, u^0)(x)$ on the staggered partition (2.2b). That is, we wish to determine $u^1(x)$, where

$$u^1(x)=\mathscr{R}^1(\mathscr{S}(\Delta t^0, u^0))(x). \tag{2.3b}$$

On the staggered partition (2.2b), exact interval averages and interval endpoint values of $\mathscr{S}(\Delta t^0, u^0)(x)$ can easily be calculated when (2.1) is a single nonlinear equation or when (2.1) is a linear hyperbolic system. To initially motivate matters, suppose that u is scalar and take $f(u)=au$ for some constant a. Define

$$\bar{u}_j^1=\frac{1}{\Delta x}\int_{I_j}\mathscr{S}(\Delta t^0, u^0)(x)\,dx.$$

Using the divergence theorem applied to (2.1), we readily find that

$$\begin{aligned} \bar{u}_j^1=&\frac{1}{\Delta x}\int_{I_j}u^0(x)\,dx\\ &-\frac{1}{\Delta x}\left(\int_{s_{j+1}}^{x_{j+1}}u^0(x)\,dx-\int_{s_{j-1}}^{x_{j-1}}u^0(x)\,dx\right), \end{aligned} \tag{2.4}$$

where $s_{j\pm1}+a\,\Delta t^0=x_{j\pm1}$. Furthermore, since $\mathscr{S}(t, u^0)(x)$ is constant along straight lines $x-at=\text{const}$, we have that

$$\mathscr{S}(\Delta t^0, u^0)(x_{j\pm1})=u^0(s_{j\pm1}). \tag{2.5}$$

We then can determine $u^1(x)=\mathscr{R}^1(\mathscr{S}(\Delta t^0, u^0))(x)$ from

the values computed in (2.4) and (2.5). Continuing this staggered grid recipe for $n \geq 1$ allows one to compute $u^n(x) = \mathscr{R}^n(\mathscr{S}(\Delta t^{n-1}, u^{n-1}))(x) \approx u(x, t^n)$, where the staggering is reflected by the fact that

$$\mathscr{J}^n = \mathscr{J}^{n-2},$$
$$\mathscr{R}^n(u) = \mathscr{R}^{n-2}(u).$$

A natural question to ask at this point is: Why jump back and forth between grid zero and grid one? Of course, for a linear problem there is little point for this. However, for a nonlinear problem there is an important advantage to this approach. The reconstruction algorithm $\mathscr{R}^n(u)$ does not always yield a continuous approximation at cell interface points x_j for $j \notin \mathscr{J}^n$. To resolve the resulting nonlinear waves coming from a jump discontinuity when the states to the left and right of the jump are not constant is a problem we wish to avoid. Suppose there is a jump at the space–time coordinate (x_j, t^n). Consider the cone given by the coordinates: (x_j, t^n), (x_{j-1}, t^{n+1}), (x_{j+1}, t^{n+1}). For $t^{n+1} - t^n$ sufficiently small, all waves coming from the jump discontinuity are trapped within the cone. Therefore, it is possible to compute the correct fluxes in (2.4) without resolving the complicated wave structure contained within the cone. When the reconstruction is piecewise constant this staggered approach is often referred to as the Lax–Friedrichs method.

Suppose now we allow $f(u)$ to be nonlinear; however, we still assume it is scalar. First, for each $n \geq 0$, we require the CFL restriction

$$\sigma^n \max_{x \in \mathbf{R}} |f'(u^n(x))| < \tfrac{1}{2}, \tag{2.6a}$$

where $\sigma^n = \Delta t^n / \Delta x$; note that this is twice as restrictive as the usual CFL restriction. Second, we require an additional restriction concerning the magnitudes of the derivatives of each *local* reconstruction. Specifically, we require that

$$\Delta t^n \max_{x \in I_j} \left| \frac{d}{dx} f'(u^n(x)) \right| < 1, \tag{2.6b}$$

for each $j \in \mathscr{J}^n$. In practice (2.6a) controls the time step size Δt; then (2.6b) is enforced by limiting the slope of the reconstruction $\mathscr{R}_j(u^n)$. One possible way to enforce (2.6b) in those instances when it is violated is to form an appropriate post-convex combination of $\mathscr{R}_j(u)$ with $\bar{u}_j$. With (2.6a) and (2.6b) we have the following lemma.

LEMMA 2.1. *Assume* (2.6) *is satisfied and* $f'(u)$ *is a Lipschitz continuous function. For each* $j \in \mathscr{J}^n$, *there is a unique backward characteristic of* (2.1) *from point* x_j *at time level* $n+1$ *back to a point* $s_j \in I_j$ *at time level* n.

Proof. The result follows by considering the characteristic equation for (2.1)

$$x(u; s) = f'(u(s))\, \Delta t^n + s. \tag{2.7}$$

$x(u^n; s)$ is a continuous function and (2.6b) implies that it is strictly increasing in s. (2.6a) implies that $x(\cdot; s)$ is greater than x_j at $s = x_{j+1}$ and less than x_j at $s = x_{j-1}$. Continuity now makes the result obvious. ∎

Suppose that $u^n(x)$ is known. We wish to determine

$$u^{n+1}(x) = \mathscr{R}^{n+1}(\mathscr{S}(\Delta t^n, u^n))(x),$$

where, as before, $\mathscr{S}(t, v)(x)$ is the solution of (2.1) with initial data $u(x, 0) = v(x)$. To accomplish this, we do not need to know $\mathscr{S}(\Delta t^n, u^n)(x)$ at all points; we only need its cell averages and its values at points x_j, $j \in \mathscr{J}^n$. First, for each $j \in \mathscr{J}^n$ solve the nonlinear characteristic equation

$$x(u^n; s_j) = x_j, \tag{2.8}$$

for the unknown s_j and evaluate $u^n(s_j)$. Since $\mathscr{S}(t, u^n)(x)$ is constant along characteristics in regions of continuity (this is a reason for staggering), we again have

$$\mathscr{S}(\Delta t^n, u^n)(x_j) = u^n(s_j). \tag{2.9}$$

For each $j \in \mathscr{J}^{n+1}$ we wish to compute the new cell average

$$\bar{u}_j^{n+1} = \frac{1}{\Delta x} \int_{I_j} \mathscr{S}(\Delta t^n, u^n)(x)\, dx.$$

As above, the cell average if found by applying the divergence theorem to (2.1) on the trapezoid with corners at $(x_{j\pm1}, \Delta t^n)$, $(s_{j\pm1}, 0)$, $j \in \mathscr{J}^{n+1}$; giving

$$\begin{aligned} \bar{u}_j^{n+1} = {} & \frac{1}{\Delta x} \int_{I_j} u^n(x)\, dx \\ & - \frac{1}{\Delta x} \left(\int_0^{\Delta t^n} f(\mathscr{S}(t, u^n)(x_{j+1}))\, dt \right. \\ & \left. - \int_0^{\Delta t^n} f(\mathscr{S}(t, u^n)(x_{j-1}))\, dt \right), \end{aligned} \tag{2.10a}$$

where

$$\begin{aligned} & \int_0^{\Delta t^n} f(\mathscr{S}(t, u^n)(x_{j\pm1}))\, dt \\ & = \int_{s_{j\pm1}}^{x_{j\pm1}} u^n(x)\, dx \\ & \quad + \Delta t^n (f(u^n(s_{j\pm1})) \\ & \quad - f'(u^n(s_{j\pm1})) \cdot u^n(s_{j\pm1})). \end{aligned} \tag{2.10b}$$

We summarize the procedure for nonlinear scalar equations: Given $u^n(x)$:

(1) For each $j \in \mathcal{J}^n$ solve the scalar nonlinear equation

$$f'(u^n(s))\,\Delta t^n + s = x_j,$$

for the unknown s_j and evaluate $\mathcal{S}(\Delta t^n, u^n)(x_j) = u^n(s_j)$.

(2) For each $j \in \mathcal{J}^{n+1}$ compute the exact cell average of $\mathcal{S}(\Delta t^n, u^n)$, using formula (2.10).

(3) Set $u^{n+1}(x) = \mathcal{R}^{n+1}(\mathcal{S}(\Delta t^n, u^n))(x)$, using the information obtained in (1) and (2) above.

Repeat (1)–(3) to advance the approximation to successive time levels.

We now demonstrate some desirable properties of the procedure presented above.

LEMMA 2.2. *Given* (2.6), *we have for each* $j \in \mathcal{J}^{n+1}$

$$\min_{x \in [s_{j-1}, s_{j+1}]} u^n(x) \leqslant \bar{u}_j^{n+1} \leqslant \max_{x \in [s_{j-1}, s_{j+1}]} u^n(x).$$

Proof. (Sketch). Consider the quantity

$$\frac{1}{\Delta x} \int_{s_{j-1}}^{s_{j+1}} u^n(s)\, dx(u^n; s), \tag{2.11}$$

where the integrand above is taken in a distributional sense at the point of discontinuity x_j. Condition (2.6), together with integration by parts reveals that (2.11) is equal to $\bar{u}_j^{n+1}$. Following Brenier [2], it can be shown by a change of variables that (2.11) is also equal to

$$\frac{1}{\Delta x} \int_{x_{j-1}}^{x_{j+1}} \mathcal{T}(\Delta t^n, u^n)(x)\, dx,$$

where $\mathcal{T}(t, u)$ is the transport-collapse operator. Since the transport-collapse approximation satisfies the result of the lemma, it now follows that $\bar{u}_j^{n+1}$ does as well. ∎

At this point we impose the following requirements on the local reconstruction operator $\mathcal{R}_j$. For each $j \in \mathcal{J}^n$ and every continuous function $u(x)$, we require that

$$\int_{I_j} \mathcal{R}_j(u)(x)\, dx = \int_{I_j} u(x)\, dx, \tag{2.12a}$$

and for all $x \in I_j$,

$$\min_{x \in I_j} u(x) \leqslant \mathcal{R}_j(u)(x) \leqslant \max_{x \in I_j} u(x), \tag{2.12b}$$

and

$$\mathrm{Var}(\mathcal{R}_j(u))|_{I_j} \leqslant \mathrm{Var}(u)|_{I_j}, \tag{2.12c}$$

where $\mathrm{Var}(u)|_I$ denotes the variation of u on the interval I. Defining an approximate solution $u^\Delta(x, t)$ of (2.1) by

$$u^\Delta(x, t) = \sum_{n=0}^{\infty} \mathcal{S}(t - t^n, u^n)(x)\, \chi_{[t^n, t^{n+1})}(t), \tag{2.13}$$

we have the following theorem.

THEOREM 2.1. *Suppose the reconstruction satisfies* (2.12) *and* u_0 *has bounded total variation. Moreover, suppose that each* Δt^n *satisfies* (2.6) *and that there is a constant* $c > 0$ *independent of* Δx *and* n *such that* $\Delta t^n \geqslant c\,\Delta x$. *Then there is a sequence of* Δx*'s tending to zero and a weak solution of* (2.1), *say* $u(x, t)$, *such that* $u^\Delta(x, t) \to u(x, t)$. *Furthermore,* u^Δ *satisfies the uniform estimates*

$$\min_{x \in \mathbf{R}} u_0(x) \leqslant u^\Delta(x, t) \leqslant \max_{x \in \mathbf{R}} u_0(x),$$

$$\mathrm{Var}(u^\Delta(\cdot, t)) \leqslant \mathrm{Var}(u_0),$$

for all $t > 0$.

Remark. The point values and cell averages need not be computed exactly for the results of the theorem to remain valid. For example, one could approximate the solution s_j of (2.8). If the resulting approximate cell average $\bar{u}_j^{n+1}$ satisfies the estimate of Lemma 2.2, the results of Theorem 2.1 remain true at discrete times t^n.

Finally, we assume that (2.1) is a hyperbolic system of equations. Specifically, let $u \in \mathbf{R}^m$ and $f(u) \in \mathbf{R}^m$ and suppose the Jacobian matrix of $f(u)$, denoted here by $Df(u) \in \mathbf{R}^{m \times m}$, has m real eigenvalues $\lambda_k(u)$, $k = 1, \ldots, m$, and a complete set of eigenvectors. The characteristic equation (2.7) is replaced for systems by

$$x_k(u; s) = \lambda_k(u(s))\,\Delta t^n + s, \tag{2.14}$$

for $k = 1, \ldots, m$. Unlike the scalar case, we cannot expect u or any function of u to be constant along characteristics, unless, of course, $f(u)$ is linear or u is constant. Nevertheless, we mimic the scalar approach taken in (2.8)–(2.10). Given $u^n(x)$, for each $j \in \mathcal{J}^n$ and each characteristic family $k = 1, \ldots, m$, solve the characteristic equations

$$x_k(u^n; s_{k,j}) = x_j, \tag{2.15}$$

for the unknowns $s_{k,j}$. Evaluate $l_{k,j} = l_k(u^n(s_{k,j}))$, where $l_k(u)$ is the kth left eigenvector of $Df(u)$ and $\lambda_{k,j} = \lambda_k(u^n(s_{k,j}))$. Note that the linearized characteristic

$$x_{k,j}(t) = \lambda_{k,j} t + s_{k,j}$$

goes through the points $(s_{k,j}, 0)$ and $(x_j, \Delta t^n)$ for each $k = 1, \ldots, m$. Again, let $\mathcal{S}(t, v)(x)$ represent the vector valued

solution operator of (2.1) with initial data $v(x) \in \mathbf{R}^m$, and to simplify notation let

$$\mathcal{S}_{k,j}(t) = \mathcal{S}(t, u^n)(x_{k,j}(t))$$

represent the exact solution of (2.1) with data $u^n(x)$ along the line $(x_{k,j}(t), t)$. Using (2.1), we have that

$$\frac{\partial}{\partial t} l^t_{k,j} \mathcal{S}_{k,j}(0) + \lambda_{k,j} \frac{\partial}{\partial x} l^t_{k,j} \mathcal{S}_{k,j}(0) = 0.$$

Therefore, by Taylor's theorem we find that

$$l^t_{k,j} \mathcal{S}_{k,j}(t) = l^t_{k,j} u^n(s_{k,j}) + O(t^2), \tag{2.16a}$$

and a similar calculation yields

$$\begin{aligned} l^t_{k,j}(f(\mathcal{S}_{k,j}(t)) - \lambda_{k,j} \cdot \mathcal{S}_{k,j}(t)) \\ = l^t_{k,j}(f(u^n(s_{k,j})) \\ - \lambda_{k,j} \cdot u^n(s_{k,j})) + O(t^2). \end{aligned} \tag{2.16b}$$

The solution u_j^{n+1} of the m linear equations,

$$l^t_{k,j} u_j^{n+1} = l^t_{k,j} u^n(s_{k,j}), \tag{2.16c}$$

determines $\mathcal{S}(\Delta t^n, u^n)(x_j)$ to second order. (The linear system is always well conditioned provided that $u^n(x)$ has small local variation.) To determine a second-order flux approximation we apply the divergence theorem to (2.1) over the triangle defined by the points $(x_j, 0)$, $(x_j, \Delta t^n)$, and $(s_{k,j}, 0)$ to find that

$$\begin{aligned} \int_0^{\Delta t^n} f(\mathcal{S}(t, u^n)(x_j))\, dt \\ = \int_{s_{k,j}}^{x_j} u^n(x)\, dx \\ + \int_0^{\Delta t^n} (f(\mathcal{S}_{k,j}(t)) - \lambda_{k,j} \cdot \mathcal{S}_{k,j}(t))\, dt. \end{aligned} \tag{2.16d}$$

Multiplying (2.16d) by $l^t_{k,j}$ and inserting (2.16b) into the result, the solution f_j^n of the m linear equations,

$$\begin{aligned} l^t_{k,j} f_j^n = l^t_{k,j} \left(\frac{1}{\Delta t^n} \int_{s_{k,j}}^{x_j} u^n(x)\, dx \right. \\ \left. + (f(u^n(s_{k,j})) - \lambda_{k,j} \cdot u^n(s_{k,j})) \right), \end{aligned} \tag{2.16e}$$

determines

$$\frac{1}{\Delta t^n} \int_0^{\Delta t^n} f(\mathcal{S}(t, u^n)(x_j))\, dt$$

to second order, where the error is a smooth function of $u^n(x)$.

Remark. Higher-order Taylor approximations can be constructed as above along the linearized characteristics. However, these would require evaluation of at least second-order derivatives of $f(u)$.

We summarize the procedure for nonlinear hyperbolic systems. Given $u^n(x) \in \mathbf{R}^m$:

(1) For each $j \in \mathcal{J}^n$ and $k = 1, \ldots, m$ solve

$$\lambda_k(u^n(s_{k,j}))\, \Delta t^n + s_{k,j} = x_j$$

for the unknowns $s_{k,j}$ and evaluate each of the following: $u^n(s_{k,j})$, $\lambda_k(u^n(s_{k,j}))$, $l_k(u^n(s_{k,j}))$, and $f(u^n(s_{k,j}))$. Then solve (2.16c) for the approximate endpoints u_j^{n+1} and solve (2.16e) for the approximate flux terms f_j^n.

(2) For each $j \in \mathcal{J}^{n+1}$ compute the approximate cell averages of $\mathcal{S}(\Delta t^n, u^n)$,

$$\bar{u}_j^{n+1} = \frac{1}{\Delta x} \int_{I_j} u^n(x)\, dx - \sigma^n(f_{j+1}^n - f_{j-1}^n).$$

(3) Set $u^{n+1}(x) = \mathcal{R}^{n+1}(\tilde{\mathcal{S}}(\Delta t^n, u^n))(x)$, where $\tilde{\mathcal{S}}(\Delta t^n, u^n)$ is an approximation to the exact solution $\mathcal{S}(\Delta t^n, u^n)$ obtained from endpoints and cell averages determined in (1) and (2) above.

Repeat (1)–(3) to advance the approximation to successive time levels.

Remark. Formulae (2.16c) and (2.16e) yield exact information for linear systems. Therefore, the results of Theorem 2.1 remain valid for linear systems where $l^t_k u$ takes the role of u found there.

We now consider nonlinear systems of equations in two space dimensions. The discussion is restricted to two dimensions for ease of presentation only.

We apply the procedure of dimensional splitting. Consider the conservation law

$$\begin{aligned} \frac{\partial}{\partial t} u + \frac{\partial}{\partial x} f(u) + \frac{\partial}{\partial y} g(u) = 0 \\ u(x, y, 0) = u_0(x, y). \end{aligned}$$

Given $u^n(x, y)$ approximating $u(x, y, t^n)$, perform two x-sweeps to generate an approximation $\tilde{u}^n(x, y)$ to $v(x, y, t)$ at $t = \Delta t^n + \Delta t^{n+1}$, where $v(x, y, t)$ is the solution to

$$\begin{aligned} \frac{\partial}{\partial t} v + \frac{\partial}{\partial x} f(v) = 0 \\ v(x, y, 0) = u^n(x, y). \end{aligned}$$

Then perform two y-sweeps to generate an approximation $u^{n+2}(x, y)$ to $w(x, y, t)$ at $t = \Delta t^n + \Delta t^{n+1}$, where $w(x, y, t)$ is the solution to

$$\frac{\partial}{\partial t} w + \frac{\partial}{\partial y} g(w) = 0$$

$$w(x, y, 0) = \tilde{u}^n(x, y).$$

Using the notation above with x and y subscripts denoting directions in which to apply the $\mathscr{R}$ and $\mathscr{S}$ operators, compute

$$\tilde{u}^n = \mathscr{R}_x^{n+2}(\mathscr{S}_{x(\Delta t}{}^{n+1}, \mathscr{R}_x^{n+1}(\mathscr{S}_x(\Delta t^n, u^n)))),$$

and then

$$u^{n+2} = \mathscr{R}_y^{n+2}(\mathscr{S}_y(\Delta t^{n+1}, \mathscr{R}_y^{n+1}(\mathscr{S}_y(\Delta t^n, \tilde{u}^n)))).$$

Two consecutive time steps are performed in each direction in order to avoid dealing with multidimensional staggered grids. To symmetrize treatment of x and y, alternate the order of x and y sweeps.

A complication arises from the use of points values in the reconstruction operators: With a standard dimensional splitting procedure, after the x-sweeps there would be no values available for point values needed for the initial y-sweep reconstruction, and vice versa. We overcome this difficulty by representing all two-dimensional functions $u^n(x, y)$ and $\tilde{u}^n(x, y)$ in the form $\{v_{i,j}: i, j \in \text{integer}\}$, where

$$v_{i,j} = \begin{cases} \dfrac{1}{\Delta x\, \Delta y} \displaystyle\int_{I_j} \int_{I_i} u(x, y)\, dx\, dy, & i \in \mathscr{J}^0, j \in \mathscr{J}^0 \\ \dfrac{1}{\Delta y} \displaystyle\int_{I_j} u(x_i, y)\, dy, & i \in \mathscr{J}^1, j \in \mathscr{J}^0 \\ \dfrac{1}{\Delta x} \displaystyle\int_{I_i} u(x, y_j)\, dx, & i \in \mathscr{J}^0, j \in \mathscr{J}^1 \\ u(x_i, y_j), & i \in \mathscr{J}^1, j \in \mathscr{J}^1. \end{cases}$$

During an x-sweep of the splitting procedure, along lines in x with $j \in \mathscr{J}^0$, we perform the one-dimensional x algorithm on x quantities averaged in y over I_j. Along lines in x with $j \in \mathscr{J}^1$ we perform the one-dimensional x algorithm on quantities associated with points y_j. The roles of x and y above are reversed during a y-sweep. In this manner we always maintain necessary quantities needed by the reconstruction operators. Unfortunately, this approach requires more work than a standard dimensional splitting procedure—twice as many sweeps in two dimensions and four times as many sweeps in three dimensions.

For a scalar multidimensional equation, the maximum principle remains in effect for this dimensional splitting procedure when it is combined with the one-dimensional procedure described above. As is usually the case, however, we cannot prove that the multidimensional variation of the approximation remains uniformly bounded.

2.2. *Reconstruction from Cell Averages and Point Values*

The most straightforward cell average preserving reconstruction is to set the local reconstruction equal to the cell average. This piecewise constant reconstruction leads to the well-known Lax–Friedrichs difference scheme. As is also well known, this scheme performs poorly due to first-order accuracy and excessive numerical diffusion. We begin our discussion of higher order reconstruction by considering the scalar equation. A piecewise linear reconstruction is obtained by the following recipe. Define the "minmod" function $S(x, y)$ by

$$S(x, y) = \begin{cases} \operatorname{sgn}(x) \cdot \min(|x|, |y|), & \text{if } x \cdot y \leqslant 0 \\ 0, & \text{if } x \cdot y > 0, \end{cases}$$

define the jth cell average of $u(x)$ by

$$\bar{u}_j = \frac{1}{\Delta x} \int_{I_j} u(x)\, dx$$

and normalized cell end point values by

$$\hat{u}_{\mathrm{L}} = u(x_{j-1}) - \bar{u}_j,$$

$$\hat{u}_{\mathrm{R}} = u(x_{j+1}) - \bar{u}_j.$$

A particular linear local reconstruction of $u(x)$ using exact point values and cell averages can be built in the usual minmod fashion;

$$\mathscr{R}_j(u)(x) = \bar{u}_j + L\left(\hat{u}_{\mathrm{R}}, \hat{u}_{\mathrm{L}}; \frac{(x - x_j)}{\Delta x}\right), \tag{2.17}$$

where

$$L(a, b; \theta) = 2S(a, b)\theta,$$

so that $\mathscr{R}_j(u)(x)$ is the unique continuous piecewise linear function with $\mathscr{R}_j(u)(x_{j-1}) = \bar{u}_j - S(\hat{u}_{\mathrm{R}}, \hat{u}_{\mathrm{L}})$ and $\mathscr{R}_j(u)(x_{j+1}) = \bar{u}_j + S(\hat{u}_{\mathrm{R}}, \hat{u}_{\mathrm{L}})$. The formula above is exact when acting on linear functions. It is also easy to verify that formula (2.17) satisfies all the requirements of (2.12). In a sense, however, this linear formula wastes point value information. As defined in (2.17), $\mathscr{R}_j(u)(x)$ usually can take on only one of its point values. By increasing the number of degrees of freedom in the basic interpolation to three, both point values and, hence, more information can be honored in the

reconstruction. We are therefore led to consider a piecewise quadratic reconstruction of the form

$$\mathscr{R}_j(u)(x) = \bar{u}_j + Q\left(\hat{u}_R, \hat{u}_L; \frac{(x-x_j)}{\Delta x}\right),$$

where

$$Q(a, b; \theta) = 3(a+b)\theta^2 + (a-b)\theta - (a+b)/4.$$

$Q(a, b; \theta)$ is the unique quadratic function with the properties that $Q(a, b; \frac{1}{2}) = a$, $Q(a, b; -\frac{1}{2}) = b$, and $\int_{-1/2}^{1/2} Q(a, b; \theta)\, d\theta = 0$. While this reconstruction is designed to satisfy (2.12a) and to be exact when acting on quadratic functions, one easily sees that it can violate (2.12b). Possible overshoots can be eliminated in a variety of ways by modifying the point values fed into the formula for Q. One possibility is the modified quadratic

$$\mathscr{R}_j(u)(x) = \bar{u}_j + Q\left(S(\hat{u}_R, 2\hat{u}_L), S(\hat{u}_L, 2\hat{u}_R); \frac{(x-x_j)}{\Delta x}\right), \tag{2.18}$$

which can be shown to damp out all the overshoots coming from the basic quatratic reconstruction. Moreover, it is possible to modify (2.18) in such a way so as to recover the quadratics exactly. We refer the reader to [13 or 14] for further details on this topic.

The performance of the high order reconstruction methods above, combined with the evolution algorithm from Section 2.1, is surprisingly good [13]. This fact runs contrary to the generally held belief that staggered grid methods such as the Lax–Friedrichs scheme are easy to program but overly diffusive. Numerical results indicate that the high order staggered grid methods in [13, 14] are excessively diffusive only when dealing with very small CFL numbers, especially with linear waves. In fact, discontinuity smearing from these methods is most pronounced when solving the equation $u_t = 0$ and clearly should be expected in this case. This shortcoming can be completely eliminated by allowing the local reconstruction to have a hinge at the center of its cell of definition.

Consider the basic hinged linear reconstruction, given by

$$\mathscr{R}_j(u)(x) = \bar{u}_j + H\left(\hat{u}_R, \hat{u}_L; \frac{(x-x_j)}{\Delta x}\right),$$

where

$$H(a, b; \theta) = 2(a+b)\,|\theta| + (a-b)\theta - (a+b)/2.$$

$H(a, b; \theta)$ is the unique continuous piecewise linear function $H(a, b; \frac{1}{2}) = a$, $H(a, b; -\frac{1}{2}) = b$ and $\int_{-1/2}^{1/2} H(a, b; \theta)\, d\theta = 0$. As with the quadratic formula above, the hinge formula automatically satisfies (2.12a) but may violate (2.12b) unless limited. To limit, we modify the point values being fed into the formula for H as

$$\mathscr{R}_j(u)(x) = \bar{u}_j + H\left(S(\hat{u}_R, \rho\hat{u}_L), S(\hat{u}_L, \rho\hat{u}_R); \frac{(x-x_j)}{\Delta x}\right). \tag{2.19c}$$

where $S(x, y)$ is the "minmod" function defined above. We find it convenient to write (2.19a) as

$$\mathscr{R}_j(u)(x) = \bar{u}_j + \begin{cases} +\dfrac{(x-x_j)}{\Delta x}(v^L + 3v^R) \\ \quad -\dfrac{v^L + v^R}{2} & \text{if} \quad x \geqslant x_j \\ -\dfrac{(x-x_j)}{\Delta x}(3v^L + v^R) \\ \quad -\dfrac{v^L + v^R}{2} & \text{if} \quad x < x_j, \end{cases} \tag{2.19b}$$

where

$$v^L = S(\hat{u}_L, \rho\hat{u}_R)$$
$$v^R = S(\hat{u}_R, \rho\hat{u}_L).$$

By inspection, $\rho = 3$ is the limiting case in the definition above to ensure that no interior extrema are created. It is easy to show that (2.19) yields a reconstruction which satisfies all requirements (2.12) for any choice of parameter ρ with $0 \leqslant \rho \leqslant 3$. In fact, setting $\rho = 0$ recovers the first-order piecewise constant reconstruction and setting $\rho = 1$ recovers the usual piecewise linea minmod reconstruction. Figures 1 and 2 illustrate situations where the difference between the reconstruction operators are most pronounced. Figure 1 depicts the hinged reconstruction (2.19) using $\rho = 0$, 1, 3 and the quadratic reconstruction (2.18), both acting on a function with cell average $\bar{u}_j = 1$ and endpoints $u(x_{j-1}) = 0$, $u(x_{j+1}) = 5$. Figure 2 depicts the results of applying 80 time steps of the staggered grid procedure to the equation

$$\frac{\partial}{\partial t} u = 0$$

$$u(x, 0) = \begin{cases} 1 & \text{if} \quad |x - 0.5| \leqslant 0.05 \\ 0 & \text{otherwise,} \end{cases}$$

with $\Delta x = 0.01$, using the same reconstructions as in Fig. 1. Except for the hinged reconstruction using $\rho = 3$, the results are excessively smeared. By far the most diffusive results are

(2) For each $j \in \mathscr{J}^{n+1}$ obtain the new approximate cell average from the formula

$$\bar{u}_j^{n+1} = \frac{1}{\Delta x}\int_{I_j} u^n(x)\,dx - \sigma^n(f_{j+1}^n - f_{j-1}^n),$$
$$\left(\sigma^n = \frac{\Delta t^n}{\Delta x}\right).$$

(3) The HINGE reconstruction operator defined by (2.19) is used to generate the approximate solution $u^{n+1}(x)$ at time t^{n+1} defined in the interval I_j, $j \in \mathscr{J}^{n+1}$, by

$$u^{n+1}(x) = \mathscr{R}_j(u^n(s_{j-1}), \bar{u}_j^{n+1}, u^n(s_{j+1}); (x - x_j)/\Delta x).$$

The notation in (3) reflects the fact that (2.19) is applied to approximate cell averages and endpoints. The only simplification above compared to the algorithm presented in Section 2 is the use of the frozen backward characteristic equation.

For a system of $m > 1$ equations it is appropriate to organize the computations so that the reconstruction operator precedes the approximate solution operator and uses the same eigenvalue—eigenvector expansions. We tart with the unreconstructed approximate solution represented in each cell I_j, $j \in \mathscr{J}^n$, by the values $u^n(x_{j-1})$, $\bar{u}_j^n$, $u^n(x_{j+1})$. For each $j \in \mathscr{J}^n$ we evaluate $\lambda_k(\bar{u}_j^n)$, $l_k(\bar{u}_j^n)$, $r_k(\bar{u}_j^n)$, $k = 1, ..., m$, where $\lambda_k(u)$, $l_k(u)$, $r_k(u)$ are respectively the eigenvalues, left eigenvectors, right eigenvectors of the matrix $Df(u)$. The eigenvectors are normalized so that $l_k(u) \cdot r_{k'}(u) = \delta_{k,k'}$. Therefore,

$$Df(\bar{u}_j^n) = R_j \Lambda_j L_j^t = R_j \Lambda_j R_j^{-1},$$

where

$$L_j = (l_1(\bar{u}_j^n), ..., l_m(\bar{u}_j^n))$$
$$\Lambda_j = \operatorname{diag}(\lambda_1(\bar{u}_j^n), ..., \lambda_m(\bar{u}_j^n))$$
$$R_j = (r_1(\bar{u}_j^n), ..., r_m(\bar{u}_j^n)).$$

For the equations of gas dynamics, the eigenvalues and eigenvectors of the Jacobian matrix $Df(u)$ can be computed analytically and relatively inexpensively, and $Df(u)$ for these equations is never evaluated directly; see the Appendix. In more general situations, an appropriate linear algebraic eigenvalue routine must be used.

We now compute the reconstruction $u^n(x)$ from point values and cell averages. As in (2.21), we change these values into characteristic variables. For $j \in \mathscr{J}^n$ we set

$$c_j^L = L_j^t u^n(x_{j-1}), \qquad \bar{c}_j = L_j^t \bar{u}_j^n,$$
$$c_j^R = L_j^t u^n(x_{j+1}). \tag{3.1}$$

For $k = 1, ..., m$ we compute normalized limited characteristic endpoint values according to the reconstruction (2.19b)

$$c_{k,j}^{LH} = S((c_{k,j}^L - \bar{c}_{k,j}), 3(c_{k,j}^R - \bar{c}_{k,j}))$$
$$c_{k,j}^{RH} = S((c_{k,j}^R - \bar{c}_{k,j}), 3(c_{k,j}^L - \bar{c}_{k,j})), \tag{3.2}$$

where $S(x, y)$ is the minmod function. c^{LH} and c^{RH} represent limited hinge reconstruction endpoint values minus the cell average $\bar{c}$. We also compute limited characteristic midpoint values

$$c_{k,j}^{MH} = \bar{c}_{k,j} - (c_{k,j}^{LH} + c_{k,j}^{RH})/2. \tag{3.3}$$

To advance the approximation to the next time level, we need, as in the scalar case,

$$s_{k,j} = x_j - \lambda_k(\bar{u}_j^n)\,\Delta t^n$$
$$c_{k,j}^{SH} = \mathscr{R}_j(c_{k,j}^L, c_{k,j}, c_{k,j}^R; (s_{k,j} - x_j)/\Delta x)$$

or equivalently,

$$\theta^* = -\sigma^n \lambda_k(\bar{u}_j^n)$$
$$c_{k,j}^{SH} = 2(c_{k,j}^{RH} + c_{k,j}^{LH})\,|\theta^*| + (c_{k,j}^{RH} - c_{k,j}^{LH})\theta^* + c_{k,j}^{MH}. \tag{3.4}$$

For systems, SHINGE only computes part of the numerical flux in characteristic variables. This reduces the number of evaluations of $f(u)$ to one per grid cell. We set

$$(cf)_{k,j}^{\text{partial}} = \frac{1}{\Delta t^n}\int_{s_{k,j}}^{x_j} \mathscr{R}_j(c_{k,j}^L, \bar{c}_{k,j}, c_{k,j}^R; (x - x_j)/\Delta x)\,dx - \lambda_k(\bar{u}_j^n) \cdot c_{k,j}^{SH}$$

or, equivalently,

$$(cf)_{k,j}^{\text{partial}} = \tfrac{1}{2}(c_{k,j}^{MH} - c_{k,j}^{SH}) \cdot \lambda_k(\bar{u}_j^n), \tag{3.5}$$

which follows by explicit integration of the HINGE reconstruction. Changing back into the conserved variable, we compute for each $j \in \mathscr{J}^n$

$$u^n(x_{j-1} + 0) = R_j c_j^{LH} + \bar{u}_j^n$$
$$u^n(x_{j+1} - 0) = R_j c_j^{RH} + \bar{u}_j^n$$
$$u^{n+1}(x_j) = R_j c_j^{SH}$$
$$f_j^{\text{partial}} = R_j (cf)_j^{\text{partial}}, \tag{3.6}$$

and to complete the evaluation of the numerical flux f_j^n, we evaluate $f(u^{n+1}(x_j))$ and set

$$f_j^n = f_j^{\text{partial}} + f(u^{n+1}(x_j)). \tag{3.7}$$

Finally, to determine the new approximate cell average at time level $n+1$, for each $j \in \mathcal{J}^{n+1}$ we compute

$$\bar{u}_j^{n+1} = \tfrac{1}{4}(u^n(x_{j-1}) + u^n(x_j - 0) + u^n(x_j + 0) + u^n(x_{j+1})) - \sigma^n(f^n_{j+1} - f^n_{j-1}). \tag{3.8}$$

This completes the calculations for one time step.

Remark. The number of matrix vector multiplies in (3.1) and (3.6) can be reduced from 7 to 6 by modifying f^{partial} so that the average of the reconstruction over the staggered cell at time level n is included in the flux terms.

Our computational treatment of dimensional splitting appears to be standard. We store only one two-dimensional array containing the representation of $u^n(x, y)$ in the format of Section 2.1. Before each double sweep over a one-dimensional line, we copy the relevant parts of $u^n(x, y)$ into temporary one-dimensional arrays. This saves substantial storage and incurs only minor CPU overhead.

In our codes, the key computations (3.2)–(3.8) are all performed in one vectorizing double loop on k and j, requiring about 20 Fortran statements. All loops on j in our code vectorize using standard Fortran. The codes solving the "ramp" and "step" problems presented in the next section each have fewer than 500 Fortran statements, including about 120 statements to evaluate eigenvalues and eigenvectors and 110 statements to set up storage and write output files. For two-dimensional gas dynamics, $m = 4$, and the subroutines that require the most CPU time are the matrix-vector multiply routine used to change to and from characteristic variables (around 40% of total CPU time), the routine performing computations (3.2)–(3.8) (30%), and the routine to evaluate eigenvalues and eigenvectors of the matrix $DF(u)$ (25%).

We experimented with variants of SHINGE that avoid some of the drawbacks (1.4)–(1.6). One variant was applied on a fixed, non-staggered grid, allowing us to increase the timestep stability limit to its usual value. Initial data for tracebacks were made continuous by, essentially, applying "phantom" staggered grid timesteps with $\Delta t = 0$. For the ramp problem described in the next section, this method required extra limiting along the lower boundary to keep the "jet" located at $x = 2.6$ in Fig. 9 from shooting ahead and impinging on the shock located at $x = 2.8$ (whether the jet actually interacts with the leading shock or not has not yet been justified by theoretical considerations). Otherwise, results were very similar to SHINGE. Another variant avoided drawback (1.6) by replacing characteristic traceback to determine predicted point values with PPM-type bicubic interpolation [4]. The resulting method was somewhat more diffusive than SHINGE for all problems, demonstrating the advantage of using characteristic information when it is available. In practice, SHINGE remains our current method of choice, as the simplest method that reliably achieves resolution comparable to the other variants.

4. NUMERICAL RESULTS

All results presented here were performed on a Cray XMP computer using SHINGE. For comparison purposes, results are presented on the same grids, to the same final times, and with the same values plotted as in [15, 16].

In all two-dimensional examples, time steps are chosen so that the input parameter $CFL = 0.8$, where

$$\Delta t^n + \Delta t^{n+1} = CFL \min_{i,j,k} \left(T_{\text{final}} - t^n, \frac{\Delta x}{\lambda^x_{k,i,j}(\bar{u}^n_{i,j})}, \frac{\Delta y}{\lambda^y_{k,i,j}(\bar{u}^n_{i,j})} \right),$$

where λ^x are the eigenvalues of $Df(u)$ and λ^y are the eigenvalues of $Dg(u)$. Timesteps for the one-dimensional examples are chosen similarly.

Boundary conditions are enforced by the "fictitious cell' method. For example, a reflecting left boundary condition at $x_0 = 0$ is handled by setting

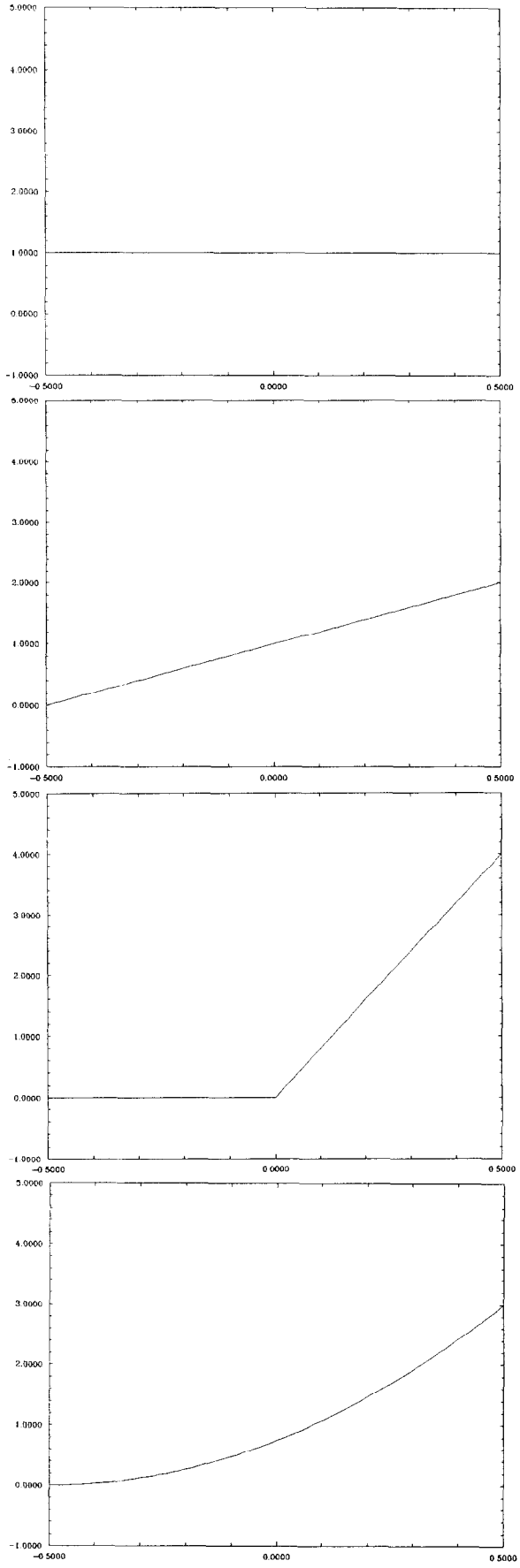
$$(u^n(x_0^-))_k = \sigma_k (u^n(x_0^+))_k$$
$$(\bar{u}^n_{-1})_k = \sigma_k (\bar{u}^n_1)_k$$
$$(u^n(x^+_{-2}))_k = \sigma_k (u^n(x_2^-))_k,$$

where $\sigma_k = -1$ when the k component corresponds to the normal velocity, and $\sigma_k = 1$ otherwise. An inflow or outflow left boundary condition would be handled by setting $u^n(x_0^+)$, $\bar{u}^n_{-1}$, and $u^n(x^-_{-2})$ to specified boundary values. Note that the three values set correspond to one cell, two point values and one cell average.

We now briefly describe our four test problems: "Sod," "Lax" [15], "step," and "ramp" [16]. The Sod and Lax problems are one-space-dimensional with $m = 3$ and

$$u = \begin{pmatrix} \rho \\ \rho u \\ \rho E \end{pmatrix}, \qquad f(u) = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho u E + up \end{pmatrix},$$

where ρ, u, E, and p are respectively density, x-velocity, total energy per unit mass, and pressure of a γ-law gas with $\gamma = 1.4$. The pressure satisfies the equation of state $p = 0.4\rho e$, where $e = E - u^2/2$ is the internal energy per unit mass. The



Even when $\rho = 3$, (2.19) may chop off local extrema in u. This can be avoided by tracking extreme values of u from the previous time step and limiting the reconstruction only if the tracked values are violated. One way to do this is to let $\hat{E}$ denote the normalized relevant extreme value of $u(x)$ over the interval I_j defined by

$$\hat{E}(x, y) = \begin{cases} \max\limits_{I_j}(u) - \bar{u}_j, & \text{if } x + y < 0 \\ \min\limits_{I_j}(u) - \bar{u}_j, & \text{if } x + y \geqslant 0, \end{cases}$$

and define a limiter $S_E(x, y)$ by

$$S_E(x, y) = \begin{cases} \operatorname{sgn}(x) \cdot \min(|x|, |2\hat{E}(x, y)| + |y|), \\ \qquad \text{if } x \cdot y \leqslant 0 \\ \operatorname{sgn}(x) \cdot \min(|x|, |2x\hat{E}(x, y)|/|x + y|), \\ \qquad \text{if } x \cdot y > 0. \end{cases}$$

Notice that $S_E(\hat{u}_R, \hat{u}_L)$ is contained in the interval $[0, \hat{u}_R]$ and $S_E(\hat{u}_L, \hat{u}_R)$ is contained in the interval $[0, \hat{u}_L]$. Moreover, S_E is constructed so that

$$\frac{S_E(\hat{u}_R, \hat{u}_L) + S_E(\hat{u}_L, \hat{u}_R)}{2}$$

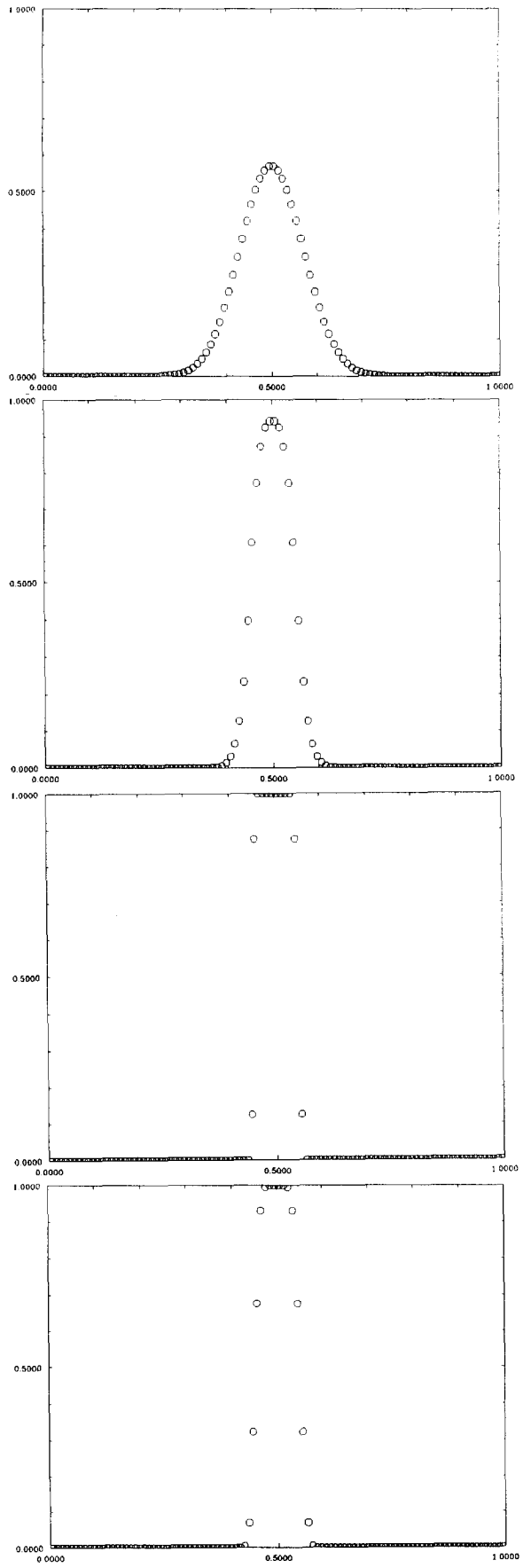
is contained in the interval $[0, \hat{E}(\hat{u}_L, \hat{u}_R)]$. Therefore, inserting this limiter into the basic hinge formula, we have that

$$\mathscr{R}_j(u)(x) = \bar{u}_j + H\left(S_E(\hat{u}_R, \hat{u}_L), S_E(\hat{u}_L, \hat{u}_R); \frac{(x - x_j)}{\Delta x}\right) \tag{2.20}$$

yields a reconstruction which satisfies all requirements of (2.12). Moreover, (2.20) is exact when acting on any function of the form $H(a, b; (x - x_j)) + c$.

Remark. Computing the exact value of the relevant extreme value of $\mathscr{S}(\Delta t, u^n)(x)$ over cell I_j is not necessary to retain the discrete time results of Theorem 2.1. The extreme value of $u^n(x)$ over the domain of dependence $[s_{j-1}, s_{j+1}]$ for cell I_j will suffice in order to guarantee the stability results found in the theorem. While limiting in this manner does not reflect the possible extremum decay from nonlinear interactions over a single time step, numerical results have always justified this approach.

FIG. 1. Hinge reconstruction with respectively $\rho = 0$, 1, 3 and quadratic reconstruction on test data with cell average equal to 1, left endpoint equal to 0 and right endpoint equal to 5.



The scalar reconstructions above are generalized to systems of equations by limiting in the characteristic components $(\cdot, l_{k,j})$. Specifically, let $l_{k,j} = l_k(\bar{u}_j)$ and let $r_{k,j} = r_k(\bar{u}_j)$, where $r_k(u)$ is the kth right eigenvector of $Df(u)$ normalized so that $(l_k(u), r_{k'}(u)) = \delta_{k,k'}$. Here $(\cdot, \cdot)$ denotes the usual vector inner product and $\delta_{k,k'}$ denotes the Kronecker delta. Set

$$\mathscr{R}_j(u)(x) = \sum_{k=1}^{m} \mathscr{R}_j^{(k)}(u)(x) \cdot r_{k,j}, \tag{2.21}$$

where the scalar $\mathscr{R}_j^{(k)}(u)(x)$ is computed using one of the scalar reconstruction algorithms (2.17)–(2.20) and using $(u(x_{j-1}), l_{k,j})$ and $(u(x_{j+1}), l_{k,j})$ as point values and $(\bar{u}_j, l_{k,j})$ as cell averages. For linear systems, each characteristic component of the resulting reconstruction satisfies the requirements of (2.12) and hence the results of Theorem 2.1 apply.

Hinged reconstruction (2.19), combined with the staggered evolution scheme of Section 2.1, defines what we refer to as HINGE.

3. A SIMPLIFIED ALGORITHM

In this section we present a simplified method which we refer to as SHINGE. This simplified method, while not having all the theoretical features for nonlinear problems as HINGE, performs essentially as well with fewer coding requirements and faster execution time per time step. The main differences between HINGE and SHINGE are that in SHINGE the eigenvalues and eigenvectors of the matrix $Df(u)$ are only evaluated at cell averages and the flux function $f(u)$ is only evaluated once per cell per time step.

HINGE and SHINGE are identical for linear systems of conservation laws. For a nonlinear scalar equation, the SHINGE evolution algorithm is: Given $u^n(x)$,

(1) For each $j \in \mathscr{J}^n$ evaluate $f'(\bar{u}_j^n)$ and set

$$s_j = x_j - f'(\bar{u}_j^n)\, \Delta t^n.$$

Evaluate $u^n(s_j)$ and $f(u^n(s_j))$ and compute the numerical flux

$$f_j^n = \frac{1}{\Delta t^n} \int_{s_j}^{x_j} u^n(x)\, dx + f(u^n(s_j)) - f'(\bar{u}_j^n) \cdot u^n(s_j).$$

FIG. 2. 80 time iterations of test problem $u_t = 0$ with pulse initial data using hinge reconstruction with $\rho = 0$, 1, 3 and quadratic reconstruction.

sound speed is $c=(1.4p/\rho)^{1/2}$. The one-space-dimensional conservation laws we approximate are therefore

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

The step and ramp problems are two-space-dimensional with $m=4$ and

$$u=\begin{pmatrix}\rho\\ \rho u\\ \rho v\\ \rho E\end{pmatrix},\qquad f(u)=\begin{pmatrix}\rho u\\ \rho u^2+p\\ \rho uv\\ \rho uE+up\end{pmatrix},$$

$$g(u)=\begin{pmatrix}\rho v\\ \rho uv\\ \rho v^2+p\\ \rho vE+vp\end{pmatrix},$$

where here v is the y-velocity and $e=E-(u^2+v^2)/2$. The two-space-dimensional conservation laws therefore have the form

$$\frac{\partial}{\partial t}u+\frac{\partial}{\partial x}f(u)+\frac{\partial}{\partial y}g(u)=0.$$

The Sod and Lax problems are simple Riemann problems given respectively by the initial data

$$(\rho(x),u(x),p(x))=\begin{cases}(1,0,1) & \text{if } x<0.5\\ (0.125,0,1) & \text{if } x\geqslant 0.5\end{cases}$$

and

$$(\rho(x),u(x),p(x))=\begin{cases}(0.445,0.698,3.528)\\ \quad\text{if } x<0.5\\ (0.5,0,0.571)\\ \quad\text{if } x\geqslant 0.5.\end{cases}$$

100 equally spaced grid points are used over the interval $[0,1]$ for both problems, with $T_{\text{final}}=0.18$ (corresponding to 97 half time steps with $CFL=0.8$) for the Sod problem (Fig. 3), and $T_{\text{final}}=0.15$ (corresponding to 176 half time steps) for the Lax problem (Fig. 4).

The step problem is solved on the domain $(x,y)\in(0,3)\times(0,1)$ with the "step" $(0.6,3)\times(0,0.2)$ omitted; t runs from $t=0$ to $t=4$, and the initial data are

$$u(x,y,0)=u_0(x,y)=\begin{pmatrix}1.4\\ 3\\ 0\\ 6.55\end{pmatrix}.$$

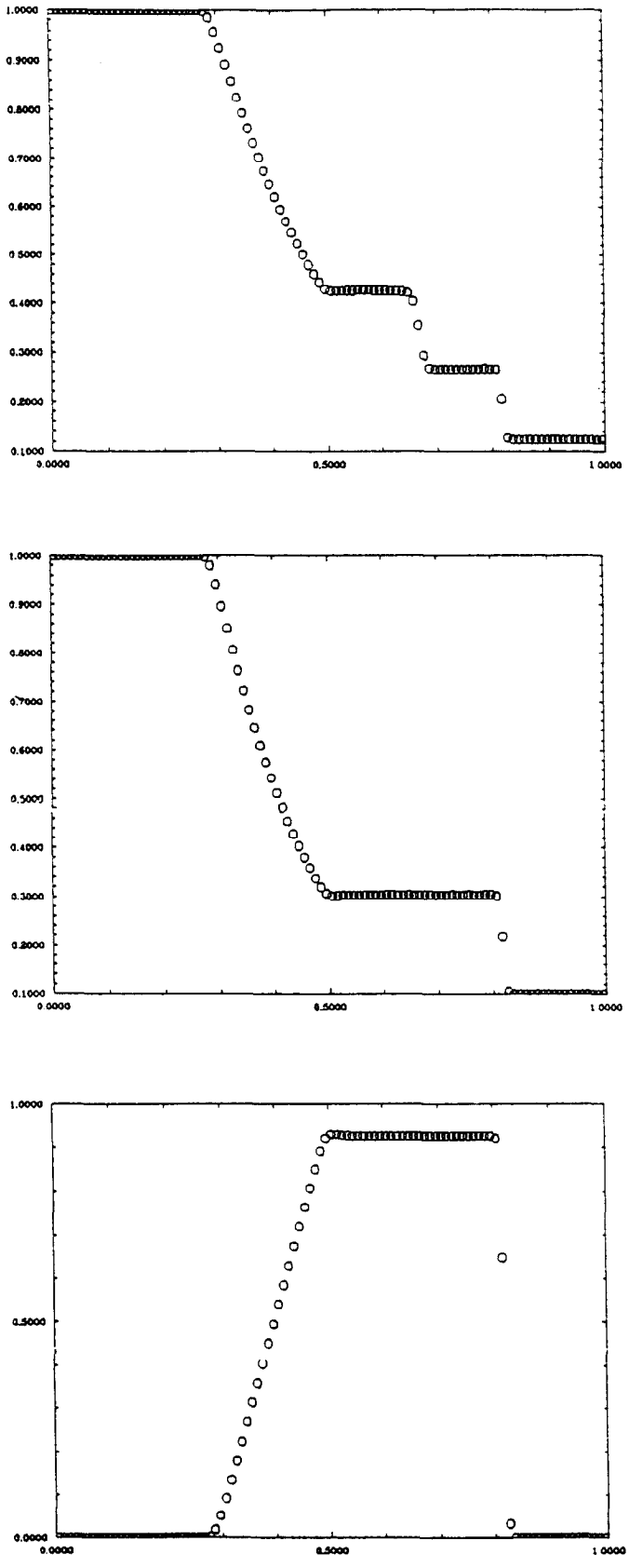


FIG. 3. The density, pressure, and velocity for the Sod problem with $\Delta x=\frac{1}{100}$.

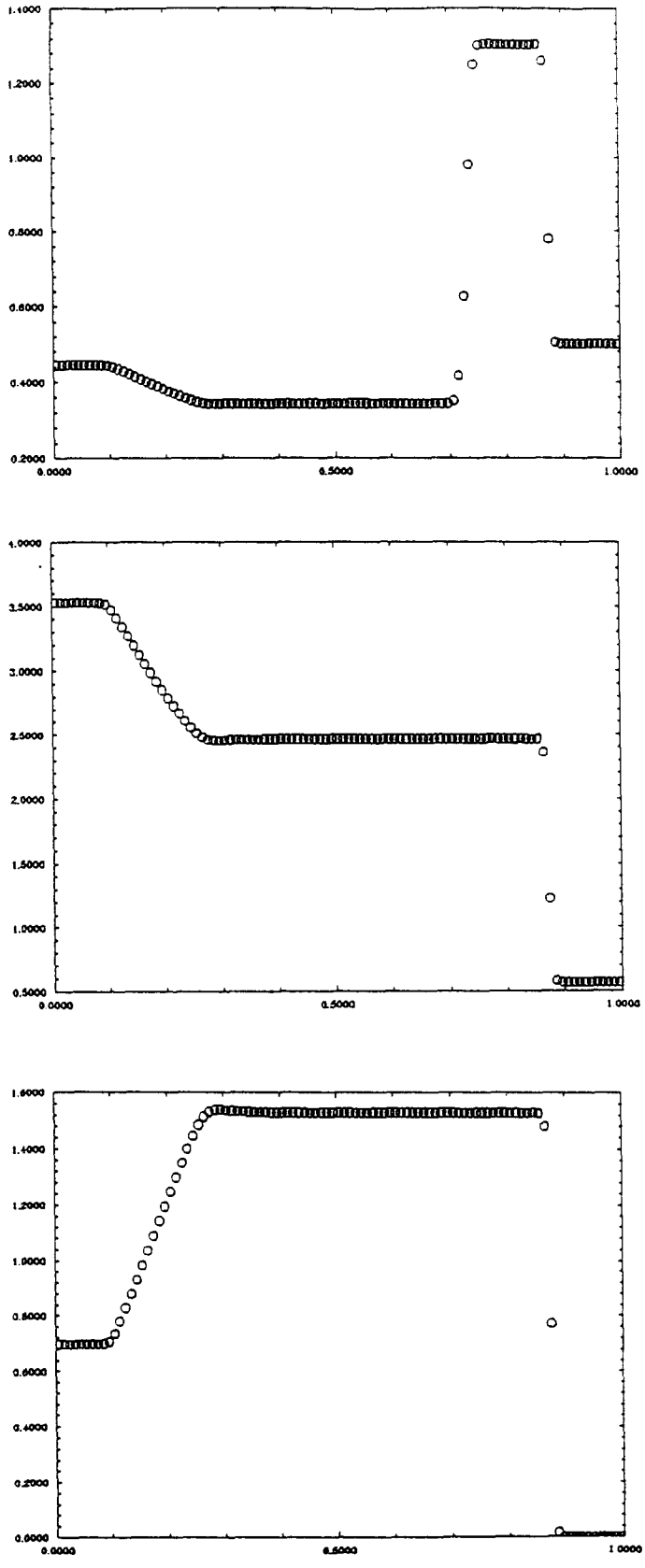


FIG. 4. The density, pressure, and velocity for the Lax problem with $\Delta x = \frac{1}{100}$.

The boundary conditions for boundaries A–F in Fig. 5 are

A: $u(x, y) = u_0(x, y)$

B: reflecting

C: reflecting

D: reflecting

E: outflow

F: reflecting.

Numerical treatment of the reentrant corner G is crucial. The treatment here can easily affect the results more than the choice of interior difference scheme. We use the method outlined in [16]. In Fig. 5, seven cells near corner G are depicted with labels "a" and "b." We reset u in the six cells labelled b based on the value of u in cell a. In each cell b, we reset

$$\rho_b = \rho_a (p_b/p_a)^{1/1.4}$$

and then use this new value to compute

$$\alpha = \frac{(u_a^2 + v_a^2)/2 + 1.4(p_a/\rho_a - p_b/\rho_b)}{(u_b^2 + v_b^2)/2}$$

and reset the vector u in cell b to

$$(\rho_b, \alpha\rho_b u_b, \alpha\rho_b v_b, 2.5p_b + \alpha^2\rho_b(u_b^2 + v_b^2)/2)^t.$$

The procedure outlined above in effect imposes an assumption that the flow is nearly steady in a region about the corner. In rare instances, computed pressures near the corner become slightly negative. When this occurs we reset the pressure to a small positive value and proceed. We have never encountered any degradation in results due to this practice.

Figure 6 depicts the density at $t = 4$ for the step problem on three grid refinements, $\Delta x = \Delta y = \frac{1}{20}$, $\Delta x = \Delta y = 1/40$ and $\Delta x = \Delta y = 1/80$. Thirty equally spaced contours are displayed with extreme contour values given by the extreme values of the approximation.

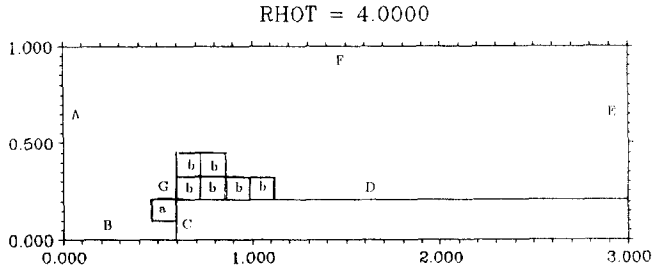


FIG. 5. The step problem domain.

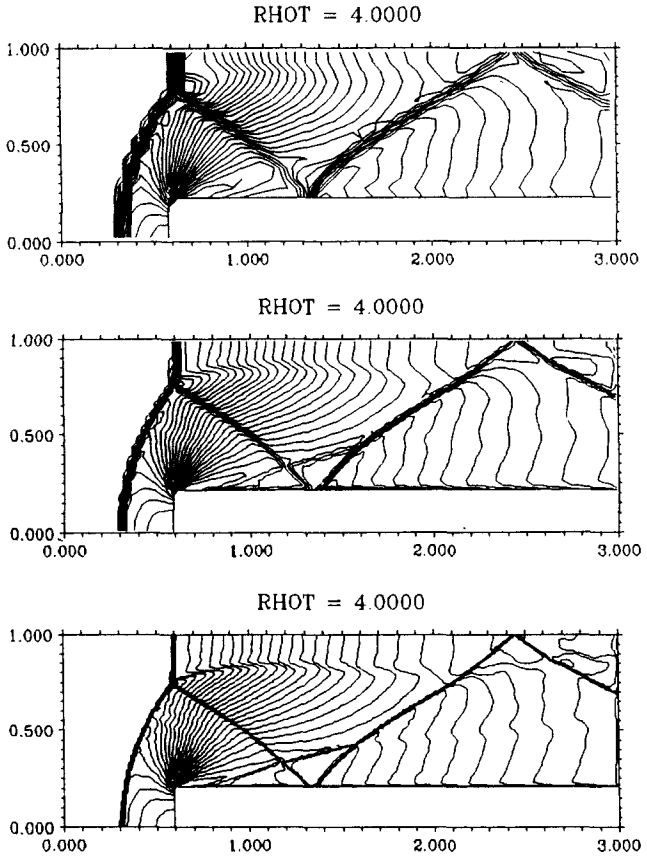


FIG. 6. The step problem density at $t=4$ with $\Delta x=\Delta y=\frac{1}{20}$, $\Delta x=\Delta y=\frac{1}{40}$ and $\Delta x=\Delta y=\frac{1}{80}$.

The ramp problem is solved on the domain $(x, y)\in(0, 4)\times(0, 1)$ with t running from $t=0$ to $t=0.2$. The initial data are

$$u(x, y, 0)=\begin{cases}u_{\mathrm{L}} & \text{for } y\geq h(x, 0)\\ u_{\mathrm{R}} & \text{for } y<h(x, 0),\end{cases}$$

where the state on the left, the state on the right, and the shock height are

$$u_{\mathrm{L}}=(8,\ 57.1597,\ -33.0012,\ 563.544)^{t}$$

$$u_{\mathrm{R}}(1.4,\ 0,\ 0,\ 2.5)^{t}$$

$$h(x, t)=\sqrt{3}(x-\tfrac{1}{6})-20t.$$

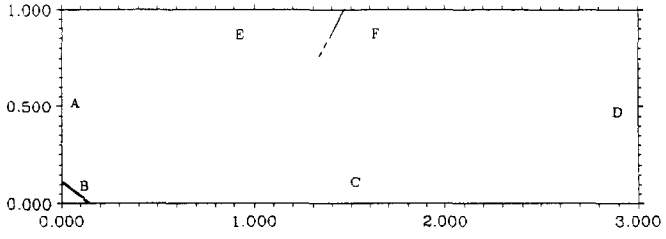


FIG. 7. The ramp problem domain.

Data u_{L} and u_{R} correspond to a Mach 10 planar shock ($\gamma=1.4$) at an angle of 60° with the x axis. The boundary conditions for boundaries A–F in Fig. 7 are time-dependent. (Note that only part of the computational domain is displayed. The actual domain extends in x to $x=4$). The location of the point separating boundaries E and F is

$$x_{EF}=\tfrac{1}{6}+\sqrt{3}(1+20t).$$

The boundary conditions are

A: $u(x, y)=u_{\mathrm{L}}$

B: same as A

C: reflecting

D: $u(x, y)=u_{\mathrm{R}}$

E: same as A

F: same as D.

The point $(\frac{1}{6}, 0)$ corresponds to the corner of the ramp and points $(x, 0)$ with $x>\frac{1}{6}$ lie on the sloping part of the ramp. Note that initial and boundary conditions in cells along the upper boundary which are intersected by $h(x, t)$ use appropriate averages of u_{L} and u_{R}.

Figure 8 depicts the density at $t=0.2$ for the ramp problem on two grid refinements, $\Delta x=\Delta y=\frac{1}{30}$ and $\Delta x=\Delta y=\frac{1}{60}$. Figure 9 depicts the density, pressure, x-velocity, and y-velocity at $t=0.2$ on a grid with $\Delta x=\Delta y=\frac{1}{120}$. This final run required approximately 2200

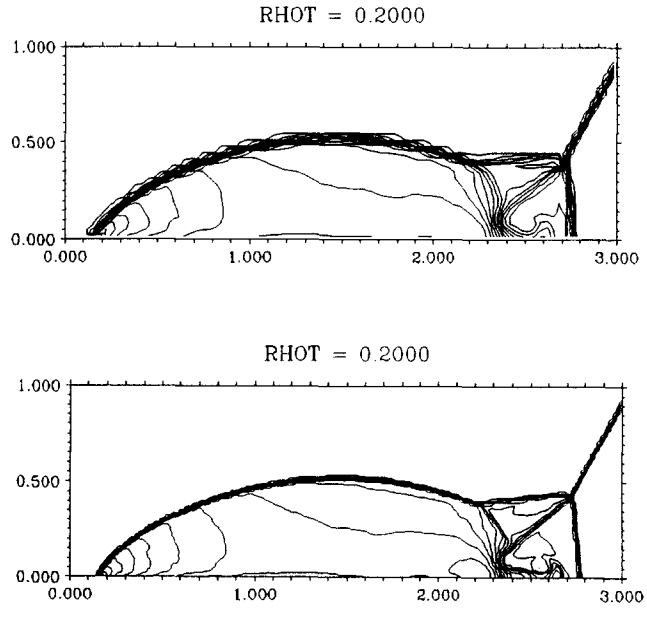


FIG. 8. The ramp problem density at $t=0.2$ with $\Delta x=\Delta y=\frac{1}{30}$ and $\Delta x=\Delta y=\frac{1}{60}$.

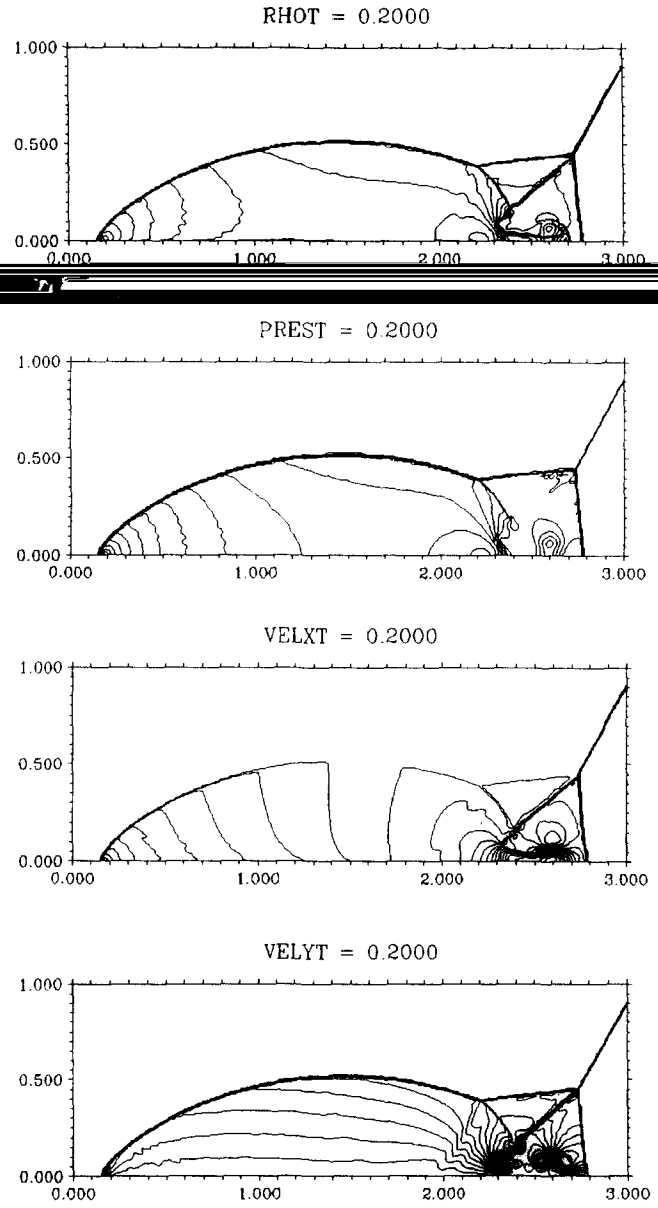


FIG. 9. The ramp problem, density, pressure, x-velocity, and y-velocity at $t = 0.2$ with $\Delta x = \Delta y = \frac{1}{120}$.

Cray CPU seconds. Again, 30 equally spaced contours are displayed with extreme contour values given by the extreme values of the approximation.

APPENDIX: THE JACOBIAN MATRICES OF 2 – d GAS DYNAMICS

The eigenvalue expansion of the 4×4 matrix $A^x(u) = Df(u)$ is

$$A^x(u) = R^x(u)\, \Lambda^x(u)(L^x(u))^t$$
$$= R^x(u)\, \Lambda^x(u)(R^x(u))^{-1},$$

where

$$R^x(u) = \begin{pmatrix} 1 & 1 & 0 & 1 \\ u-c & u & 0 & u+c \\ v & v & \rho & v \\ s_0 - s_7 + s_8 & s_0 & \rho v & s_0 + s_7 + s_8 \end{pmatrix},$$

$$\Lambda^x(u) = \text{diag}(u-c,\ u,\ u,\ u+c)$$

$$(L^x(u))^t = \begin{pmatrix} -s_3 + s_4 & s_1 - s_5 & -s_6 & s_2 \\ 1 - 2s_4 & 2s_5 & 2s_6 & -2s_2 \\ -v/\rho & 0 & 1/\rho & 0 \\ s_3 + s_4 & -s_1 - s_5 & -s_6 & s_2 \end{pmatrix},$$

and

$$(s_0, s_1, s_3, s_4, s_5, s_6, s_7, s_8)$$
$$= ((u^2+v^2)/2,\ -1/(2c),\ 1/(5c^2),$$
$$-u/(2c),\ (u^2+v^2)/(10c^2),$$
$$u/(5c^2),\ v/(5c^2),\ cu,\ 2.5c^2).$$

The eigenvalue expansion of the 4×4 matrix $A^y(u) = Dg(u)$ is

$$A^y(u) = R^y(u)\, \Lambda^y(u)(L^y(u))^t$$
$$= R^y(u)\, \Lambda^y(u)(R^y(u))^{-1},$$

where

$$R^y(u) = \begin{pmatrix} 1 & 1 & 0 & 1 \\ u & u & -\rho & u \\ v-c & v & 0 & v+c \\ t_0 - t_7 + t_s & t_0 & -\rho u & t_0 + t_7 + t_s \end{pmatrix},$$

$$\Lambda^y(u) = \text{diag}(v-c,\ v,\ v,\ v+c)$$

$$(L^y(u))^t = \begin{pmatrix} -t_3 + t_4 & -t_6 & t_1 - t_5 & t_2 \\ 1 - 2t_4 & 2t_6 & 2t_5 & -2t_2 \\ u/\rho & -1/\rho & 0 & 0 \\ t_3 + t_4 & -t_6 & -t_1 - t_5 & t_2 \end{pmatrix},$$

and

$$(t_0, t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8)$$
$$= ((u^2+v^2)/2,\ -1/(2c),\ 1/(5c^2),$$
$$-v/(2c),\ (u^2+v^2)/(10c^2),$$
$$v/(5c^2),\ u/(5c^2),\ cv,\ 2.5c^2).$$

ACKNOWLEDGMENTS

We thank Phil Colella for an explanation of his treatment of the singularity at the reentrant corner of the step problem, to the Cray Research Corporation for providing compute time to run the numerical examples found in this paper, and the management of Exxon Production Research Company for permission to publish this paper.

REFERENCES

1. J. B. Bell and G. R. Shubin, *in* Proceedings, 1985 SPE Reservoir Simulation Symposium, Dallas, February 1985, Paper 13514 (unpublished).
2. Y. Brenier, *SIAM J. Numer. Anal.* **21**, 1013 (1984).
3. P. Colella, *SIAM J. Sci. Statist. Comput.* **6**, 104 (1985).
4. P. Colella and P. R. Woodward, *J. Comput. Phys.* **54**, 174 (1984).
5. R. E. Ewing, T. F. Russell, and M. F. Wheeler, *in* Proceedings, 1983 SPE Reservoir Simulation Symposium, San Francisco, November 1983, Paper 12241 (unpublished).
6. S. K. Godunov, *Mat. Sb.* **47**, 271 (1959).
7. A. Harten, B. Engquist, S. Osher, and S. R. Chakravarthy, *J. Comput. Phys.* **71**, 231 (1987).
8. A. Harten and S. Osher, *SIAM J. Numer. Anal.* **24**, 279 (1987).
9. P. D. Lax, *Commun. Pure Appl. Math.* 7, 159 (1954).
10. S. Osher, *SIAM J. Numer. Anal.* **21**, 217 (1984).
11. P. L. Roe, Lectures in Appliedd Mathematics, Vol. 22 (Springer-Verlag, New York/Berlin, 1985), p. 163.
12. R. Sanders, *Math. Comput.* **40**, 91 (1983).
13. R. Sanders, *Math. Comput.* **51**, 535 (1988).
14. R. Sanders and A. Weiser, *Comput. Methods Appl. Mech. Eng.* **75**, 91 (1989).
15. G. A. Sod, *J. Comput. Phys.* **27**, 1 (1978).
16. P. Woodward and P. Colella, *J. Comput. Phys.* **54**, 115 (1984).