JOURNAL OF COMPUTATIONAL PHYSICS 87, 171–200 (1990)

Multidimensional Upwind Methods for Hyperbolic Conservation Laws*

PHILLIP COLELLA

Mechanical Engineering Department, University of California, Berkeley, California 94720

Received June 20, 1984; revised October 21, 1987

We present a class of second-order conservative finite difference algorithms for solving numerically time-dependent problems for hyperbolic conservation laws in several space variables. These methods are upwind and multidimensional, in that the numerical fluxes are obtained by solving the characteristic form of the full multidimensional equations at the zone edge, and that all fluxes are evaluated and differenced at the same time; in particular, operator splitting is not used. Correct behavior at discontinuities is obtained by the use of solutions to the Riemann problem, and by limiting some of the second-order terms. Numerical results are presented, which show that the methods described here yield the same high resolution as the corresponding operator split methods. © 1990 Academic Press, Inc.

INTRODUCTION

Over the last several years, there has been considerable development of upwind-type numerical methods for solving nonlinear systems of hyperbolic conservation laws in several space dimensions. These methods, generally speaking, are all second-order extensions of Godunov's first-order method [11]. They incorporate into the numerical solutions the nonlinear wave propagation properties of the solution, in the form of Riemann problems and characteristic equations, leading to algorithms which are robust and accurate, even in the presence of nonlinear discontinuities. However, all of the methods currently in use are derived using the characteristic form of the equations in one space dimension, with most of these algorithms being extented to several space dimensions using operator splitting. Nonetheless, these algorithms, particularly the operator split ones, have been quite successful in resolving complex patterns of interacting discontinuities and smooth waves; for further details see [22] and the references cited there.

In this paper, we will consider a class of conservative finite difference algorithms

* Work supported by the Applied Mathematical Sciences subprogram of the Office of Energy Research of the U.S. Department of Energy at the Lawrence Berkeley Laboratory under Contract DE-AC03-76SF00098; by the U.S. Defence Nuclear Agency under DNA task code Y99QAXSG; and by the Office of Naval Research under Contract N00014-76-C-0316.

0021-9991/90 $3.00

581/87/1-12

for hyperbolic conservation laws in several space variables which do not make use of operator splitting, for which the multidimensional wave propagation properties of the solution are used to calculate fluxes. Unsplit schemes are customarily used in a variety of applications, including petroleum reservoir simulation [18], ionospheric physics [24], and Lagrangian hydrodynamics [1]. Thus, one of our goals is to provide algorithms which have the same robustness and resolution as the existing operator split algorithms, but which have the same unsplit structure as the existing algorithms used in the applications codes in those areas. In addition, there are two specific applications for which these methods were developed which are the subject of our current research. One is as a method to be coupled with a front tracking method [3], where the tracked front is represented locally by a polygonal line which divides the cells into two pieces. In each piece, the solution is updated by a method that is necessarily unsplit, in order to preserve the Rankine–Hugoniot relations for the tracked front. The second application is as a starting point for the extension to more than one space dimension of implicit/explicit methods of the type discussed in [10]. In these methods, propagation along each of the characteristic families is treated implicitly or explicitly, depending on whether the CFL number for that characteristic is greater than or less than 1. Thus we require an explicit algorithm with properties similar to those of the 1-dimensional algorithms in [7], but which can be hybridized continuously to an implicit algorithm, in order to have steady states which are independent of Δt.

The design of the algorithm described here is broken into two steps. First, we specify an algorithm for a linear scalar advection equation, which, in smooth regions, is second-order accurate, to which a monotonicity condition, related to those used in [20] for advection algorithms in one dimension, is applied. We then construct the algorithm for systems by introducing a predictor-corrector formalism and by replacing various derivatives in the predictor step by finite differences, using the advection algorithm as guide: upwind differences for advection become differences of Godunov fluxes for systems, and monotonized central differences for advection become monotonized central differences with monotonicity constraints applied to the appropriate choice of transformed variables. Independently of the present work, van Leer also derived multidimensional upwind methods for hyperbolic conservation laws, following a similar line or reasoning; in particular, both methods lead to the algorithm for advection given in the next section. However, his extension to systems is rather different from the predictor-corrector formalism described here; for details, see [21].

A major problem in the program outlined above is the specification of design criteria which guarantee oscillation-free results, even in the one for a linear scalar equation. The principal criterion in one space dimension is that the scheme be total variation diminishing [13]; however, a straightforward generalization of this criterion to more than one dimension has been shown in [12] to imply that the scheme is at most first-order accurate for smooth solutions. The approach taken in the present work is to specify cetain necessary conditions that the scheme must satisfy, and which are satisfied by the schemes described here. These are:

(1) For a 1-dimensional problem aligned with one of the grid directions, the algorithm should reduce to a second-order Godunov method of a type described in [7].

(2) The second-order scheme without limiting, and the first-order scheme obtained by imposing the full limiting of the fluxes at all mesh points, should have as linear difference schemes, the same CFL stability limit on the time step. This CFL stability limit should be the same as for an operator split scheme, with the component 1-dimensional algorithm as in [7].

(3) In the case of linear advection, the fully limited scheme should satisfy a maximum principle.

In the following, we will restrict our attention to the case of two space variables. Although the formalism developed here carries over to higher dimensions, the trade-offs between performance and cost change as the number of dimensions grow; a proper evaluation of what those trade-offs are can only be made by numerical experimentation. In three dimensions, such a study would strain the capabilities of present computer technology. Some discussion of these considerations will be made in the final section of this paper.

1. Advection Algorithms

We consider the scalar advection equation in two space variables

$$\frac{\partial \rho}{\partial t} + \mathbf{u} \cdot \nabla \rho = 0$$

$$\mathbf{x} = (x, y), \qquad \rho = \rho(\mathbf{x}, t) \qquad \nabla = \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}\right) \qquad \mathbf{u} = (u, v) \qquad u, v > 0. \tag{1.1}$$

We want to solve numerically initial value problems for (1.1). To this end, we will attempt to construct algorithms which generalize upstream-centered algorithms in [20] to two space variables, without replacing the operator approximating the time evolution of (1.1) by the product of 1-dimensional evolution operators. Our strategy will be to start from a well-behaved first-order upwind algorithm for solving (1.1). We add to the evolution operator the terms necessary to make the algorithm second-order accurate in a way such that they can be limited, i.e., subtracted off, at discontinuities.

Let Δx, Δy be spatial increments, Δt a time increment. We assume that we know ρ_{ij}^n, the average of ρ at time t^n:

$$\rho_{i,j}^n = \frac{1}{\sigma_{i,j}} \int_{\Delta_{i,j}} \rho(\mathbf{x}, t^n)\, d\mathbf{x}.$$

Here $\Delta_{i,j} = [(i-\frac{1}{2})\Delta x, (i+\frac{1}{2})\Delta x] \times [(j-\frac{1}{2})\Delta y, (j+\frac{1}{2})\Delta y]$, $\sigma_{i,j} = (\text{area of } \Delta_{i,j})$.

We wish to calculate $\rho_{i,j}^{n+1}$, the solution to (1.1) at time $t^{n+1} = t^n + \Delta t$. A natural algorithm for doing this is to trace backward in time from $t^n + \Delta t$ the set $\Delta_{i,j}$, along the characteristics of (1.1), to obtain $\Delta'_{i,j}$. Then $\rho_{i,j}^{n+1}$ is set equal to the average over $\Delta'_{i,j}$ of the trivial interpolation function $\rho^I(\mathbf{x}) = \rho_{i,j}^n$ if $\mathbf{x} \in \Delta_{i,j}$:

$$\rho_{i,j}^{n+1} = \frac{1}{\sigma'_{i,j}} \int_{\Delta'_{i,j}} \rho^I(x, y)\, dx\, dy$$

$$= (A_1 \rho_{i,j}^n + A_2 \rho_{i,j-1}^n + A_3 \rho_{i-1,j}^n + A_4 \rho_{i-1,j-1}^n) \frac{1}{\sigma'_{i,j}} \tag{1.2}$$

where the A_k's are the areas in each of the four upstream zones swept out by $\mathbf{u}$, as indicated in Fig. 1.

We can put this scheme in explicit conservation form

$$\rho_{i,j}^{n+1} = \rho_{i,j}^n + \frac{u\,\Delta t}{\Delta x}(\rho_{i-1/2,j}^{n+1/2} - \rho_{i+1/2,j}^{n+1/2}) + \frac{v\,\Delta t}{\Delta y}(\rho_{i,j-1/2}^{n+1/2} - \rho_{i,j+1/2}^{n+1/2}) \tag{1.3}$$

$$\begin{aligned} \rho_{i,j+1/2}^{n+1/2} &= \rho_{i,j}^n + \frac{u\,\Delta t}{2\,\Delta x}(\rho_{i-1,j}^n - \rho_{i,j}^n) \\ \rho_{i+1/2,j}^{n+1/2} &= \rho_{i,j}^n + \frac{v\,\Delta t}{2\,\Delta y}(\rho_{i,j-1}^n - \rho_{i,j}^n). \end{aligned} \tag{1.4}$$

One way of deriving the formulas for $\rho_{i+1/2,j}^{n+1/2}$, $\rho_{i,j+1/2}^{n+1/2}$ is to notice that they are the averages of P^I over the region swept out by the characteristics through the zone edges centered, respectively, at $(i+\frac{1}{2}, j)$ and $(i, j+\frac{1}{2})$ (Fig. 2). We shall refer to this scheme as the corner transport upwind (CTU) scheme, since it takes into account the effect of information propagating across corners of zones in calculating the flux. This scheme is first-order accurate. It also satisfies a maximum principle, since $\rho_{i+1/2,j}^{n+1/2}$, $\rho_{i,j+1/2}^{n+1/2}$ are weighted sums, with nonnegative weights, of values of the solution at time t^n.

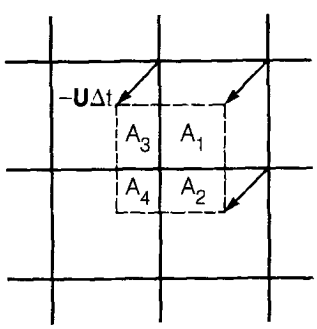

FIG. 1. The region over which we average ρ^I to obtain the new value for ρ is outlined with a dotted line. It is obtained by following the integral curves of the vector field $\mathbf{u}$ (in this case, straight lines) backwards in time by Δt from points in Δ_{ij}.

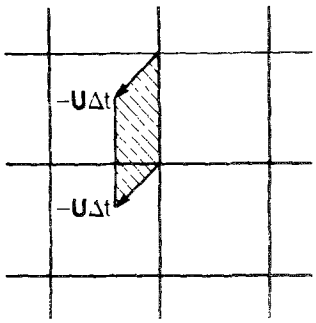

FIG. 2. The shaded region is the region over which one averages ρ^l to obtain the CTU flux at the zone edge bounding that region. It is the set of all points from which characteristics can reach that zone edge between time t^n and $t^n + \Delta t$.

One fact that is immediately seen from the formula given above for the floxes is the difference between the CTU scheme and the conventional donor cell differencing. In the latter case, $\rho^{n+1/2}_{i-1/2,j} = \rho^n_{i,j}$, $\rho^{n+1/2}_{i,j-1/2} = \rho^n_{i,j}$. Thus, in this scheme, we are adding a time-centered correction term to the donor-cell flux which estimates the effect on the flux of the gradients in the transverse direction. This corresponds to subtracting from the donor cell algorithm a term which, to leading order in the truncation error, is always destabilizing. This is reflected in the differing CFL time step limits for the two schemes:

$$\text{CTU: } \max\left(\frac{u\,\Delta t}{\Delta x}, \frac{v\,\Delta t}{\Delta y}\right) \leqslant 1. \tag{1.5}$$

$$\text{Donor-cell: } \quad \frac{u\,\Delta t}{\Delta x} + \frac{v\,\Delta t}{\Delta y} \leqslant 1, \tag{1.6}$$

where (1.5) is a sufficient condition, and (1.6) is a necessary condition, as is easily checked using Fourier analysis.

One can view schemes of the form (1.3)–(1.4) as being predictor-corrector schemes. One regards the calculation of $\rho^{n+1/2}_{i+1/2,j}$, $\rho^{n+1/2}_{i,j+1/2}$ as the predictor step, with the conservative differencing as the corrector step. Thus, if $\rho^{n+1/2}_{i+1/2,j}$ were to be calculated in such a way as to have a local truncation error of $O(\Delta t^2)$ in smooth regions, then the scheme would be second-order accurate. To obtain such an estimate for $\rho^{n+1/2}_{i+1/2,j}$ one must have

$$\begin{aligned}
\rho^{n+1/2}_{i+1/2,j} &= \rho^n_{i,j} + \frac{\Delta t}{2}\frac{\partial p}{\partial t} + \frac{\Delta x}{2}\frac{\partial \rho}{\partial x} \\
&= \rho^n_{i,j} - \frac{\Delta t}{2}\left(u\frac{\partial \rho}{\partial x} + v\frac{\partial \rho}{\partial y}\right) + \frac{\Delta x}{2}\frac{\partial \rho}{\partial x} \\
&= \rho^n_{i,j} + \left(\frac{\Delta x}{2} - \frac{u\,\Delta t}{2}\right)\frac{\partial \rho}{\partial x} - \frac{v\,\Delta t}{2}\frac{\partial \rho}{\partial y}.
\end{aligned} \tag{1.7}$$

The only terms in (1.7) missing for the CTU flux (1.4) are the ones involving $\partial\rho/\partial x$. Thus, we add that term to $\rho^{n+1/2}_{i+1/2,j}$ to obtain a second-order flux:

$$\rho^{n+1/2}_{i+1/2,j} = \rho^n_{i,j} + \left(\frac{\Delta x}{2} - u\frac{\Delta t}{2}\right)\frac{\Delta^x \rho_{i,j}}{\Delta x} - \frac{v\,\Delta t}{2\,\Delta y}(\rho^n_{i,j} - \rho^n_{i,j-1}). \tag{1.8}$$

Here $\Delta^x \rho_{i,j}/\Delta x$ should be a difference approximation to $(\partial\rho/\partial x)|_{(i\Delta x, j\Delta y)}$, and $\Delta^x\rho$ should also be limited to suppress oscillations at discontinuities. The simplest choice is a central difference approximation to $(\partial\rho/\partial x)$, with the 1-dimensional limiter given in [20]:

$$\begin{aligned}(\Delta^x\rho)_{i,j} &= \min(\tfrac{1}{2}|\rho^n_{i+1,j} - \rho^n_{i-1,j}|, 2|\rho^n_{i+1,j} - \rho^n_{i,j}|, 2|\rho^n_{i,j} - \rho^n_{i-1,j}|) \\ &\quad \times \operatorname{sgn}(\rho^n_{i+1,j} - \rho^n_{i-1,j}) \qquad \text{if} \quad (\rho^n_{i+1,j} - \rho^n_{i,j})(\rho^n_{i,j} - \rho^n_{i-1,j}) > 0; \\ &= 0 \qquad \text{otherwise.}\end{aligned} \tag{1.9}$$

Similarly, we define

$$\rho^{n+1/2}_{i,j+1/2} = \rho^n_{i,j} + \left(\frac{\Delta y}{2} - \frac{\Delta t}{2}v\right)\frac{(\Delta^y\rho)_{i,j}}{\Delta y} - \frac{\Delta t}{2\,\Delta x}u(\rho^n_{i,j} - \rho^n_{i-1,j}),$$

where $\Delta^y\rho$ is a monotonized central difference formula, such as the one given by (1.9) with the roles of i and j reversed. Because of the nonlinear switch in the definition of $\Delta^x\rho$, $\Delta^y\rho$, one cannot perform a formal error analysis on this algorithm. However, in smooth regions, one expects $\Delta^x\rho$, $\Delta^y\rho$ to be given by the central difference operators $(\Delta^x\rho)_{i,j} = \frac{1}{2}(\rho_{i+1,j} - \rho_{i-1,j})$, $(\Delta^y\rho)_{i,j} = \frac{1}{2}(\rho_{i,j+1} - \rho_{i,j-1})$. In this case, one can perform the linear error analysis and find that the scheme is second-order accurate. We have also calculated the amplification factor and evaluated it numerically; we have found that, as long as the time step satisfies (1.5), the second-order algorithm does not amplify any Fourier modes.

There is not a great deal one can say about the monotonicity properties of this algorithm, save that, when the slopes are fully limited, i.e., $\Delta^y\rho = \Delta^x\rho = 0$, it reduces to the first-order CTU scheme described above. In order to have this property, it is necessary to treat the spatial derivatives in the predictor step in a non-symmetric way: the derivatives in the direction tangent to the zone edge are approximated by upwind differences, and are not subject to monotonicity constraints, while the derivatives in the direction normal to the zone edge are approximated by monotonized central differences. For linear advection of a discontinuity oblique to the grid, the algorithm appears to produce monotone results.

A different approach to the one taken here, more in line with the geometric constructions in [20], would be to construct piecewise linear interpolants of ρ, suitably monotonized, and to integrate over surfaces swept out by the characteritics to obtain fluxes, similar to what was done to obtain the flux form (1.4) for the CTU scheme. We have not done so here: for a development along such lines, see [21]. However, for strongly nonlinear problems, we find that a somewhat more elaborate

treatment of the transverse derivatives than simply using first-order upwind differencing will be required, leading to an algorithm which is intermediate in complexity. This algorithm will be discussed in the next section.

2. Systems of Conservation Laws

In this section, we will consider algorithms for solving numerically the initial value problem

$$\begin{gathered}\frac{\partial U}{\partial t}+\nabla\cdot\mathbf{F}=0\\ U(\mathbf{x},t)=U:\mathbf{R}^2\times[0,T]\to\mathbf{R}^M\\ \mathbf{F}=(F^x,F^y)\in\mathbf{R}^M\times\mathbf{R}^M\\ U(\mathbf{x},0)=U_0(\mathbf{x}).\end{gathered}\tag{2.1}$$

For each $\mathbf{n}\in\mathbf{R}^2$ we define the projected equations (along $\mathbf{n}$) to be the 1-dimensional system of conservation laws

$$\frac{\partial U}{\partial t}+\frac{\partial F^{\mathbf{n}}}{\partial \chi}=0\qquad F^{\mathbf{n}}(U)=\mathbf{n}\cdot\mathbf{F}(U).\tag{2.2}$$

We say that the system (2.1) is hyperbolic if, for every $\mathbf{n}$ the projected equations

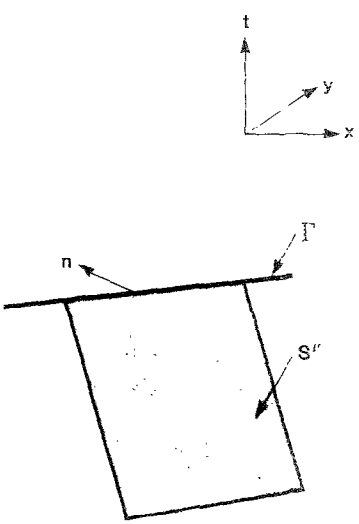

FIG. 3. Characteristic surfaces in two space dimensions. Γ is a curve in the spatial plane with normal vector field $\mathbf{n}$, and S^ν is one of the M characteristic surfaces in space-time passing through Γ.

(2.2) are hyperbolic, i.e., that the linearized coefficient matrix $\nabla_U F^{\mathbf{n}} = A^{\mathbf{n}}$ has M real eigenvalues $\lambda^{\mathbf{n},1} \leqslant \cdots \leqslant \lambda^{\mathbf{n},M}$ corresponding to M linearly independent left and right eigenvectors $(l^{\mathbf{n},\nu}, r^{\mathbf{n},\nu})$, $\nu = 1, \ldots, M$. We also have $A^{\mathbf{n}} = \mathbf{n} \cdot \mathbf{A}$, where $\mathbf{A} = (A^x, A^y)$, $A^x = \nabla_U F^x$, $A^y = \nabla_U F^y$. The left and right eigenvectors can be chosen so as to be biorthonormal, i.e., $l^{\mathbf{n},\nu} \cdot r^{\mathbf{n},\nu'} = \delta_{\nu,\nu'}$, so that the expansion of a vector $w \in \mathbf{R}^M$ in terms of the $r^{\mathbf{n},\nu}$'s is given by $w = \sum_{\nu=1,\ldots,M} \alpha^{\mathbf{n},\nu} r^{\mathbf{n},\nu}$, with $\alpha^{\mathbf{n},\nu} = l^{\mathbf{n},\nu} \cdot w$.

Our algorithm for the calculation of conservative fluxes is motivated in part by a version of the multidimensional theory of characteristics, which we review briefly here; for a more extensive discussion, see [8, 16]. If Γ is a curve in the plane $\{(\mathbf{x}, t): t = t_0\}$, then there exist surfaces $S^1, \ldots, S^M$ called characteristic surfaces, passing through Γ, such that the normal to S^ν at a point $(\mathbf{x}, t)$ is of the form $(\mathbf{n}, -\lambda^{\mathbf{n},\nu})$, where $\lambda^{\mathbf{n},\nu}$ is the νth eigenvalue of the projected equations in the direction of the unit vector $\mathbf{n}$ (see Fig. 3). The significance of these surfaces is that along each of these surfaces, a continuous, piecewise C^1 solution to (2.1) satisfies the following interior partial differential relation:

$$
\begin{aligned}
0 &= l^{\mathbf{n},\nu} \cdot \left(\frac{\partial U}{\partial t} + \mathbf{A} \cdot \nabla U \right) \\
&= l^{\mathbf{n},\nu} \cdot \left(\frac{\partial U}{\partial t} + (\mathbf{n} \cdot \mathbf{A})(\mathbf{n} \cdot \nabla U) + (\mathbf{t} \cdot \mathbf{A})(\mathbf{t} \cdot \nabla U) \right) \\
&= l^{\mathbf{n},\nu} \cdot \left(\frac{\partial U}{\partial t} + \lambda^{\mathbf{n},\nu} \mathbf{n} \cdot \nabla U + (\mathbf{t} \cdot \mathbf{A})(\mathbf{t} \cdot \nabla U) \right),
\end{aligned} \tag{2.3}
$$

where $\mathbf{t}$ is a unit vector orthogonal to $\mathbf{n}$ in the plane. Since $(\lambda^{\mathbf{n},\nu}\mathbf{n}, 1)$ and $(\mathbf{t}, 0)$ are tangent to S^ν, then (2.3) contains only derivatives in directions tangent to S^ν. In particular, if we define $d/d\sigma^\nu$ to be differentiation in the direction of the vector field $(\lambda^{\mathbf{n},\nu}\mathbf{n}, 1)$, then (2.3) becomes

$$
l^{\mathbf{n},\nu} \cdot \frac{dU}{d\sigma^\nu} + (l^{\mathbf{n},\nu} \cdot A^{\mathbf{t}})(\mathbf{t} \cdot \nabla U) = 0; \tag{2.4}
$$

i.e., we obtain the ordinary differential relation from the theory of characteristics in one dimension for the system projected in the $\mathbf{n}$ direction, with the derivatives in the $\mathbf{t}$ direction acting as source terms.

Finally, we assume that the Riemann problem for the projected system (2.2) is well posed for all $\mathbf{n} \in \mathbf{R}^2$, i.e., that the initial value problem for (2.2) given by

$$
\begin{aligned}
U(\chi, 0) &= U_L \qquad \text{for} \quad \chi < 0 \\
&= U_R \qquad \text{for} \quad \chi > 0
\end{aligned}
$$

has a unique solution with appropriate entropy conditions, for any choice of U_L, U_R for which (2.2) is hyperbolic. This solution is a function only of the similarity

variable χ/t; throughout this paper, when we require the solution to a Riemann problem, it will be at the point $\chi/t = 0$.

We assume, as in the scalar case, that we know $U^n_{i,j}$, the average of the solution over $\Delta_{i,j}$, the zone centered at $(i\,\Delta x, j\,\Delta y)$:

$$U^n_{i,j} = \frac{1}{\sigma_{i,j}} \int_{\Delta_{i,j}} U(\mathbf{x}, t^n)\, d\mathbf{x}.$$

We want to extend the algorithm described in the previous section to calculate $U^{n+1}_{i,j}$. The difficulty here is that the different modes of wave propagation can carry gradient information from different sides of the zone edge where the flux is to be evaluated. We solve this problem by using predictor calculations similar to (1.8) to calculate two states at a zone edge, representing the propagation of signals coming from the left and the right of the zone edge. We then obtain a single value for the flux by solving a Riemann problem given the two states, with the jump assumed to be parallel to the zone edge.

The algorithm can be broken up into the following four steps:

(1) the calculation of monotonized central difference approximations to

$$\frac{\Delta^x U}{\Delta x} \approx \left.\frac{\partial U}{\partial x}\right|_{(i\Delta x, j\Delta y)}, \qquad \frac{\Delta^y U}{\Delta y} \approx \left.\frac{\partial U}{\partial y}\right|_{(i\Delta x, j\Delta y)};$$

(2) the construction of time-centered left and right states at the zone edges: $U^{n+1/2}_{i+1/2,j,L}$, $U^{n+1/2}_{i+1/2,j,R}$ at $((i+\frac{1}{2})\,\Delta x, j\,\Delta y)$, and $U^{n+1/2}_{i,j+1/2,L}$, $U^{n+1/2}_{i,j+1/2,R}$ at $(i\,\Delta x, (j+\frac{1}{2})\,\Delta y)$;

(3) the solution of the Riemann problem at the zone edges for the projected equations along the normal to that zone edge, given the left and right states computed in (2), to obtain $U^{n+1/2}_{i+1/2,j}$, $U^{n+1/2}_{i,j+1/2}$;

(4) the conservative differencing of the fluxes $F^x_{i+1/2,j} = F^x(U^{n+1/2}_{i+1/2,j})$, $F^y_{i,j+1/2} = F^y(U^{n+1/2}_{i,j+1/2})$ to obtain $U^{n+1}_{i,j}$:

$$U^{n+1}_{i,j} = U^n_{i,j} + \frac{\Delta t}{\Delta x}(F^x_{i-1/2,j} - F^x_{i+1/2,j}) + \frac{\Delta t}{\Delta y}(F^y_{i,j-1/2} - F^y_{i,j+1/2}).$$

We will describe the details of only the calculation of $F^x_{i+1/2,j}$; the other fluxes are calculated along the same lines, interchanging the roles of i and j, x and y.

The calculation of slopes follows the pattern seen in the scalar case: we use central difference to approximate the spatial derivatives of U and constrain them using a 1-dimensional monotonicity algorithm. In imposing monotonicity constraints, there are two strategies which have been used successfully in one dimension. The first is to perform a nonlinear change of variables such that the new dependent variables are the Riemann invariants, i.e., a set of variables $(v^1, \ldots, v^M)^T$ such that $l^\nu \cdot \nabla_U v^\mu = \delta_{\nu\mu}$, and interpolate those variables componentwise using monotonized

interpolation such as the one given for the scalar case in the previous section. This procedure can be done only for special systems, since such a set of Riemann invariants does not, in general, exist when $M > 2$. A variation on this procedure is done for Euler's equations for compressible flow, where the primitive variables are interpolated; this is discussed in Section 4. The second approach, due to Harten [14], is to expand the central difference approximation to the spatial derivatives in terms of the right eigenvectors of the coefficient matrix of the linearized equation and constrain the amplitudes in that expansion. Since the latter procedure is well defined for general systems of conservation laws, we will describe it here.

To calculate $(\Delta^x U)_{i,j}$ we define the expansions,

$$\begin{aligned} \tfrac{1}{2}(U_{i+1,j} - U_{i-1,j}) &= \sum \alpha_C^\nu r^{x,\nu}, \\ 2(U_{i+1,j} - U_{i,j}) &= \sum \alpha_R^\nu r^{x,\nu}, \\ 2(U_{i,j} - U_{i-1,j}) &= \sum \alpha_L^\nu r^{x,\nu}, \end{aligned} \tag{2.5}$$

where $l^{x,\nu}$, $r^{x,\nu}$, $\lambda^{x,\nu}$ are the eigenvectors and eigenvalues of the equations projected in the x coordinate direction. Then $(\Delta^x U)_{i,j}$ is given by

$$(\Delta^x U)_{i,j} = \sum \alpha^\nu r^{x,\nu} \tag{2.6}$$

$$\begin{aligned} \alpha^\nu &= \min(|\alpha_C^\nu|, |\alpha_L^\nu|, |\alpha_R^\nu|) \times \operatorname{sgn}(\alpha_C^\nu) \qquad \text{if} \quad \alpha_L^\nu \alpha_R^\nu > 0 \\ &= 0 \qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{otherwise.} \end{aligned}$$

Next, we define the left and right states at the zone edges $U^{n+1/2}_{i+1/2,j,L}$, $U^{n+1/2}_{i+1/2,j,R}$. We extrapolate from the zone centers on either side of the zone edge at $((i+\frac{1}{2})\Delta x, j\Delta y)$, using a formula similar to (1.7):

$$\begin{aligned} U^{n+1/2}_{i+1/2,j,S} &= U^n_{i+k,j} \pm \frac{\Delta x}{2}\frac{\partial U}{\partial x} + \frac{\Delta t}{2}\frac{\partial U}{\partial t} \\ &= U^n_{i+k,j} \pm \frac{\Delta x}{2}\frac{\partial U}{\partial x} - \frac{\Delta t}{2}\left(\frac{\partial F^x}{\partial x} + \frac{\partial F^y}{\partial y}\right) \\ &= U^n_{i+k,j} + \left(\pm\frac{\Delta x}{2} - \frac{\Delta t A^x}{2}\right)\frac{\partial U}{\partial x} - \frac{\Delta t}{2}\frac{\partial F^y}{\partial y}. \end{aligned} \tag{2.7}$$

Here, and in what follows, we use expressions such as (2.7) involving the symbols $(S, \pm, k)$ to mean a pair of expressions: one with $(S, \pm, k)$ replaced by $(L, +, 0)$, the other with $(S, \pm, k)$ replaced by $(R, -, 1)$. In calculating $U^{n+1/2}_{i+1/2,j,S}$, we approximate $\partial U/\partial x$ by the monotonized central differences $\Delta^x U/\Delta x$ and the $\partial F^y/\partial y$ term by a difference of Godunov fluxes, the extension to nonlinear systems in one dimension of upwind differencing for linear scalar equations.

It is convenient to view the calculation of $U^{n+1/2}_{i+1/2,j,L}$, $U^{n+1/2}_{i+1/2,j,R}$ as consisting of two steps, the first involving the monotonized central difference approximations to $\partial U/\partial x$, the second involving the transverse derivatives:

$$\hat{U}_{i+1/2,j,S} = U^n_{i+k,j} + \left(\pm\frac{\Delta x}{2} - \frac{\Delta t}{2}A^x\right)\frac{\partial U}{\partial x} \tag{2.8}$$

$$U^{n+1/2}_{i+1/2,j,S} = \hat{U}_{i+1/2,j,S} - \frac{\Delta t}{2}\frac{\partial F^y}{\partial y}. \tag{2.9}$$

In order to calculate $\hat{U}_{i+1/2,j,S}$ for linear problems, it would suffice simply to replace $\partial U/\partial x$ by $(\Delta^x U)_{i,j}/\Delta x$. However, we make two changes in (2.8), which, for constant coefficient problems, are redundant operations leading to identical values for $U^{n+1/2}_{i+1/2,j}$, but which have been seen to lead to a somewhat more robust algorithm for strongly nonlinear problems. This first is to discard in the $\partial U/\partial x$ term the components corresponding to characteristics which do not propagate towards the zone edge. The second is to introduce arbitrary reference states $\tilde{U}_L$, $\tilde{U}_R$, taking advantage of the fact that the characteristic projection operators appearing in both the construction of the left and right states and in the solution of the Riemann problem act on increments of U. The resulting algorithm is given as follows:

$$\hat{U}_{i+1/2,j,S} = \tilde{U}_S + P_S(U^n_{i+k,j} - \tilde{U}_S) + P_S\left(\pm\frac{1}{2} - \frac{\Delta t}{2\Delta x}A^x(U_{i+k,j})\right)(\Delta^x U)_{i+k,j}$$
$$P_S w = \sum_{\mp\lambda^{x,v}(U_{i+k,j})>0} (l^{x,v}_{i+k,j}\cdot w)\, r^{x,v}_{i+k,j}. \tag{2.10}$$

The reference states $\tilde{U}_L$, $\tilde{U}_R$ are chosen so as to reduce to as great an extent as possible the size of the sum of the terms multiplied by the characteristic projection operators P_S. One possibility is to take

$$\tilde{U}_L = U^n_{i,j} + \left(\frac{1}{2} - \max(\lambda^{x,M}(U_{i,j}), 0)\frac{\Delta t}{2\Delta x}\right)\Delta^x U_{i,j}$$
$$\tilde{U}_R = U^n_{i+1,j} - \left(\frac{1}{2} + \min(\lambda^{x,1}(U_{i+1,j}), 0)\frac{\Delta t}{2\Delta x}\right)\Delta^x U_{i+1,j}. \tag{2.11}$$

The additional cost of applying the characteristic projection operators is small. Because of the monotonicity algorithm, we already know the expansion of $\Delta^x U$ in terms of the right eigenvectors. Applying the characteristic projection operators to $(\Delta^x U)$ is accomplished by setting to zero the coefficients of the eigenvector expansion of $(\Delta^x U)$ which have associated propagation speeds with the wrong sign. Finally, the calculation of the terms involving A^x is easily accomplished using the fact that the projection operators are sums of eigenprojections of A^x, implying that $P_S A^x \Delta^x U = \sum_{\pm\lambda^{x,v}>0} \lambda^{x,v}\alpha^v r^{x,v}$. Using this fact, and with the above choice of $\tilde{U}_L$, $\tilde{U}_R$, we obtain the following explicit expression for (2.10):

$$\hat{U}_{i+1/2,j,L} = \tilde{U}_L + \frac{\Delta t}{2\Delta x} \sum_{v:\lambda^{x,v}_{i,j}>0} (\lambda^{x,M}_{i,j} - \lambda^{x,v}_{i,j})\, \alpha^{x,v}_{i,j} r^{x,v}_{i,j}$$
$$\hat{U}_{i+1/2,j,R} = \tilde{U}_R + \frac{\Delta t}{2\Delta x} \sum_{v:\lambda^{x,v}_{i+1,j}<0} (\lambda^{x,1}_{i+1,j} - \lambda^{x,v}_{i+1,j})\, \alpha^{x,v}_{i+1,j} r^{x,v}_{i+1,j}, \tag{2.12}$$

where the $\alpha^{x,v}_{i,j}$'s are the expansion coefficients of $(\Delta^x U)_{i,j}$ given by (2.6). This procedure is essentially that given in [7] for computing the left and right states for the 1-dimensional algorithm, applied to the case of piecewise linear interpolation.

To complete the calculation of $U^{n+1/2}_{i+1/2,j,S}$ we approximate $(\partial F^y/\partial y)|_{(i\Delta x, j\Delta y)}$ by some appropriate upwind flux difference. The simplest choice is to use Godunov's first-order method to evaluate $\partial F^y/\partial y$. If we define $U^T_{i,j+1/2}$ to be the solution to the Riemann problem for the projected equations along the y-direction, with left and right states

$$(U^{\mathrm{T}}_{i,j+1/2,L},\, U^{\mathrm{T}}_{i,j+1/2,R}) = (U^n_{i,j},\, U^n_{i,j+1}) \tag{2.13}$$

then

$$U^{n+1/2}_{i+1/2,j,S} = \hat{U}_{i+1/2,j,S} - \frac{\Delta t}{2\Delta y}(F^y(U^{\mathrm{T}}_{i+k,j+1/2}) - F^y(U^{\mathrm{T}}_{i+k,j-1/2})) \tag{2.14}$$

is a sufficiently accurate approximation to (2.9) to yield an algorithm that is second-order accurate. For problems involving moderately strong nonlinear discontinuities which are oblique to the mesh directions, it is necessary to use a slightly more complicated algorithm to evaluate the effect of the transverse derivative term $(\partial F^y/\partial y)(\Delta t/2)$ on the left and right states. This term estimates the change in the solution due to the y-gradients. In the case of an oblique discontinuity, if the estimate is sufficiently different from the actual change calculated in the conservation step, the solution will overshoot, or the discontinuity will spread, depending on the relative signs of the gradient and the error. To alleviate this problem, we use an estimate for $\partial F^y/\partial y$ which is closer to what we will actually use in the conservation step, by taking $U^{\mathrm{T}}_{i,j+1/2}$ to be the solution to the Riemann problem for the equations projected along the y-direction with left and right states

$$(U^{\mathrm{T}}_{i,j+1/2,L},\, U^{\mathrm{T}}_{i,j+1/2,R}) = (\hat{U}_{i,j+1/2,L},\, \hat{U}_{i,j+1/2,R}), \tag{2.15}$$

where $\hat{U}_{i,j+1/2,L}$, $\hat{U}_{i,j+1/2,R}$ is computed using the analogue of (2.10) for the zone edge at $(i\,\Delta x, (j+\frac{1}{2})\,\Delta y)$.

Given the left and right states defined as above, we solve the Riemann problem for the 1-dimensional equation projected along the x direction to obtain $U^{n+1/2}_{i+1/2,j}$. In the case of constant coefficient equations, it is easy to check that $U^{n+1/2}_{i+1/2,j}$ satisfies the following linear equations, independent of the choice of $\tilde{U}_L$, $\tilde{U}_R$:

$$l^{x,v} \cdot (U^{n+1/2}_{i+1/2,j} - U_{i+1/2,j,v}) - \frac{\Delta t}{2\Delta y} l^{x,v} \cdot (F^y(U^{\mathrm{T}}_{i+k,j+1/2}) - F^y(U^{\mathrm{T}}_{i+k,j-1/2})) = 0, \tag{2.16}$$

where

$$U_{i+1/2,j,v} = U^n_{i,j} + \left(\frac{1}{2} - \lambda^{x,v}\frac{\Delta t}{2\,\Delta x}\right)(\Delta^x U)_{i,j}, \qquad \text{if } \lambda^{x,v} > 0$$

$$= U^n_{i+1,j} - \left(\frac{1}{2} + \lambda^{x,v}\frac{\Delta t}{2\,\Delta x}\right)(\Delta^x U)_{i+1,j}, \qquad \text{otherwise.}$$

This is a finite difference approximation to the characteristic form of Eqs. (2.4) on the M characteristic surfaces intersecting the line $\{(x, y): x = (i+\frac{1}{2})\,\Delta x\}$ at time $t^{n+1/2}$. The proof is a routine calculation using the characteristic projection operators; the key fact that is required is that the solution to the Riemann problem for (2.2) with left and right states W_L, W_R is given by

$$W = P_L W_L + P_R W_R,$$

where P_L, P_R are the projection operators defined in (2.10). In the case where the equations are nonlinear, but the solutions are smooth, $U^{n+1/2}_{i+1/2,j}$ satisfies (2.16) modulo terms which are second order in the mesh spacing, provided that $\tilde{U}_S - U^n_{i+k,j}$ is of the order of the mesh spacing, where the eigenvectors and eigenvalues are evaluated at $U^{n+1/2}_{i+1/2,j}$. This fact describes one sense in which the algorithm described here is upstream-centered for smooth solutions: the value of the predictor $U^{n+1/2}_{i+1/2,j}$ is given as a solution to M linear equations which are finite difference approximations to the characteristic equations.

Finally, we need to specify a bound on the time step for stability. We expect that the CFL condition should be given by

$$\max_{i,j,v}\left(\left|\lambda^{x,v}_{i,j}\frac{\Delta t}{\Delta x}\right|, \left|\lambda^{y,v}_{i,j}\frac{\Delta t}{\Delta y}\right|\right) \leq 1, \tag{2.17}$$

by analogy with the stability condition (1.5) for the advection equation. In the case where A^x and A^y commute, the above stability condition holds in the sense that it held for the scalar equation, i.e., that the fully limited scheme, and the scheme without limiting, both have (2.17) as necessary and sufficient conditions for Fourier stability. This follows easily from the analogous result for scalar equations, plus the fact that the system can be diagonalized. We have not proven (2.17) for any problem for which A^x and A^y do not commute. However, we have used the above condition as a time step control for our gas dynamics calculations and have seen no evidence of instability.

3. Quadrilateral Grids

The above algorithm can be extended to the case of arbitrary quadrilateral grids. For the purposes of deriving the algorithm we will assume that our grid comes from

a smooth coordinate mapping, although the final difference algorithm will be expressed only in terms of differences between coordinates of the corners of the quadrilateral mesh.

We now assume that our computational domain is divided into quadrilaterals $\Delta_{i,j}$ with corners located at $(x_{i+1/2,j+1/2}, y_{i+1/2,j+1/2})$. Furthermore, we assume there is a smooth map $(\xi, \eta) \leftrightarrow (x, y)$ between some coordinate space and physical space, with a rectangular mesh in (ξ, η) space with corners located at $(\xi_{i+1/2}, \eta_{j+1/2})$ such that $(x_{i+1/2,j+1/2}, y_{i+1/2,j+1/2}) = (x(\xi_{i+1/2}, \eta_{j+1/2}), y(\xi_{i+1/2}, \eta_{j+1/2}))$. We can transform the system (2.1) to the (ξ, η) coordinate system:

$$\frac{\partial (JU)}{\partial t} + \frac{\partial F^{\xi}}{\partial \xi} + \frac{\partial F^{\eta}}{\partial \eta} = 0 \tag{3.1}$$

$$J = \mathrm{Det}(\nabla_{(\xi,\eta)}(x, y))$$

$$F^{\xi} = \mathbf{n}^{\eta} \cdot \mathbf{F}, \qquad F^{\eta} = \mathbf{n}^{\xi} \cdot \mathbf{F}$$

$$\mathbf{n}^{\eta} = \left(\frac{\partial y}{\partial \eta}, -\frac{\partial x}{\partial \eta}\right), \qquad \mathbf{n}^{\xi} = \left(-\frac{\partial y}{\partial \xi}, \frac{\partial x}{\partial \xi}\right).$$

Without loss of generality we assume here that $J > 0$. We define finite difference approximations to the derivatives of the grid mapping function:

$$(\Delta^{\xi}\mathbf{x})_{i,j+1/2} = \mathbf{x}_{i+1/2,j+1/2} - \mathbf{x}_{i-1/2,j+1/2} \approx \left.\frac{\partial \mathbf{x}}{\partial \xi}\right|_{\xi_i, \eta_{j+1/2}} \Delta\xi_i$$

$$(\Delta^{\eta}\mathbf{x})_{i+1/2,j} = \mathbf{x}_{i+1/2,j+1/2} - \mathbf{x}_{i+1/2,j-1/2} \approx \left.\frac{\partial \mathbf{x}}{\partial \eta}\right|_{\xi_{i+1/2}, \eta_j} \Delta\eta_j$$

$$(\Delta^{\xi}\mathbf{x})_{i,j} = \tfrac{1}{2}((\Delta^{\xi}\mathbf{x})_{i,j+1/2} + (\Delta^{\xi}\mathbf{x})_{i,j-1/2}) \tag{3.2}$$

$$(\Delta^{\eta}\mathbf{x})_{i,j} = \tfrac{1}{2}((\Delta^{\eta}\mathbf{x})_{i+1/2,j} + (\Delta^{\eta}\mathbf{x})_{i-1/2,j})$$

$$\sigma_{i,j} = \tfrac{1}{2}((x_{i+1/2,j-1/2} - x_{i-1/2,j+1/2})(y_{i+1/2,j+1/2} - y_{i-1/2,j-1/2}) + (x_{i+1/2,j+1/2} - x_{i-1/2,j-1/2})(y_{i-1/2,j+1/2} - y_{i+1/2,j-1/2})).$$

Using these finite differences, we can make the connection between the mapping derivatives appearing in the transformed equations (3.1) and the geometry of the finite difference grid in physical space (Fig. 4): $\sigma_{ij} \approx J(\xi_i, \eta_j)\,\Delta\xi_i\,\Delta\eta_j$ is the area of the (i, j)th zone, and $\mathbf{n}^{\xi}\,\Delta\xi_i \approx -(\Delta^{\xi}\mathbf{x})^{\perp}_{i,j+1/2}$, $\mathbf{n}^{\eta}\,\Delta\eta_j \approx (\Delta^{\eta}\mathbf{x})^{\perp}_{i+1/2,j}$ are normal to the zone edges, where we use the notation $(w_1, w_2)^{\perp} = (w_2, -w_1)$.

As in the previous section, we will assume that, at time step n, we know $U^n_{i,j}$, the average of U over $\Delta_{i,j}$. The procedure for calculating $U^{n+1}_{i,j}$ follows the same basic outline as that for the rectangular grid case. We construct time-centered left and right states at the zone edges, solve the Riemann problem, and difference the fluxes conservatively, taking care that, at each step, the effect of the quadrilateral mesh is accounted for in a suitable fashion.

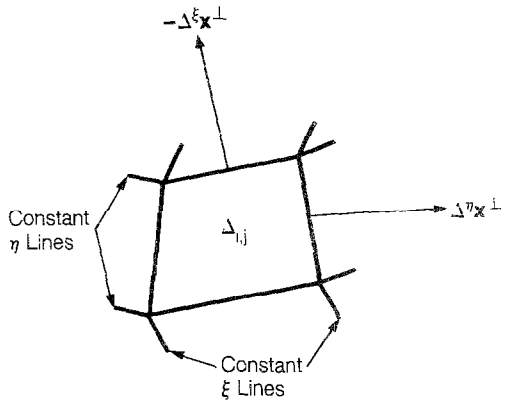

FIG. 4. Geometric interpretation of the difference approximations to the derivatives of the grid mapping.

Our conservative difference step will be of the "finite volume" type:

$$U_{i,j}^{n+1} = U_{i,j}^{n} + \frac{\Delta t}{\sigma_{i,j}}\left((\Delta^{\eta}\mathbf{x})^{\perp}_{i-1/2,j}\cdot\mathbf{F}(U^{n+1/2}_{i-1/2,j}) - (\Delta^{\eta}\mathbf{x})^{\perp}_{i+1/2,j}\cdot\mathbf{F}(U^{n+1/2}_{i+1/2,j})\right.$$
$$\left. - (\Delta^{\xi}\mathbf{x})^{\perp}_{i,j-1/2}\cdot\mathbf{F}(U^{n+1/2}_{i,j-1/2}) + (\Delta^{\xi}\mathbf{x})^{\perp}_{i,j+1/2}\cdot\mathbf{F}(U^{n+1/2}_{i,j+1/2})\right). \tag{3.3}$$

It is clear that this formula is a conservative finite difference approximation to (3.1). This formula can also be obtained by integrating (2.1) over $\Delta_{i,j}\times[t^n, t^{n+1}]$, applying the divergence theorem, and approximating the resulting surface integrals using the midpoint formula. From that point of view, each of the terms multiplied by $\Delta t/\sigma_{i,j}$ represents a time- and space-averaged flux through one of the edges of $\Delta_{i,j}$.

Our strategy for obtaining values for $U^{n+1/2}_{i+1/2,j}$, $U^{n+1/2}_{i,j+1/2}$ follows the pattern used in the rectangular grid case. We extrapolate time-centered left and right limiting states at the zone edges using (3.1). We then solve the Riemann problem using these states for Eqs. (2.1) projected in the direction of the normal to the zone edges in physical space. We consider, for example, the zone edge centered at $(i+1/2, j)$ and we wish to construct $U^{n+1/2}_{i+1/2,j,L}$, $U^{n+1/2}_{i+1/2,j,R}$, the left and right states at that zone edge. The starting point for this is to consider the extrapolation formulae analogous to (2.7) for the system (3.1):

$$U^{n+1/2}_{i+1/2,j,S} = U^{n}_{i+k,j} \pm \frac{\Delta\xi_{i+k}}{2}\frac{\partial U}{\partial\xi} + \frac{\Delta t}{2}\frac{\partial U}{\partial t}$$
$$= U^{n}_{i+k,j} \pm \frac{\Delta\xi_{i+k}}{2}\frac{\partial U}{\partial\xi} - \frac{\Delta t}{2J}\left(\frac{\partial F^{\xi}}{\partial\xi} + \frac{\partial F^{\eta}}{\partial\eta}\right)$$
$$= U^{n}_{i+k,j} + \left(\pm\frac{1}{2} - \frac{\Delta t}{2J\Delta\xi_{i+k}}A^{\xi}\right)\frac{\partial U}{\partial\xi}\Delta\xi_{i+k} - \frac{\Delta t}{2J}\frac{\partial\mathbf{n}^{\eta}}{\partial\xi}\cdot\mathbf{F} - \frac{\Delta t}{2J}\frac{\partial F^{\eta}}{\partial\eta}, \tag{3.4}$$

where $A^\xi = \mathbf{n}^\eta \cdot \mathbf{A}$. The term $(\Delta t/2J)(\partial \mathbf{n}^\eta/\partial \xi)\cdot \mathbf{F}$ comes from putting $\partial F^\xi/\partial \xi$ in non-conservation form and is equal to zero in the rectangular grid case. We break this procedure into two steps:

$$\hat{U}_{i+1/2,j,S} = U^n_{i+k,j} + \left(\pm\frac{1}{2} - \frac{\Delta t}{2J\,\Delta\xi_{i+k}} A^\xi\right)\frac{\partial U}{\partial \xi}\Delta\xi_{i+k} \tag{3.5}$$

$$U^{n+1/2}_{i+1/2,j,S} = \hat{U}_{i+1/2,j,S} - \frac{\Delta t}{2J}\left(\frac{\partial \mathbf{n}^\eta}{\partial \xi}\cdot\mathbf{F} + \frac{\partial F^\eta}{\partial \eta}\right). \tag{3.6}$$

We approximate $\partial U/\partial \xi$ by monotonized central differences and $\partial F^\eta/\partial \eta$ by upwind differences. The term $(\partial \mathbf{n}^\eta/\partial \xi)\cdot\mathbf{F}$ is differenced in such a way as to exactly cancel

We first consider the calculation of $\hat{U}_{i+1/2,j,S}$. We approximate

$$\left(\pm\frac{1}{2} - \frac{\Delta t}{2J\,\Delta\xi_{i+k}} A^\xi\right) \approx \left(\pm\frac{1}{2} - \frac{\Delta t}{2\sigma_{i+k,j}} (\Delta^\eta \mathbf{x})^\perp_{i+k,j}\cdot\mathbf{A}(U^n_{i+k,j})\right), \tag{3.7}$$

where we have replaced J and $\mathbf{n}^\xi$, $\mathbf{n}^\eta$ by the appropriate difference approximations from (3.2). By analogy with the rectangular grid case, we want to approximate $(\partial U/\partial \xi)\,\Delta\xi_i$ with $(\Delta^\xi U)_{i,j}$, a central difference approximation to which some form of monotonicity constraint has been applied. If the coordinate mapping is smooth, then the formula (2.5) for equally spaced zones can be used without modification, while retaining second-order accuracy in regions where the solution is smooth. However, we replace the eigenvectors in the monotonicity constraints in (2.6) by $(l^{\xi,v}_{i,j}, r^{\xi,v}_{i,j})$, $v = 1, \ldots, M$, the left and right eigenvectors corresponding to the eigenvalues $\lambda^{\xi,1}_{i,j} \leqslant \cdots \leqslant \lambda^{\xi,M}_{i,j}$ of $(\Delta^\eta \mathbf{x})^\perp_{i,j}\cdot\mathbf{A}(U^n_{i,j})$. As before, we can also discard terms in (3.7) corresponding to signals propagating away from the zone edge and allow for an arbitrary choice of reference state $\tilde{U}_S$, obtaining the following analogue of (2.10) for a general quadrilateral grid:

$$\begin{aligned}\hat{U}_{i+1/2,j,S} = {} & \tilde{U}_S + P_S(U^n_{i,j} - \tilde{U}_S) \\ & + P_S\left(\pm\frac{1}{2} - \frac{\Delta t}{2\sigma_{i+k,j}} (\Delta^\eta \mathbf{x})^\perp_{i+k,j}\cdot\mathbf{A}(U^n_{i+k,j})\right)\cdot(\Delta^\xi U)_{i+k,j},\end{aligned} \tag{3.8}$$

where

$$P_S w = \sum_{v:\, \pm\lambda^{\xi,v}_{i+k,j} > 0} (l^{\xi,v}_{i+k,j}\cdot w)\, r^{\xi,v}_{i+k,j}.$$

We approximate $(\Delta t/2J)(\partial F^\eta/\partial \eta)$ by an appropriate upwind difference approximation. In general, it is of the form of the corresponding difference approximation in the conservative difference step (3.3):

$$-\frac{\Delta t}{2J}\frac{\partial F^\eta}{\partial \eta} \approx \frac{\Delta t}{2\sigma_{i,j}}\left((\Delta^\xi \mathbf{x})^\perp_{i,j+1/2}\cdot\mathbf{F}(U^{\mathrm{T}}_{i,j+1/2}) - (\Delta^\xi \mathbf{x})^\perp_{i,j-1/2}\cdot\mathbf{F}(U^{\mathrm{T}}_{i,j-1/2})\right). \tag{3.9}$$

Here $U^{\mathrm{T}}_{i,j+1/2}$ is calculated by solving a Riemann problem for the projected equations along $-(\Delta^{\xi}\mathbf{x})^{\perp}_{i,j+1/2}$ with left and right states $(U^{\mathrm{T}}_{i,j+1/2,L}, U^{\mathrm{T}}_{i,j+1/2,R})$. As in the rectangular grid case, $U^{\mathrm{T}}_{i,j+1/2,S}$ may be set to $U^{n}_{i,j+1}$ or $\hat{U}_{i,j+1/2,S}$. Finally, we approximate $(\Delta t/2J)(\partial\mathbf{n}^{\eta}/\partial\eta)\cdot\mathbf{F}$ using the finite difference approximations (3.2):

$$\frac{\Delta t}{2J}\frac{\partial \mathbf{n}^{\eta}}{\partial \xi}\cdot\mathbf{F} \approx \frac{\Delta t}{2\sigma_{i,j}}\left((\Delta^{\eta}\mathbf{x})^{\perp}_{i+1/2,j}-(\Delta^{\eta}\mathbf{x})^{\perp}_{i-1/2,j}\right)\cdot\mathbf{F}(U^{n}_{i,j}). \tag{3.10}$$

Collecting our difference approximations, our final value for $U^{n+1/2}_{i+1/2,j,S}$ is given by

$$\begin{aligned} U^{n+1/2}_{i+1/2,j,S} = \hat{U}_{i+1/2,j,S} + \frac{\Delta t}{2\sigma_{i,j}}\Big[&(\Delta^{\xi}\mathbf{x})^{\perp}_{i+k,j+1/2}\cdot\mathbf{F}(U^{\mathrm{T}}_{i+k,j+1/2}) \\ &-(\Delta^{\xi}\mathbf{x})^{\perp}_{i+k,j-1/2}\cdot\mathbf{F}(U^{\mathrm{T}}_{i+k,j-1/2}) \\ &-\left((\Delta^{\eta}\mathbf{x})^{\perp}_{i+1/2+k,j}-(\Delta^{\eta}\mathbf{x})^{\perp}_{i-1/2+k,j}\right)\cdot\mathbf{F}(U^{n}_{i+k,j})\Big]. \end{aligned} \tag{3.11}$$

We obtain $U^{n+1/2}_{i+1/2,j}$ by solving the Riemann problem for the projected equations along $(\Delta^{\eta}\mathbf{x})^{\perp}_{i+1/2,j}$ with left and right states $U^{n+1/2}_{i+1/2,j,L}$, $U^{n+1/2}_{i+1/2,j,R}$. $U^{n+1/2}_{i+1/2,j}$ satisfies finite difference approximations to the characteristic equations (2.4) for the characteristic surfaces through the $(i+1/2, j)$th zone edge in physical space, similar to (2.14).

The appropriate generalization of (2.17) as a CFL condition on the time step is given by

$$\max_{i,j,v}\left(\left|\lambda^{\xi,v}_{i,j}\frac{\Delta t}{\sigma_{i,j}}\right|, \left|\lambda^{\eta,v}_{i,j}\frac{\Delta t}{\sigma_{i,j}}\right|\right) \leq 1. \tag{3.12}$$

This is dimensionally correct since $\lambda^{\xi,v}_{i,j}$, $\lambda^{\eta,v}_{i,j}$ contain factors of $\Delta^{\eta}\mathbf{x}$, $\Delta^{\xi}\mathbf{x}$. In the case of advection, and if the coordinate transformation is a linear map, one can demonstrate by numerical evaluation of the Fourier transform, as was done for the rectangular mesh case, that this is the correct CFL condition. In general, the time step bound (3.12) has the following interpretation in terms of characteristics: Δt must be less than the time it takes a wave propagating in a direction normal to a zone edge to reach an opposite zone edge.

4. Gas Dynamics

We give in this section a detailed description of an algorithm of the type described above for the case of Euler's equations for inviscid compressible flow in two space variables, in planar geometry, on a general quadrilateral grid. The system we wish to solve is of the form (2.1), with $M=4$, and

$$U=\begin{pmatrix}\rho\\ \rho u\\ \rho v\\ \rho E\end{pmatrix},\qquad F^x(U)=\begin{pmatrix}\rho u\\ \rho u^2+p\\ \rho uv\\ \rho uE+up\end{pmatrix},\qquad F^y(U)=\begin{pmatrix}\rho v\\ \rho uv\\ \rho v^2+p\\ \rho vE+p\end{pmatrix}, \tag{4.1}$$

where ρ is the density, $(u, v)=\mathbf{u}$ the x and y components of velocity, and E the total energy per unit mass. The pressure is derived from these quantities via an equation of state, $p=p(\rho, e)$, where e is the internal energy per unit mass, given by $e=E-\frac{1}{2}(u^2+v^2)$. In this section, we will describe an algorithm suitable for use with a polytropic equation of state, i.e., for p given by $p(\rho, e)=\rho e(\gamma-1)$, and the adiabatic speed of sound c given by $c^2=\gamma p/\rho$. The case of a general convex equation of state is a straightforward extension of ideas in [6].

The projected equations for the system (4.1), are essentially those of gas dynamics in one dimension. If we project the equations in the $\mathbf{n}$ direction for $\mathbf{n}$ a unit vector, we can make a change of variables to obtain the following system equivalent to (2.2):

$$\frac{\partial W}{\partial t}+\frac{\partial G(W)}{\partial \chi}=0 \tag{4.2}$$

$$W=\begin{pmatrix}\rho\\ \rho u^{\mathrm{N}}\\ \rho u^{\mathrm{T}}\\ \rho E\end{pmatrix},\qquad G(W)=\begin{pmatrix}\rho u^{\mathrm{N}}\\ \rho(u^{\mathrm{N}})^2+p\\ \rho u^{\mathrm{N}}u^{\mathrm{T}}\\ \rho u^{\mathrm{N}}E+u^{\mathrm{N}}p\end{pmatrix}$$

Here $u^{\mathrm{N}}=\mathbf{u}\cdot\mathbf{n}$, $u_{\mathrm{T}}=\mathbf{u}\cdot\mathbf{n}^{\perp}$ with the other variables defined as before. Since $\mathbf{n}$ is a unit vector, $u^2+v^2=(u^{\mathrm{N}})^2+(u^{\mathrm{T}})^2$ so the formula for the internal energy e can use either quantity. From these equations, it is clear that the eigenvectors and eigenvalues of the linearized system, as well as the solution to the Riemann problem, are given by those for the 1-dimensional gas dynamucs equations, with u^{T} being treated as a passively advected quantity. Hence, we can use the techniques of [4, 7] for calculating solutions to the Riemann problem and for manipulating characteristic variables.

Although the algorithm described here follows the same basic outline as those given in the previous two sections, there are some differences, mainly with the calculation of $\hat{U}_{i+1/2,j,S}$. For the purpose of calculating $\hat{U}_{i+1/2,j,S}$, we make a nonlinear change of variables, performing the difference calculation of (3.5) in terms of the primitive variables ρ, u, v, p, as was done in [7] for gas dynamics in one space variable. We then transform back to the conserved variables to calculate $U^{n+1/2}_{i+1/2,j,S}$. This procedure enables us to perform our central difference calculation componentwise on the primitive variables, using formulas similar to (1.9), rather than on the amplitudes of an expansion of $\Delta^{\xi}U$ in terms of the right eigenvectors. Also, since we are working in terms of the primitive variables, we can use the more

elaborate central difference algorithm given in [6], which gives rise to a steeper representation of dicontinuities than (1.9).

In order to justify the use of the more elaborate algorithm for computing $\partial U/\partial\xi$ and, more generally, to understand the errors introduced by using difference approximations to $\partial U/\partial\xi$, such as (2.5), it is useful to make a local change of variables $(\xi, \eta) \leftrightarrow (a, b)$

$$a(\xi,\eta)=\int_{\xi_i}^{\xi}\left(\left(\frac{\partial x}{\partial\xi}\right)^2+\left(\frac{\partial y}{\partial\xi}\right)^2\right)^{1/2}d\xi'$$
$$b(\xi,\eta)=\int_{\eta_j}^{\eta}\left(\left(\frac{\partial x}{\partial\eta}\right)^2+\left(\frac{\partial y}{\partial\eta}\right)^2\right)^{1/2}d\eta'. \tag{4.3}$$

The coordinate (a, b) measure arc length along the grid lines $\{\eta = \text{const}\}$, $\{\xi = \text{const}\}$, respectively. It is easy to check that, for (ξ, η) sufficiently close to (ξ_i, η_j) the Jacobian of the above map is nonsingular, since the cross derivatives $\partial a/\partial\eta$, $\partial b/\partial\xi = O((\xi-\xi_i), (\eta-\eta_j))$. Using the chain rule, we compute $\partial U/\partial\xi$ to be

$$\frac{\partial U}{\partial\xi}\Delta\xi=\frac{\partial U}{\partial a}\frac{\partial a}{\partial\xi}\Delta\xi+\frac{\partial U}{\partial b}\frac{\partial b}{\partial\xi}\Delta\xi.$$

Thus, the central difference approximation to $\partial U/\partial\xi$ used in (3.8) can be viewed as using a central difference approximation for $\partial U/\partial a$ and dropping the term proportional to $\partial b/\partial\xi$, since it is of one order smaller in the mesh spacing. In terms of the mesh in physical space, this corresponds to the assumption that the arc length along each of the coordinate directions is a smoothly varying function of the other coordinate. This is a condition satisfied in a wide variety of applications, even when the grid mapping as a whole is not smooth, such as in the case of highly stretched grids used in aerodynamics calculations. In the latter situation, one can retain the formalism developed here but use an approximation to the derivatives appropriate for a strongly varying mesh in the a- or b- direction.

In terms of the coordinate system (4.3), we can express $U^{n+1/2}_{i+1/2,j,S}$ in the form

$$\hat{U}_{i+1/2,j,S}=U^n_{i,j}+\left(\pm\frac{1}{2}-\frac{\Delta t\,\Delta b_{i+k,j}}{2\sigma_{i+k,j}}\mathbf{n}^b_{i+k,j}\cdot A(U^n_{i+k,j})\right)\frac{\partial U}{\partial a}\Delta a_{i,j} \tag{4.4}$$

$$\begin{aligned}U^{n+1/2}_{i+1/2,j,S}=\hat{U}_{i+1/2,j,S}-\frac{\Delta t}{2\sigma_{i+k,j}}\big[&\Delta a_{i+k,j+1/2}\mathbf{n}^a_{i+k,j+1/2}\mathbf{F}(U^T_{i+k,j+1/2})\\&-\Delta a_{i+k,j-1/2}\mathbf{n}^a_{i+k,j-1/2}\cdot\mathbf{F}(U^T_{i+k,j-1/2})\\&+\mathbf{F}(U^n_{i+k,j})\cdot(\Delta b_{i+1/2+k}\mathbf{n}^b_{i+1/2+k,j}-\Delta b_{i-1/2+k}\mathbf{n}^b_{i-1/2+k,j})\big],\end{aligned} \tag{4.5}$$

where

$$
\begin{aligned}
(\Delta a)_{i,j[+1/2]} &= ((\Delta^{\xi}x)^2_{i,j[+1/2]} + (\Delta^{\xi}y)^2_{i,j[+1/2]})^{1/2} \\
(\Delta b)_{i[+1/2],j} &= ((\Delta^{\eta}x)^2_{i[+1/2],j} + (\Delta^{\eta}y)^2_{i[+1/2],j})^{1/2} \\
\mathbf{n}^b_{i[+1/2],j} &= \frac{(\Delta^{\eta}\mathbf{x})^{\perp}_{i[+1/2],j}}{\Delta b_{i[+1/2],j}} \\
\mathbf{n}^a_{i,j[+1/2]} &= -\frac{(\Delta^{\xi}\mathbf{x})^{\perp}_{i,j[+1/2]}}{\Delta a_{i,j[+1/2]}}.
\end{aligned}
\tag{4.6}
$$

We calculate $\hat{U}_{i+1/2,j,S}$ by transforming to the variables $V=(\rho, u, v, p)^t$ before applying (4.5):

$$
\begin{aligned}
V^n_{i,j} &= V(U^n_{i,j}) \\
\hat{V}_{i+1/2,j,S} &= \tilde{V}_S + P_S(V^n_{i,j} - \tilde{V}_S) + P_S\left(\pm\frac{1}{2} - \frac{\Delta t\, \Delta b_{i+k,j}}{2\sigma_{i+k,j}} T^{-1}_{i+k,j} A^a_{i+k,j} T_{i+k,j}\right) \\
&\quad \times \frac{\partial V}{\partial a}\Delta a_{i+k,j} \\
\hat{U}_{i+1/2,j,S} &= U(\hat{V}_{i+1/2,j,S}).
\end{aligned}
\tag{4.7}
$$

Here $T_{i,j} = \nabla_V U|_{U^n_{i,j}}$ and P_S is defined by $P_S w = \sum_{\nu:\pm\lambda^{a,\nu}_{i+k,j}>0} (l^{a,\nu}_{i+k,j} w)\, r^{a\nu}_{i+k,j}$, where $l^{a,\nu}_{i,j}$, $r^{a,\nu}_{i,j}$, $\lambda^{a,\nu}_{i,j}$, $\nu = 1, \ldots, 4$ are the eigenvectors and eigenvalues of $T^{-1}_{i,j} \cdot A^a_{i,j} \cdot T_{i,j}$:

$$
\lambda^{a,1} = \mathbf{u}\cdot\mathbf{n}^b - c, \qquad \lambda^{a,2} = \lambda^{a,3} = \mathbf{u}\cdot\mathbf{n}^b, \qquad \lambda^{a,4} = \mathbf{u}\cdot\mathbf{n}^b + c
$$

$$
r^{a,1} = \begin{pmatrix} 1 \\ -\dfrac{n^b_x c}{\rho} \\ -\dfrac{n^b_y c}{\rho} \\ c^2 \end{pmatrix}, \qquad r^{a,2} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \qquad r^{a,3} = \begin{pmatrix} 0 \\ -n^b_y \\ b^b_x \\ 0 \end{pmatrix}, \qquad r^{a,4} = \begin{pmatrix} 1 \\ \dfrac{n^b_x c}{\rho} \\ \dfrac{n^b_y c}{\rho} \\ c^2 \end{pmatrix}
$$

$$
\begin{aligned}
l^{a,1} &= \left(0, -\frac{n^b_x \rho}{2c}, -\frac{n^b_y \rho}{2c}, \frac{1}{2c^2}\right) \\
l^{a,2} &= \left(1, 0, 0, -\frac{1}{c^2}\right) \\
l^{a,3} &= (0, -n^b_y, n^b_x, 0) \\
l^{a,4} &= \left(0, \frac{n^b_x \rho}{2c}, \frac{n^b_y \rho}{2c}, \frac{1}{2c^2}\right).
\end{aligned}
$$

Here $\mathbf{n}^b = (n_x^b, n_y^b)$ and the subscripts i, j are suppressed. The time step control (3.12) in terms of the above eigenvalues, is given by

$$\max_{i,j,\nu}\left(\left|\lambda_{i,j}^{a,\nu}\frac{\Delta b_{i,j}\,\Delta t}{\sigma_{i,j}}\right|, \left|\lambda_{i,j}^{b,\nu}\frac{\Delta a\, i,j\,\Delta t}{\sigma_{i,j}}\right|\right) \leqslant 1.$$

The approximation to $(\partial V/\partial a)|_{i,j}\,\Delta a_{i,j}$ we use is obtained by using a formula like (1.9) for each component of V. For example, we define, for $q = p, \rho, u, v$,

$$\begin{aligned}(\Delta^a_{\lim} q)_{i,j} &= 2\min(|q^n_{i+1,j} - q^n_{i,j}|, |q^n_{i,j} - q^n_{i-1,j}|)\\ &\qquad \text{if} \quad (q^n_{i+1,j} - q^n_{i,j})(q^n_{i,j} - q^n_{i-1,j}) > 0,\\ &= 0 \qquad \text{otherwise},\\ (\Delta^a_f q)_{i,j} &= \min(\tfrac{1}{2}|q^n_{i+1,j} - q^n_{i+1,j}|, (\Delta^a_{\lim} q)_{i,j}) \times \operatorname{sgn}(q^n_{i+1,j} - q^n_{i+1,j}))\end{aligned}$$

and set $(\Delta^a q)_{i,j} = (\Delta^a_f q)_{i,j}$ to obtain the algorithm analogous to (1.9). In the calculations presented in Section 5, we use the following algorithm, taken from [5], which yields a steeper representation of discontinuities:

$$(\Delta^a q)_{i,j} = \min\left(4\,\frac{|q_{i+1,j} - q_{i-1,j} - (1/4)((\Delta^a_f q)_{i+1,j} + (\Delta^a_f q)_{i-1,j})|\,\Delta a_{i,j}}{(\Delta a_{i-1,j} + 4\,\Delta a_{i,j} + \Delta a_{i+1,j})}, (\Delta^a_{\lim} q)_{i,j}\right)$$
$$\times \operatorname{sgn}(q^n_{i+1,j} - q^n_{i-1,j}).$$

Given the values for $\Delta^a V$, we can give explicit formulas for $\hat{V}_{i+1/2,j,S}$:

$$\tilde{V}_L = V^n_{i,j} + \left(\frac{1}{2} - \max(\mathbf{u}^n_{i,j}\cdot\mathbf{n}^b_{i,j} + c^n_{i,j}, 0)\,\frac{\Delta t\,\Delta b_{i,j}}{2\sigma_{i,j}}\right)\Delta^a V_{i,j}$$

$$\tilde{V}_R = V^n_{i+1,j} - \left(\frac{1}{2} + \min(\mathbf{u}^n_{i+1,j}\cdot\mathbf{n}^b_{i+1,j} - c_{i+1,j}, 0)\,\frac{\Delta t\,\Delta b_{i+1,j}}{2\sigma_{i+1,j}}\right)\Delta^a V_{i+1,j}$$

$$\hat{V}_{i+1/2,j,S} = \tilde{V}_S + \sum_{\nu}\beta_{i+1/2,j,S}\,r^{a,\nu}_{i+k,j}$$

$$\beta^{\nu}_{i+1/2,j,L} = \frac{\Delta t\,\Delta b_{i,j}}{2\sigma_{i,j}}(\lambda^{a,4}_{i,j} - \lambda^{a,\nu}_{i,j})(l^{a,\nu}_{i,j}\cdot\Delta^a V_{i,j}) \qquad \text{if} \quad \lambda^{a,\nu}_{i,j} > 0,$$

$$= 0 \qquad \text{otherwise:}$$

$$\beta^{\nu}_{i+1/2,j,R} = \frac{\Delta t\,\Delta b_{i+1,j}}{2\sigma_{i+1,j}}(\lambda^{a,1}_{i+1,j} - \lambda^{a,\nu}_{i+1,j})(l^{a,\nu}_{i+1,j}\cdot\Delta^a V_{i+1,j}) \qquad \text{if} \quad \lambda^{a,\nu}_{i+1,j} < 0,$$

$$= 0 \qquad \text{otherwise.}$$

The formulas for $\hat{V}_{i,j+1/2,S}$ are identical to those given above, with the interchange of i and j, $\mathbf{n}^a$ and $\mathbf{n}^b$.

The calculation of $U^{n+1/2}_{i+1/2,j,S}$ given $\hat{U}_{i+1/2,j,S}$ is given by (4.5), with $U^{\mathrm{T}}_{i,j+1/2}$ the solution to the Riemann problem for the equations projected in the $\mathbf{n}^a_{i,j+1/2}$ direction, with left and right states given by $U^{\mathrm{T}}_{i,j+1/2,S} = \hat{U}_{i,j+1/2,S}$ or $U^{\mathrm{T}}_{i,j+1,2,s} = U^n_{i,j+k}$. In the calculations shown below, we use the latter choice.

The final conservative difference step is given by (3.3). We define

$$F^{\xi}_{i+1/2,j} = \begin{pmatrix} m^{n+1/2}_{i+1/2,j} \\ m^{n+1/2}_{i+1/2,j} u^{n+1/2}_{i+1/2,j} + p^{n+1/2}_{i+1/2,j} \Delta^{\eta} y_{i+1/2,j} \\ m^{n+1/2}_{i+1/2,j} v^{n+1/2}_{i+1/2,j} - p^{n+1/2}_{i+1/2,j} \Delta^{\eta} x_{i+1/2,j} \\ m^{n+1,2}_{i+1/2,j} (E^{n+1/2}_{i+1,2,j} + p^{n+1/2}_{i+1/2,j}/\rho^{n+1/2}_{i+1/2,j}) \end{pmatrix}$$

$$F^{\eta}_{i,j+1/2} = \begin{pmatrix} m^{n+1/2}_{i,j+1/2} \\ m^{n+1/2}_{i,j+1/2} u^{n+1/2}_{i,j+1/2} - p^{n+1/2}_{i,j+1/2} \Delta^{\xi} y_{i,j+1/2} \\ m^{n+1/2}_{i,j+1,2} v^{n+1/2}_{i,j+1/2} p^{n+1/2}_{i,j+1/2} \Delta^{\xi} x_{i,j+1/2} \\ m^{n+1/2}_{i,j+1/2} (E^{n+1/2}_{i,j+1/2} + p^{n+1,2}_{i,j+1,2}/\rho^{n+1/2}_{i,j+1/2}) \end{pmatrix},$$

where $m^{n+1/2}_{i+1/2,j} = \Delta b_{i+1/2,j} \rho^{n+1/2}_{i+1/2,j} (\mathbf{n}^b_{i+1/2,j} \cdot \mathbf{u}^{n+1/2}_{i+1/2,j})$, $m^{n+1/2}_{i,j+1/2} = \Delta a_{i,j+1/2} \rho^{n+1/2}_{i,j+1/2}$ $(\mathbf{n}^a_{i,j+1/2} \cdot \mathbf{u}^{n+1/2}_{i,j+1/2})$ are the mass fluxes through the zone edges at $(i+\frac{1}{2}, j)$ and $(i, j+\frac{1}{2})$. Then (3.3) is given by

$$U^{n+1}_{i,j} = U^n_{i,j} + \frac{\Delta t}{\sigma_{i,j}} (F^{\xi}_{i-1/2,j} - F^{\xi}_{i+1/2,j} + F^{\eta}_{i,j-1/2} - F^{\eta}_{i,j+1/2}).$$

Dissipation Mechanisms

In [7], it was noticed that, in one space dimension, and near strongly nonlinear shocks, the dissipation implicit in monotonicity constraints such as (3.6) and (4.8), was insufficient to guarantee the correct jump in the Riemann invariants transported along the characteristic families which cross the shock. For that reason, it was suggested that additional dissipation be added to the algorithm near such discontinuities in the form of flattening of the interpolation functions and by adding a small viscous dissipation term to the fluxes. Since both these forms of dissipation were required for 1-dimensional problems, it is expected that similar dissipation would be required for the present algorithm, since, for 1-dimensional problems, it is similar to the algorithm in [7]. The second-order artificial viscosity used in [7] can be applied without modification to the present algorithms simply by adding the dissipative flux to each of the four fluxes, prior to the conservative differencing step. The form these dissipative fluxes take in the case of a general quadrilateral grid is also standard; see, e.g., [19]. The simplest flattening algorithm in [7] can be used, with one important modification: in each zone, the slopes corresponding to the

derivatives in each of the grid directions should be flattened by the same amount. We define flattening χ^a, χ^b,

$$\tilde{\chi}^a_{i,j} = \zeta\left(\frac{|p_{i+1,j}-p_{i-1,j}|}{|p_{i+2,j}-p_{i-2,j}|}\right) \quad \text{if } (\mathbf{u}_{i-1,j}-\mathbf{u}_{i+1,j})\cdot\mathbf{n}^a_{i,j}>0,\ \frac{|p_{i+1,j}-p_{i-1,j}|}{\min(p_{i+1,j},p_{i-1,j})}>\delta$$
$$= 0 \qquad \text{otherwise} \tag{4.9}$$
$$\chi^a_{i,j} = \min(\tilde{\chi}^a_{i-s_{ij},j}, \tilde{\chi}^a_{i,j}),$$

where

$$\zeta(z) = 0 \qquad \text{if } z > z_1,$$
$$= 1 \qquad \text{if } z < z_0,$$
$$= 1 - \frac{z-z_0}{z_1-z_0} \qquad \text{if } z_0 < z < z_1,$$

and

$$s_{i,j} = \text{sign}(p_{i+1,j}-p_{i-1,j}).$$

We define $\chi^b_{i,j}$ similarly, with the roles of i and j reversed. Then the slopes $\Delta^a q$, $\Delta^b q$ obtained from (4.8) are reset to

$$\Delta^a q_{i,j}, \Delta^b q_{i,j} \to \chi_{i,j}\Delta^a q_{ij}, \chi_{i,j}\Delta^b q_{i,j}, \tag{4.10}$$

where

$$\chi_{i,j} = \min(\chi^a_{i,j}, \chi^b_{i,j}).$$

In the runs discussed in the next section, the parameters in the above algorithm were set to be $\delta = 0.33$, $z_0 = 0.75$, $z_1 = 0.85$. In addition, we used the 2-dimensional Lapidus viscous flux discussed in [7] with a coefficient of 0.1. These were the choice of the parameters used in the corresponding algorithms for operator split calculations described in [7] and have been found to give adequate results when used with the present algorithm over a wide range of problems.

Boundary Conditions

It is straightforward to impose various continuation-type boundary conditions (inflow, outflow, periodic, etc.) in regions where the grid has a natural extension beyond the computational domain. Since the numerical domain of dependence of a grid point is contained in the 9×9 block of grid points containing the point at the center, then one can extend the original computational mesh by four grid points in each direction and set the values of the extended part of the grid at the beginning of each time step using the boundary conditions, thus supplying sufficient data to calculate the values on the original grid.

The most common situation where one cannot extend the grid is in the case of an impermeable surface, particularly on a body-fitted grid. Let us assume, for example, that the curve $\{\xi(\mathbf{x}) = \xi_{i_0-1/2}\}$ is a reflecting surface, with the fluid contained in the region $\{\xi(\mathbf{x}) > \xi_{i_0-1/2}\}$. The algorithm described above can be applied without modification, if we specify values for the slopes $\Delta_f^b q_{i_0-1,2,j}$, $\Delta^b q_{i_0-1/2,j}$ and for the fluxes $\mathbf{F}(U^{\mathrm{T}}_{i_0-1/2,j})$, $\mathbf{F}(U^{n+1,2}_{i_0-1\,2,j})$. The slopes are given by

$$\Delta q_{i_0,j} = \Delta_i q_{i_0,j} = 0, \qquad q = p, \rho, \mathbf{n}^{b\perp}_{i_0-1\,2,j}\cdot\mathbf{u}$$

$$\mathbf{n}^b_{i_0-1/2,j}\cdot\Delta_f^b\mathbf{u}_{i_0,j} = \min(|\mathbf{u}_{i_0,j}\cdot\mathbf{n}^b_{i_0-1/2,j}|,\, 2|(\mathbf{u}_{i_0+1,j}-\mathbf{u}_{i_0,j})\cdot\mathbf{n}^b_{i_0-1/2,j}|)\,\mathrm{sgn}(\mathbf{u}_{i_0,j}\cdot\mathbf{n}^b_{i_0-1/2,j})$$

$$\text{if} \quad (\mathbf{u}_{i_0,j}\cdot\mathbf{n}^b_{i_0-1/2,j})(\mathbf{u}_{i_0+1,j}-\mathbf{u}_{i_0,j})\cdot\mathbf{n}^b_{i_0-1.2,j} > 0$$

$$= 0 \qquad \text{otherwise.} \tag{4.11}$$

Given the slope information, it is possible to calculate $\hat{U}_{i_0-1/2,j,R}$, $U^{n+1/2}_{i_0-1/2,j,R}$. To obtain the states $U^{\mathrm{T}}_{i_0-1/2,j}$, $U^{n+1/2}_{i_0-1/2,j}$, we solve Riemann problems projected in the $\mathbf{n}^b_{i_0-1/2,j}$ direction, with left and right state given by

$$\hat{q}_{i_0-1/2,j,L},\, q^{n+1/2}_{i_0-1/2,j,L} = \hat{q}_{i_0-1/2,j,R},\, q^{n+1/2}_{i_0-1/2,j,R}, \qquad q = p, \rho, \mathbf{n}^{b\perp}_{i_0-1/2,j}\cdot\mathbf{u}$$

$$\mathbf{n}^b_{i_0-1.2,j}\cdot\hat{\mathbf{u}}_{i_0-1/2,j,L},\, \mathbf{n}^b_{i_0-1\cdot2,j}\cdot\mathbf{u}^{n+1/2}_{i_0-1/2,j,L} = -\mathbf{n}^b_{i_0-1/2,j}\cdot\hat{\mathbf{u}}_{i_0-1/2,j,R},\, -\mathbf{n}^b_{i_0-1,2,j,R}. \tag{4.12}$$

With this choice of left and right states, it is clear that $\mathbf{u}_{i_0-1/2,j} = 0$, so that the advective terms in the fluxes at $(i_0 - \frac{1}{2}, j)$ vanish, leaving only the pressure terms in the x- and y-momentum equations. Whatever approximate solution to the Riemann problem is used should guarantee that the advective terms vanish in the flux calculation at the wall.

5. Numerical Results

The gas dynamics algorithm described here has been used in a variety of applications in two dimensions, including flow in cascades and channels with body-fitted meshes [9], in adaptive mesh refinement calculations [2], and in a conservative front-tracking algorithm [3]. In addition, various forms of the algorithm for scalar equations have been used to calculate flow in porous media [15].

We will present here two gas dynamics calculations, both done on rectangular grids. The first is the calculation of a steady state regular shock reflection described in [23], which has been used extensively as a test problem for numerical methods used in aerodynamic calculations [25]. The second test problem is the double Mach reflection of a shock off an oblique surface, used in [22] as a test problem for comparing the performance of various difference methods on problems involving strong shocks. Since our purpose is to demonstrate that the current method has the same resolution as the corresponding operator split algorithm, we present also a calculation of the latter problem performed by using in an operator

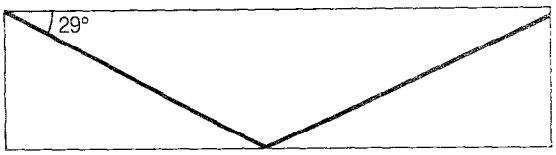

FIG. 5. Steady state regular reflection problem.

split formulation the 1-dimensional algorithm obtained by restricting the algorihtm described in Section 4 to one dimension.

In the first test problem, the computational domain is a rectangle of length 4 and height 1 (Fig. 5). This domain is divided into a 60×20 rectangular grid, with $\Delta x = \frac{1}{15}$, $\Delta y = \frac{1}{20}$. The boundary conditions are that of a reflecting surface along the bottom boundary, supersonic outflow along the right boundary, and Dirichlet conditions on the other two sides, given by

$$(\rho, u, v, p)|_{(0, y, t)} = (1., 2.9, 0., 1/1.4)$$

$$(\rho, u, v, p)|_{(x, 1, t)} = (1.69997, 2.61934, .50632, 1.52819).$$

Initially, we set the solution in the entire domain to be that at the left boundary; we then iterate for 500 time steps using a CFL condition of 0.9, at which time the solution reaches a steady state.

In Fig. 6, we show a contour plot of the pressure. The contours are equally

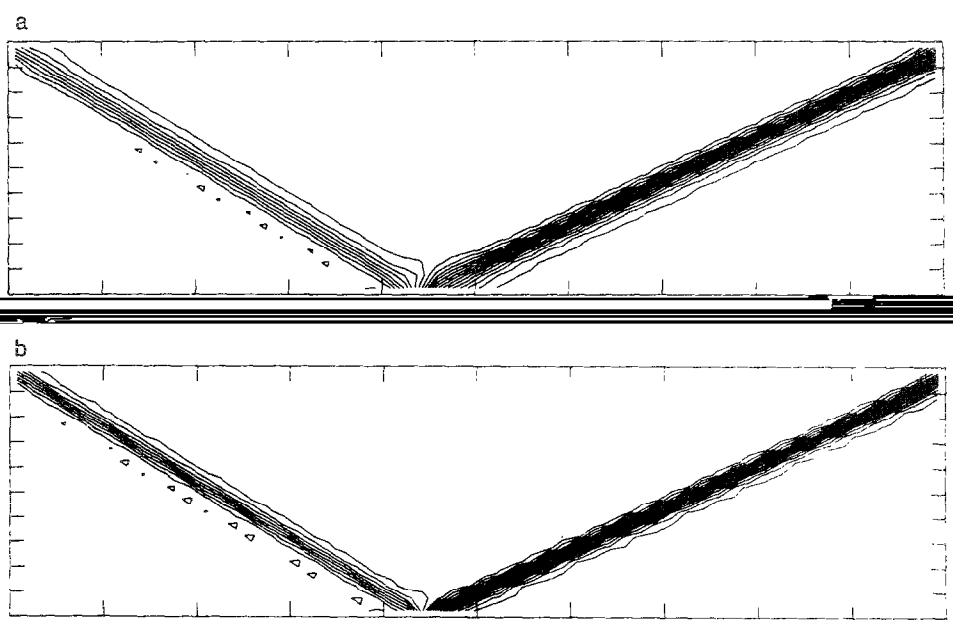

FIG. 6. Numerical solution to regular reflection problem: (a) with flattening; (b) without flattening.

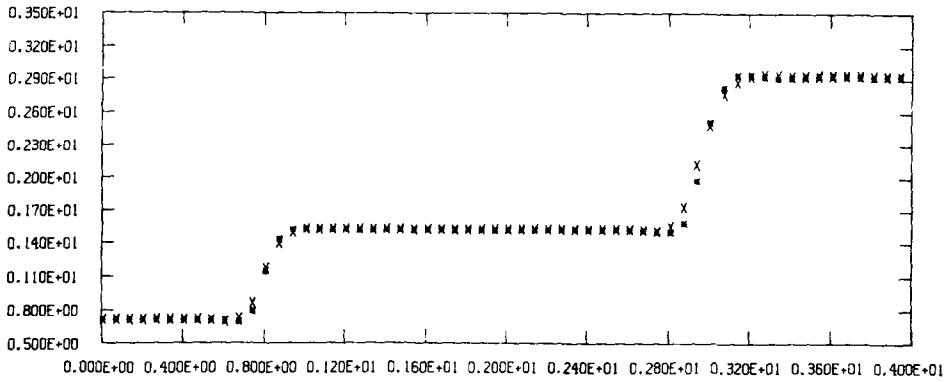

FIG. 7. Comparison of pressure profiles for regular reflection problem along the line $y = 0.525$ ($j = 11$): x—with flattening, $*$—without flattening.

spaced, with contour levels of 0.1, beginning at 0. The shocks have a nearly monotone transition, and are fairly narrow, with some slight spreading on the high pressure side of each shock. This spreading is due to the flattening algorithm (4.10). We see this in Fig. 7, where we plot profiles of the solution at $y = 0.525$, computed with and without flattening. The width of the shocks is about 2–$2\frac{1}{2}$ zones in the normal direction, where this figure is obtained by counting the number of points in the transition in Fig. 7, and multiplying it by $\sin(\tan^{-1}((\Delta x/\Delta y)|\tan(\alpha)|))$, where α is the angle between the direction tangent to the shock and the x direction. The shock transition with flattening is slightly broader; however, the transition without flattening has some low-amplitude oscillations, which are not present in the solution obtained with flattening. Even though the shocks are supersonic on both

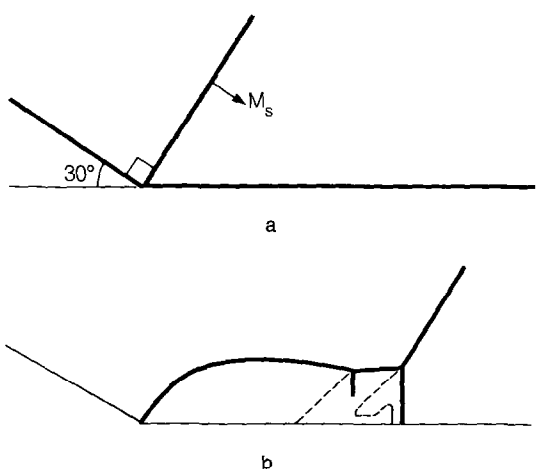

FIG. 8. Ramp reflection problem: (a) initial configuration; (b) double Mach reflection at later times: solid lines are shocks; dotted lines are slip surfaces.

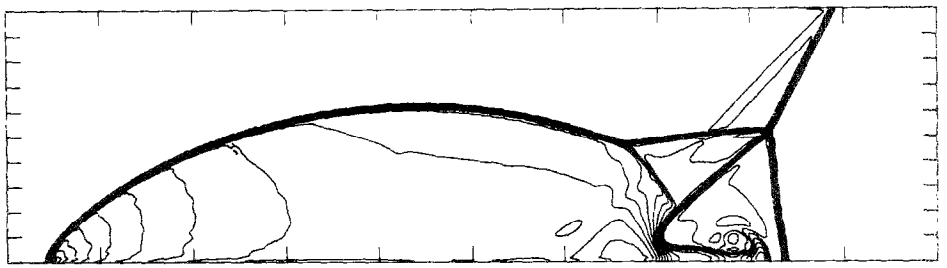

FIG. 9. Numerical solution of ramp problem using the method described in Section 4. The mesh is a rectangular mesh of 400×100 zones, with the reflecting wall beginning 20 mesh lengths from the lower left corner. $\Delta x = \Delta y = \frac{1}{120}$, and the time shown is $t = 0.2$; thus this calculation corresponds to the finest grid results in [22]

sides, there is no difficulty with uncontrolled diffusion of the discontinuities. This is in contrast to the results obtained with first-order upwind methods, where steady shocks remain quite sharp if the transition is supersonic/subsonic, but which spread over many zones if the transition is supersonic/supersonic. Indeed, the main difficulty for the present method is to ensure that the shocks are broad enough so that sufficient dissipation occurs across the shock, as was the case with the operator split second-order methods.

The second test problem is unsteady shock reflection problem. A planar shock is incident on an oblique surface, with the surface at a 30° angle to the direction of propagation of the shock (Fig. 8). The fluid in front of the shock has zero velocity, and the shock Mach number is equal to 10. The solution to this problem is self-similar, with U a function of (x, y, t) only in the combination $(x/t, y/t)$. In Fig. 9, we show the results of calculation of this test problem performed with the present unsplit second-order method; in Fig. 10, the corresponding results obtained with the operator split method. The results of the two calculations are essentially identical, supporting the assertion that the unsplit method has the same resolution as the corresponding operator split method. However, a considerable degree of care was required in the unsplit scheme for this to be the case. The choice of (2.15), rather

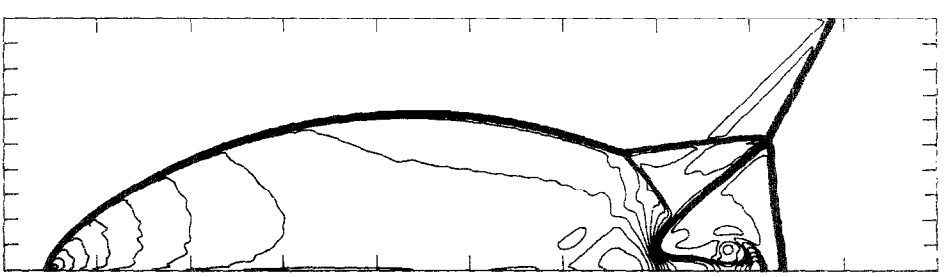

FIG. 10. Numerical solution of ramp reflection problem, using operator split method, with numerical parameters the same as for Fig. 9.

than (2.13), in calculating the transverse derivative in the predictor step is essential; otherwise, one obtains considerably lower resolution in the jet along the wall in the double Mach region. The accuracy in the double Mach region is also sensitive to the reflecting boundary conditions. The former difficulty has no analogue in the operator split method; as for the latter problem, the operator split method gives the same results which much simpler boundary conditions. Finally, the multidimensional flattening algorithm given by (4.10) was required to eliminate low-amplitude noise behind the shocks, whereas the operator split algorithms required only the 1-dimensional flattening algorithm in [7] to be applied in each sweep.

6. Discussion and Conclusions

In this paper, we have derived explicit second-order Godunov-type methods in two space variables by using the wave propagation properties for multidimensional hyperbolic equations and by limiting some of the second-order terms to suppress oscillations. The calculations in Section 5 indicate that we have been successful in the goal stated in the Introduction of producing an algorithm with comparable performance to the operator split second-order Godunov methods, at a comparable cost. In retrospect, this is not surprising, since the multidimensional algorithm consists of combinations of the 1-dimensional operators which appear in the operator split schemes. In particular, the same Riemann problems appear in the present method as in the operator split methods, since in the former case averaging the solution to the characteristic form of the equations over a zone edge provides, via (2.4), a natural choice of a direction in which to project the multidimensional equations for solving the Riemann problem. However, there are differences between the present algorithms and the operator split approach. The algorithms discussed here are somewhat more expensive, requiring twice as many solutions to the Riemann problem as the corresponding operator split algorithm. Since the cost of solving the Riemann problem for a polytropic equations of state constitutes half the cost of the calculation in one dimension [6], this leads to an algorithm which takes 50% more time than the operator split algorithm. In the regular reflection problem, the vectorized implementation on the Cray 1 advanced about 24,000 zones by one time step in each cpu second, consistent with this estimate and the timing figures for the corresponding 1-dimensional algorithm given in [6]. Also, the multidimensional algorithms appear to be more sensitive to various details of the implementation, requiring a greater degree of care, such as for the reflecting boundary conditions (4.11)–(4.12), and for the flattening algorithm (4.10).

There are a number of straightforward applications and extensions of the methods described here. It is possible to introduce quadratic interpolants, as in [7], to evaluate $\hat{U}$ in the predictor step in order to improve the resolution of linear discontinuities by means of contact detection and steepening. Conservation laws for which the fluxes have an explicit spatial dependence, such as for incompressible multiphase flow in porous media, can be easily treated using similar techniques to

the ones used for the general quadrilateral meshes. The treatment of a general equation of state via the techniques in [6] is accomplished by introducing an additional transport equation for $\gamma = p/\rho e + 1$ for use in the predictor step for the transverse derivatives. Thus introduces some additional complication into the method, which is more than offset by the fact one need only evaluate the equation of state once per zone per time step.

There are some problems for which the formalism given here is attractive, but for which the extensions are not entirely straightforward. One of these is the extension of this method for calculation of problems in Lagrangian coordinates in two dimensions. The difficulty here is that the motion of the grid must be obtained from the solution itself; unlike in one dimension, neither the solution nor the fluxes are defined at the corners of the mesh, where it is most natural to specify the motion of the grid. Consequently, some form of averaging of the velocities must be introduced in order to move the grid, but one which does not degrade the resolution of the method [17]. Finally, there is the question of the extension of these ideas to three dimensions. If we just take as our advection algorithm the 3-dimensional analogue of (1.2), we arrive at an algorithm for systems which satisfies the properties (1)–(3) in the Introduction, but requires 12 solutions to the Riemann problem

the fact that for each coordinate direction in three dimensions, the analogue of the predictor step for the transverse derivatives (2.9) requires a calculation comparable to the full 2-dimensional calculation described in this paper. However, if we are willing to relax the third requirement somewhat, we obtain an algorithm which requires only 6 solutions to the Riemann problem by using the extension of donor-cell differencing to systems to evaluate the transverse derivatives in the predictor step; equivalently, we would be ignoring the contributions due to transport from zones offset by one mesh length in all three directions, which correspond to third-order terms in the truncation error. In both cases, we would obtain algorithms which, for 2-dimensional problems aligned with one of the mesh directions, give identical results to the algorithms described in this paper. The question as to what the appropriate formulation is for problems in three dimensions is undoubtedly problem dependent, and probably can be resolved only by numerical experiments.

REFERENCES

1. *Methods of Computational Physics*, Vol. 3, edited by B. Alder and S. Fernbach (Academic Press, New York, 1964).
2. M. BERGER AND P. COLELLA, Lawrence Livermore National Laboratory Report UCRL-97196; *J. Comput. Phys.* **82**, 64 (1989).
3. I.-L. CHERN AND P. COLELLA, Lawrence Livermore National Laboratory Report UCRL-97200, *J. Comput. Phys.*, in press.
4. P. COLELLA, *SIAM J. Sci. Stat. Comput.* **3**, 76 (1982).
5. P. COLELLA, *SIAM J. Sci. Stat. Comput.* **6**, 107 (1985).

6. P. COLELLA AND H. M. GLAZ, *J. Comput. Phys.* **59**, 264 (1985).
7. P. COLELLA AND P. R. WOODWARD, *J. Comput. Phys.* **54**, 174 (1984).
8. R. COURANT AND D. HILBERT, *Methods of Mathematical Physics*, Vol. II (Interscience, New York, 1963).
9. S. EIDELMAN, P. COLELLA, AND R. P. SHREEVE, *AIAA J.* **22**, 1609 (1984).
10. B. A. FRYXELL, P. R. WOODWARD, P. COLELLA, AND K.-H. WINKLER, *J. Comput. Phys.* **63**, 283 (1986).
11. S. K. GODUNOV, A. V. ZABRODYN, AND G. P. PROKOPOV, *USSR Comput. Math. Math. Phys.* **1**, 1187 (1961).
12. J. B. GOODMAN AND R. LEVEQUE, *Math. Comput.* **45**, 15 (1985).
13. A. HARTEN, *J. Comput. Phys.* **49**, 357 (1983).
14. A. HARTEN, "On Second Order Accurate Godunov-type Schemes," 1982 (unpublished).
15. C. H. LAI, G. S. BODVARSSON, AND P. A. WITHERSPOON, "Numerical Studies of Silica Precipitation/Dissolution," Lawrence Berkeley Laboratory Earth Sciences Division, 1985 (unpublished).
16. R. D. RICHTMYER AND K. W. MORTON, *Finite Difference Methods for Initial-Value Problems* (Interscience, New York, 1967).
17. J. S. SALTZMAN AND P. COLELLA, Los Alamos National Laboratory Report LAUR-85-678, 1985 (unpublished).
18. G. R. SHUBIN AND J. B. BELL, *Comput. Meth. Appl. Mech. Eng.* **47**, 47 (1984).
19. J. T. STEGER, *AIAA J.* **16**, 679 (1978).
20. B. VAN LEER, *J. Comput. Phys.* **23**, 276 (1977).
21. B. VAN LEER, in *Computing Methods in Applied Sciences and Engineering VI*, edited by R. Glowinski and J.-L. Lions (North-Holland, Amsterdam, 1984), p. 493.
22. P. R. WOODWARD AND P. COLELLA, *J. Comput. Phys.* **54**, 115 (1984).
23. H. C. YEE, R. F. WARMING, AND A. HARTEN, in *Proceedings, 8th International Conference on Numerical methods in Fluid Dynamics*, Lecture Notes in Physics Vol. 141 (Springer-Verlag, New York/Berlin, 1982), p. 547.
24. S. T. ZALESAK, *J. Comput. Phys.* **31**, 335 (1979).
25. "Proceedings, Sixth AIAA Computational Fluid Dynamics Conference, Danvers, MA, June, 1983."