JOURNAL OF COMPUTATIONAL PHYSICS 71, 194–223 (1987)

Grid Adaptive Algorithms for the Solution of the Euler Equations on Irregular Grids

J. PIKE

Royal Aircraft Establishment, Bedford, England

Received October 16, 1984; revised August 22, 1986

The effect of grid irregularity on the accuracy of algorithms for one-dimensional unsteady flow is investigated. It is shown how standard second order accurate algorithms for equally spaced grids may be extended to act as first order accurate algorithms on an irregular grid. The loss of accuracy on irregular grids is found to cause a significant reduction in the quality of the results. To obtain solutions comparable in quality with second order accurate algorithms on equally spaced grids, a class of second order accurate algorithms is derived, which are conservative and compatible on randomly spaced grids. They are shown to give solutions to the equations similar in quality to those obtainable on equally spaced grids.

1. INTRODUCTION

In the computation of finite difference solutions to the Euler equations for engineering applications, it is important that the grids are easily produced. There has been little systematic investigation of the effect of irregularity in the grid spacing on the accuracy of the solution [1, 2], although recently [14–16] the problem has been receiving more attention. In practice the grids are made as smooth as possible, a process which is time consuming for all but the simplest test problems. One technique, which has been applied to keep the grid smooth, is to superimpose or embed patches of a finer grid within the main grid. This technique, however, introduces sudden changes in grid spacing at the boundaries of the patch, which usually require special handling by the algorithm if errors or reflections from this grid boundary are to be avoided. These sudden changes in grid spacing are a mainly one-dimensional problem in a direction normal to the edge of the patch. Indeed, as most finite difference algorithms for two and three dimensions use successive "time-split" application of a one-dimensional algorithm in each dimension, improvements in the one-dimensional algorithm to handle irregularities in the grid in the direction of application of the algorithm are related directly to the multi-dimensional problem. The object of the present investigation is to discover whether

0021-9991/87 $3.00

results for one-dimensional flows on irregular grids can be made as good as results on equally spaced grids.

The algorithms are developed using the linear wave equation as a model equation, i.e.,

$$U_t + CU_x = 0, \tag{1}$$

where C is a positive constant. The Euler equations in conservation form can be written

$$\mathbf{U}_t + \mathbf{F}_x + \mathbf{G}_y + \mathbf{H}_z = 0, \tag{2}$$

where

$$\mathbf{U} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho e \end{bmatrix}, \quad \mathbf{F} = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uw \\ u(\rho e + p) \end{bmatrix}, \quad \mathbf{G} = \begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ \rho vw \\ v(\rho e + p) \end{bmatrix}, \quad \mathbf{H} = \begin{bmatrix} \rho w \\ \rho uw \\ \rho vw \\ \rho w^2 + p \\ w(\rho e + p) \end{bmatrix}. \tag{3}$$

By applying linearized eigenvector analysis and time-splitting techniques [3], the solution of the Euler equations can be constructed by the local superposition of solutions to Eq. (1).

The algorithms for solving Eq. (1) on one-dimensional irregular grids are built in a step-by-step manner, to obtain an algorithm which will represent both smooth and discontinuous features of the flow with high accuracy. When these algorithms are used to solve the Euler equations by a method based on local superposition of linearised solutions, an error in any one of the superimposed solutions may influence the others and cause a large loss of accuracy for the system. Consequently, considerable effort is applied here to obtaining solutions which are free from local oscillations caused by flow discontinuities, grid irregularities, or a combination of both.

In the remainder of the Introduction we consider in more detail some of the types of error that can be introduced by the algorithm. Suppose we apply the standard first order accurate upwind algorithm to the linear wave equation (Eq. (1)) on a grid with random spacing as shown in Fig. 1. The algorithm can be written (for the case $C > 0$) as

$$U' = U_i - C\,\Delta t \frac{U_i - U_{i-1}}{X_i - X_{i-1}}, \tag{4}$$

where U_i is the value of U at $X = X_i$ at time t, and U' is the value of U_i at time $t + \Delta t$. The application of this algorithm to compute a flow on a random grid gives the results shown in Fig. 4(a), where the grid is indicated at the top of the figure. The initial data for the computation is the V data shown in Fig. 5(a). The exact solution of Eq. (1) for this data is shown in Fig. 5(b), where the V can be seen to be

propagating to the right at velocity C without changing shape. When we compare the solution in Fig. 4(a) with the exact solution, we see that the amplitude of the V has been greatly reduced and the solution is locally ragged.

A second drawback to the upwind algorithm of Eq. (4) on an irregular grid, is that it does not conserve the total amount of the flow variable U. A conservative algorithm would ensure that the area of the V in Fig. 4(a) remainded constant with time. Perhaps more importantly, it would also have ensured that any discontinuities in the solution would propagate at the correct rate for their strength. To determine the condition for conservation on an irregular grid we need to estimate the total amount of U in a flow which is defined only at discrete data points. To make this estimate we assume that the mean volume of U between any two data points is equal to the average value of U at those data points, i.e.,

$$\bar{U}_i = (U_i + U_{i+1})/2. \tag{5}$$

Using this definition, the change in the amount of U from a change δU_i at X_i is given by $\delta U_i(X_{i+1} - X_{i-1})/2$. The same expression is obtained if we assume that each data point represents the average value of U over the region of space closest to the data point. Thus for the algorithm to be conservative we need

$$\sum \delta U_i(X_{i+1} - X_{i-1}) = 0. \tag{6}$$

For the upwind algorithm δU_i is $U^i - U_i$ from Eq. (4), hence the algorithm is conservative if ΔX is constant.

To make Eq. (4) conservative for an irregular grid we must replace $X_i - X_{i-1}$ with $(X_{i+1} - X_{i-1})/2$. However, this introduces an error proportional to $(X_{i+1} - X_i)/(X_i - X_{i-1}) - 1$ into the finite difference approximation, so that the algorithm is no longer even first order accurate on an irregular grid. We describe this condition as zero order accurate. The solution using the zero order accurate conservative algorithm with the same initial V data on a random grid is shown in Fig. 6(c) to be similar in quality to the first order accurate non-conservative solution shown in Fig. 4(a).

To combine first order accuracy with conservation we need to use more data points. If we allow the right-hand side of Eq. (4) to influence U^{i-1} as well as U^i, we can obtain a first order accurate conservative algorithm. However, there is then a tendency for "wiggles" to appear in the solution in a similar way to those which typically appear in second order accurate algorithms on equally spaced grids. An example is shown in Fig. 4(b) for a Lax–Wendroff-type algorithm, which is conservative and first order accurate on a random grid. To remove the wiggles, a similar technique is used to that for the second order accurate algorithms on equally spaced grids, by introducing the Total Variation Decreasing (TVD) criterion [4–8]. The result is shown in Fig. 7(c). To improve the accuracy of the solution further, we need an additional point in the algorithm and a more elaborate process to ensure the TVD criterion is satisfied. This has been implemented and leads to the results of Fig. 8.

The various steps in the development of the algorithms mentioned above and their application to the linear wave equation, Burgers equation and the Euler equations is considered in Sections 2–5. In Section 2 we give formal definitions of accuracy and exhibit the constraints which these place upon an algorithm in conservation form. These constraints take the form of recurrence relations which relate the properties of the algorithms in neighbouring intervals. In Section 3 we investigate the local stability of these algorithms and find ways of solving the recurrence relations so that the stability criteria are always met. In Section 4 we show how to modify the algorithms so as to avoid non-physical over-shoots in the solution. The results for the test problems are to be found in Section 5.

2. Conservation and Accuracy

2.1. *Definitions and Notation*

Consider a set of unequally spaced grid points along the X axis as shown in Fig. 1, where the intervals between grid points ($\Delta X_i = X_{i+1} - X_i$) may vary with i. The initial value of the data function U at $t = 0$ is $U(X_i) = U_i$. The analytic solution to the linear wave equation (Eq. (1)) for U after a time interval Δt is then

$$U^i = U(X_i - C\,\Delta t) = U(X_i - v_i\,\Delta X_i), \tag{7}$$

where v_i ($= C\,\Delta t/\Delta X_i$) is the local Courant number. To obtain a definition of first and second order accurate algorithms, we expand this solution in a Taylor series about X_i; that is,

$$U^i = U_i - v_i\,\Delta X_i U_i' + \tfrac{1}{2} v_i^2 (\Delta X_i)^2\, U_i'' + O(\Delta X_i)^3. \tag{8}$$

Then the change at X_i in time Δt is given by

$$\delta U_i = U^i - U_i = -v_i\,\Delta X_i U_i' + \tfrac{1}{2} v_i^2 (\Delta X_i)^2\, U_i'' + O(\Delta X_i)^3. \tag{9}$$

First order accurate algorithms are defined to be those which satisfy this equation to $O(\Delta X)^2$ and second order accurate algorithms to $O(\Delta X)^3$.

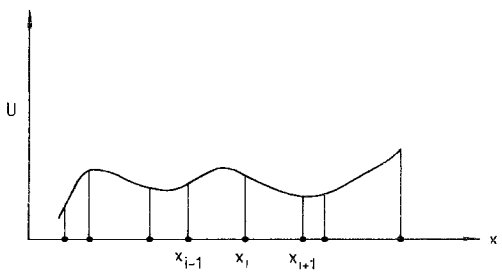

FIG. 1. An irregularly spaced grid of points.

We also introduce the notion of first and second order exact algorithms. These are algorithms for which the solution is *exact* for linear and quadratic data, respectively. A first order exact algorithm differs from a first order accurate algorithm in that the ΔX term on the right-hand side of Eq. (9) is represented exactly for an exact algorithm but may differ from it by $O(\Delta X)^2$ for a first order accurate algorithm. Note, however, that a first order exact algorithm is only first order accurate for general data, because the neglected third term on the right-hand side of Eq. (9) is $O(\Delta X)^2$. Similarly we define second order exact algorithms as the subclass of second order accurate algorithms which propagate quadratic data exactly.

The algorithms we seek are not only of prescribed accuracy (or exactness), but must also have the property of conserving the total amount of U in the sense defined in the introduction. It is convenient to introduce this latter requirement by considering the algorithms in what is known as "increment" form [9]. We compute in each X interval, the quantity

$$\Phi_i = -C\frac{\Delta t}{\Delta X_i}\Delta U_i = -v_i(U_{i+1} - U_i), \tag{10}$$

where Φ_i is interpreted as the local accumulation of the conserved variable U in the cell $X_i \leqslant X \leqslant X_{i+1}$ in time Δt. For values of Δt of $O(\Delta X)^2$ this accumulation in the cell may be represented by equal changes to U_i at each end of the cell; a process which for an equally spaced grid gives a central difference algorithm. For more practical values of Δt (i.e., $O(\Delta X)$), to maintain accuracy and stability it is necessary to introduce some upwind bias by distributing the larger part of the accumulation to the downwind end of the cell. Indeed to obtain second order accuracy on irregular grids, we see later that Φ_i needs to be distributed not only to the cell endpoints X_i and X_{i+1} but also to one other point, taken most naturally to be the next downwind point. For C positive (i.e., flow from left to right in Fig. 1) this point is X_{i+2} and the analysis which follows is based on this assumption. For C negative the equivalent point is X_{i-1} and a corresponding algorithm can be deduced from simple considerations of symmetry.

We shall assume that Φ_i is distributed to the points X_i, X_{i+1}, X_{i+2} with weights α_i, β_i, and γ_i, respectively, such that

$$\alpha_i + \beta_i + \gamma_i = 1. \tag{11}$$

Then the changes in U_i, U_{i+1}, and U_{i+2} which will conserve the total amount of U are given by

$$\alpha_i \Phi_i \frac{2\Delta X_i}{\Delta X_i + \Delta X_{i-1}} \tag{12}$$

$$\beta_i \Phi_i \frac{2\Delta X_i}{\Delta X_{i+1} + \Delta X_i} \tag{13}$$

and

$$\gamma_i \Phi_i \frac{2\Delta X_i}{\Delta X_{i+2} + \Delta X_{i+1}}, \tag{14}$$

respectively. That is the accumulated "mass" $\Delta X_i \Phi_i$ is distributed to the nodal points X_i, X_{i+1}, and X_{i+2}, causing a "density" change, which depends on the adjacent cell sizes. The total change in U_i in time Δt is given by adding the α, β, and γ contributions from adjacent cells, that is,

$$\begin{aligned}\delta U_i &= \frac{2\alpha_i \Delta X_i \Phi_i + 2\beta_{i-1} \Delta X_{i-1} \Phi_{i-1} + 2\gamma_{i-2} \Delta X_{i-2} \Phi_{i-2}}{\Delta X_i + \Delta X_{i-1}} \\ &= \frac{-2C \Delta t(\alpha_i \Delta U_i + \beta_{i-1} \Delta U_{i-1} + \gamma_{i-2} \Delta U_{i-2})}{\Delta X_i + \Delta X_{i-1}}.\end{aligned} \tag{15}$$

To assess the accuracy and exactness of this algorithm, it is compared with the Taylor expansion in Eq. (9). To expand Eq. (15) in terms of ΔX, we assume U can be represented in smooth regions of the flow by the polynomial form

$$U = \sum K_j (X - X_i)^j, \qquad j \geqslant 0, \tag{16}$$

where the K_j are constants. The value of U_i' in Eq. (9) is then K_1 and the values of ΔU are given by

$$\Delta U_i = U_{i+1} - U_i = K_1 \Delta X_i + K_2 (\Delta X_i)^2 + O(\Delta X)^3 \tag{17}$$

$$\Delta U_{i-1} = U_i - U_{i-1} = K_1 \Delta X_{i-1} - K_2 (\Delta X_{i-1})^2 + O(\Delta X)^3 \tag{18}$$

$$\Delta U_{i-2} = U_{i-1} - U_{i-2} = K_1 \Delta X_{i-2} - K_2 \Delta X_{i-2} (2\Delta X_{i-1} + \Delta X_{i-2}) + O(\Delta X)^3. \tag{19}$$

The first order exact condition is obtained by equating the terms $O(\Delta X)$ in Eqs. (9) and (15), using (17)–(19), to give

$$\alpha_{i+1} \Delta X_{i+1} + \beta_i \Delta X_i + \gamma_{i-1} \Delta X_{i-1} = \tfrac{1}{2}(\Delta X_{i+1} + \Delta X_i), \tag{20}$$

and for first order accuracy, this equation must be true to $O(\Delta X)^2$. Similarly for the algorithm to be second order exact, Eq. (9) and (15) must be matched to $O(\Delta X)^2$. Thus a second order exact algorithm has the additional constraint on α, β, and γ given by

$$\begin{aligned}&\alpha_{i+1} (\Delta X_{i+1})^2 - \beta_i (\Delta X_i)^2 - \gamma_{i-1} \Delta X_{i-1} (2\Delta X_i + \Delta X_{i-1}) \\ &\quad = -\tfrac{1}{2} C \Delta t (\Delta X_{i+1} + \Delta X_i).\end{aligned} \tag{21}$$

For second order accuracy Eqs. (20) and (21) need only be true to $O(\Delta X)^3$.

2.2. *First Order Algorithms*

We have shown in 2.1 that algorithms which are conservative and first order exact obey the relationships of Eqs. (11) and (20). Eliminating β_i from these equations we obtain

$$\alpha_{i+1}\,\Delta X_{i+1}-\alpha_i\,\Delta X_i-\gamma_i\,\Delta X_i+\gamma_{i-1}\,\Delta X_{i-1}=\tfrac{1}{2}(\Delta X_{i+1}-\Delta X_i). \tag{22}$$

This equation can be rearranged in "recurrence" form, i.e.,

$$\alpha_{i+1}\,\Delta X_{i+1}-\gamma_i\,\Delta X_i-\tfrac{1}{2}\Delta X_{i+1}=\alpha_i\,\Delta X_i-\gamma_{i-1}\,\Delta X_{i-1}-\tfrac{1}{2}\Delta X_i, \tag{23}$$

whence

$$\alpha_i\,\Delta X_i=C_1+\tfrac{1}{2}\Delta X_i+\gamma_{i-1}\,\Delta X_{i-1}, \tag{24}$$

where C_1 is a constant. This constant has the dimensions of length and can either be defined using a reference length ΔX_R or by using the length $C\Delta t$. We use whichever is convenient to obtain particular properties for the algorithm. It should be noted that C_1 only needs to be strictly constant for conservative first order exact algorithms. For conservative first order accurate algorithms C_1 may vary by $O(\Delta X)^2$ along the grid.

As an example consider first order irregular grid algorithms with the property that the algorithm is second order exact on an equally spaced grid. The conditions relating α, β, and γ for conservative, second order exact algorithms on an equally spaced grid can be obtained from (11) or (20), and (21) with ΔX constant as

$$\alpha=\tfrac{1}{2}(1-\nu)+\gamma \tag{25}$$

$$\beta=\tfrac{1}{2}(1+\nu)-2\gamma=1-\alpha-\gamma. \tag{26}$$

In this form, γ can be considered as a parameter describing the algorithm. For example, the Lax–Wendroff, Fromm and third order accurate algorithm are described by $\gamma=0$, $(\nu-1)/4$, and $(\nu^2-1)/6$, respectively.

For Eq. (24) to reduce to Eq. (25) when ΔX is constant, the value of C_1 needs to be $-\frac{1}{2}C\Delta t$. Substituting this value of C_1 in Eq. (24) we obtain

$$\alpha_i=\tfrac{1}{2}(1-\nu_i)+\gamma_{i-1}\,\Delta X_{i-1}/\Delta X_i. \tag{27}$$

Note also that Eq. (11) reduces to Eq. (26) for equally spaced grids, thus *any algorithm which is second order exact for equally spaced grids can be used as a first order exact algorithm for irregular grids, if Eqs.* (25) *and* (26) *are interpreted in the form of Eqs.* (27) *and* (11). A similar result extends second order *accurate* algorithms to provide first order accurate algorithms for irregular grids.

As an example, setting $\gamma_i = 0$ in Eqs. (11) and (27) gives the Lax–Wendroff algorithm for an irregular grid, i.e.,

$$\alpha_i = \tfrac{1}{2}(1 - \nu_i) \tag{28}$$

$$\beta_i = \tfrac{1}{2}(1 + \nu_i). \tag{29}$$

This algorithm is first order exact on an irregular grid, second order exact on an equally spaced grid and on a sufficiently smooth non-uniform grid it can be second order accurate. This algorithm has been considered previously in reference [10], together with the conditions required for stability and "monotonicity." These conditions are investigated for more general algorithms in later sections.

2.3. *Second Order Algorithms*

For an algorithm to be second order exact in X, Eq. (21) must be satisfied in addition to Eqs. (11) and (20), with relaxation to $O(\Delta X)^3$ in these inequalities for second order accuracy. Substituting for α_{i+1} and β_i in Eq. (21) from Eqs. (11) and (24), we obtain

$$\begin{aligned} &\gamma_i \Delta X_i(\Delta X_{i+1} + \Delta X_i) - \gamma_{i-1} \Delta X_{i-1}(\Delta X_i + \Delta X_{i-1}) \\ &\quad = -\tfrac{1}{2}(\Delta X_{i+1} + \Delta X_i)(2C_1 + C\,\Delta t + \Delta X_{i+1} - \Delta X_i). \end{aligned} \tag{30}$$

This equation can be written in "recurrence" form, i.e.,

$$\begin{aligned} &\gamma_i \Delta X_i(\Delta X_{i+1} + \Delta X_i) + \tfrac{1}{2}\Delta X_{i+1}(2C_1 + C\,\Delta t + \Delta X_{i+1}) \\ &\quad = \gamma_{i-1} \Delta X_{i-1}(\Delta X_{i-1} + \Delta X_i) + \tfrac{1}{2}\Delta X_i(2C_1 + C\,\Delta t + \Delta X_i) \\ &\quad = \tfrac{1}{2}C_2, \end{aligned} \tag{31}$$

where C_2 is a second constant (with dimensions of length squared) which needs to be defined. From Eqs. (11), (24), and (31), α_i, β_i, and γ_i can now be written as

$$\gamma_i = \frac{C_2 - 2C_1 \Delta X_{i+1} - (\Delta X_{i+1})^2(1 + \nu_{i-1})}{2\Delta X_i(\Delta X_i + \Delta X_{i+1})} \tag{32}$$

$$\alpha_i = \tfrac{1}{2} + C_1/\Delta X_i + \gamma_{i-1}(\Delta X_{i-1}/\Delta X_i) \tag{33}$$

$$\beta_i = 1 - \alpha_i - \gamma_i, \tag{34}$$

representing a class of second order exact algorithms which depend on the definition of C_1 and C_2. If second order accuracy rather than second order exactness is required, then, as for the first order case, small variations in C_1 and C_2 to $O(\Delta X)^2$ and $O(\Delta X)^3$, respectively, are permitted along the grid.

The values of C_1 and C_2 in Eq. (32) can be chosen to give the algorithm particular properties. We investigate definitions of C_1 and C_2 which further increase the accuracy of the algorithm and then in the next section show how this algorithm is

unsatisfactory because it can produce local instabilities. We have already seen how C_1 can be defined in terms of the lengths ΔX_R or $C\,\Delta t$. Since the constant C_2 has dimensions of length squared, it can be defined using any linear combination of the terms $(\Delta X_R)^2$, $C\,\Delta t\,\Delta X_R$, and $(C\,\Delta t)^2$.

For an algorithm to be third order accurate for the special case of an equally spaced grid with spacing ΔX_R, the values of C_1 and C_2 must be such that Eq. (32) reduces to the third order form for γ when $\Delta X_i \equiv \Delta X_R$, i.e.,

$$\gamma_i = (v_R^2 - 1)/6. \tag{35}$$

Particular definitions of C_1 and C_2 which achieve this reduction are

$$C_1 = -\tfrac{1}{2}C\,\Delta t \tag{36}$$

$$C_2 = ((\Delta X_R)^2 + 2(C\,\Delta t)^2)/3. \tag{37}$$

Other choices of C_1 and C_2 can be made which also cause Eq. (32) to reduce to Eq. (35). The expressions are more complicated, but some properties such as stability may be improved. Further investigation in this area might be rewarding.

The values of C_1 and C_2 of Eqs. (36) and (37) give an algorithm which is third order accurate for $\Delta X_i \equiv \Delta X_R$. In order to approach third order accuracy on a non-uniform grid, it would be desirable to vary ΔX_R from point to point in some way. However, any change in ΔX_R affects the value of C_2 defined in Eq. (37), and to retain second order accuracy the value of C_2 must not vary more rapidly along the grid than $O(\Delta X)^3$, This restricts the type of average that can be used for ΔX_R.

A practical difficulty associated with the variation of C_2 and similar adaptive strategies, is that it is difficult to ensure that the algorithm will not become unstable for some particular combination of data and grid spacing. This can introduce local oscillations in the flow and eventually upset the accuracy of the whole solution. We proceed therefore, to examine the stability of algorithms in an attempt to construct algorithms which are more robust.

3. Stability

The difficulty of analysing the stability of a linear equation on an irregular grid is similar to that of analysing a non-linear equation on an equally spaced grid, and to reduce this difficulty we adopt a similar policy. That is, we confine ourselves to rejecting algorithms which are unstable for local regions of equally spaced grid. This necessary condition for stability provides a restriction on the relationship between γ_i and the grid pitch $\Delta X_i/\Delta X_R$. Although we cannot prove that this condition is sufficient for stability on irregular grids, our experience has been that algorithms designed with this restriction are satisfactory.

On equally spaced grids the stability of three parameter algorithms has been

investigated previously [11]. The stable regions found in (γ, ν) space are shown in Fig. 2. Several well-known algorithms are plotted on Fig. 2(a) and we can see, for example, that the Lax–Wendroff algorithm ($\gamma = 0$) is stable for $-1 \leqslant \nu \leqslant 1$, and the upwind algorithm ($\gamma = \frac{1}{2}(\nu - 1)$) is stable for $0 \leqslant \nu \leqslant 2$.

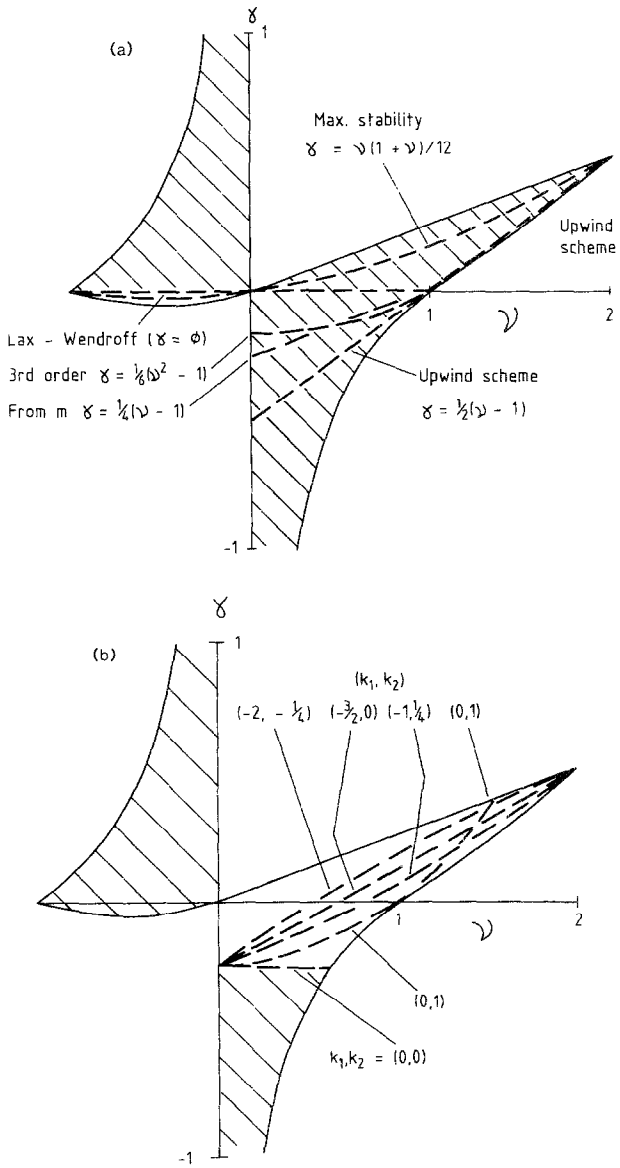

FIG. 2(a) Stable region (shaded) for algorithms on equally spaced grids. (b) Stability of second order algorithms which are "scale invariant" on grids of different pitch.

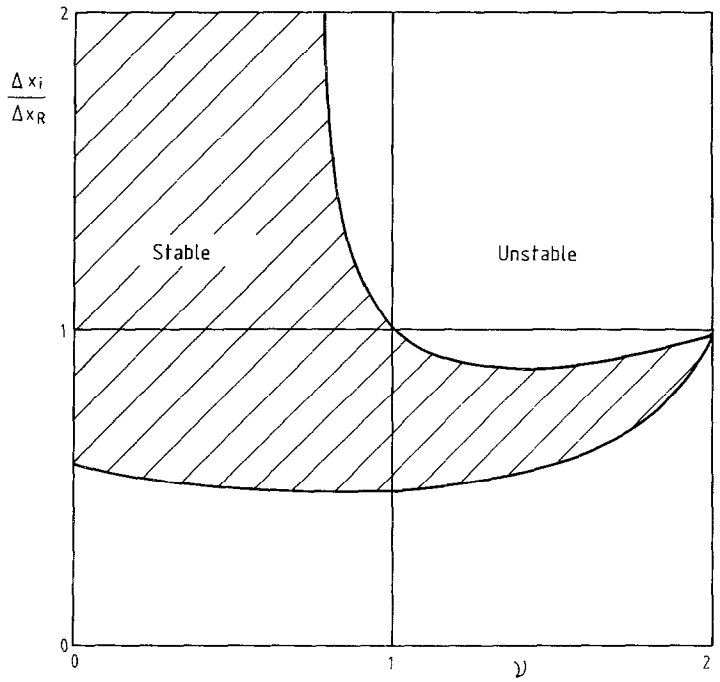

FIG. 3. Effect of grid pitch on the algorithm which is third order accurate when $\Delta x_i = \Delta x_R$.

For algorithms which are second order accurate on irregular grids, the ability to choose γ is restricted by Eq. (32). We have already seen how a second order accurate algorithm can be made third order accurate on an equally spaced grid by choosing C_1 and C_2 such that γ_i is given by

$$\gamma_i = (2v_i^2 - 3(\Delta X_R/\Delta X_i)^2 + 1)/12. \tag{38}$$

For any given value of grid scaling ratio $(\Delta X_i/\Delta X_R)$, we can plot Eq. (38) on Fig. 2 and establish its stability range. This stability range, as a function of scaling ratio, is shown in Fig. 3. The variation in the range of the stability due to the $(\Delta X_R/\Delta X_i)$ term in Eq. (38) makes it difficult to ensure that local instabilities will not occur on a random grid. We remove the dependence of the stability on the grid scaling ratio by defining C_1 and C_2 such that X_R does not appear, that is,

$$C_1 = \tfrac{1}{2}(K_1 - 1)C\,\Delta t, \qquad C_2 = K_2(C\,\Delta t)^2, \tag{39}$$

where K_1 and K_2 are constants. Substituting these values in Eq. (32) gives

$$\gamma_i = \frac{K_2 v_i^2\,\Delta X_i^2 - K_1 v_i\,\Delta X_i\,\Delta X_{i+1} - (\Delta X_{i+1})^2}{2\Delta X_i(\Delta X_i + \Delta X_{i+1})}. \tag{40}$$

We analyse the stability of these algorithms on equally spaced grids, when Eq. (40) has the form

$$= \tfrac{1}{4}(K_2 v^2 - K_1 v - 1). \tag{41}$$

Equation (41) can be plotted as a single curve on Fig. 2 for any particular choice of K_1 and K_2. For example, choosing $K_1 = -1$ and $K_2 = 0$ gives a second order accurate algorithm which reduces to Fromm's algorithm on any equally spaced grid. This algorithm and those for other values of K_1 and K_2 are shown on Fig. 2(b). Very extensive numerical studies would be needed to establish the properties of all these algorithms. We concentrate on an algorithm which has an increased stability range, whilst choosing γ to be zero at $v = 1$ to maintain maximum accuracy. The algorithm is given by $K_1 = 0$ and $K_2 = 1$ which has coefficients

$$\gamma_i = \frac{v_i^2 - (\Delta X_{i+1}/\Delta X_i)^2}{2(1 + \Delta X_{i+1}/\Delta X_i)} \tag{42}$$

$$\alpha_i = \frac{1 - v_i}{2}\left(1 - \frac{1 + v_i}{1 + \Delta X_{i-1}/\Delta X_i}\right) \tag{43}$$

$$\beta_i = 1 - \alpha_i - \gamma_i. \tag{44}$$

This algorithm is shown in Fig. 2(b) to be stable for $0 \leqslant v \leqslant 1.6$, and in Section 5 we will show numerical results from this algorithm when applied to the Euler equations. Other algorithms which have not been so thoroughly tested could also be of interest. For example, algorithms which are stable for $0 \leqslant v \leqslant 2$ can be obtained by putting $K_2 = \frac{3}{4} + \frac{1}{2}K_1$ with $\frac{1}{3} \leqslant -K_1 \leqslant 2$. One such algorithm with $K_2 = 0$ and $K_1 = -\frac{3}{2}$ is the straight line from $v = 0$, $\gamma = -\frac{1}{4}$ to $v = 2$, $\gamma = \frac{1}{2}$. Another such algorithm, which is close to Fromm's algorithm for small v, is given by $K_1 = -1$ and $K_2 = \frac{1}{4}$.

4. Compatibility

The accuracy and stability investigations were based on the assumption that the solution remains smooth and differentiable. However, hyperbolic differential equations permit discontinuous solutions, and the development and propagation of these discontinuities may be an important feature of the solution. If the solution is advanced using algorithms based on smooth-data concepts, the discontinuities may emit spurious oscillations or "wiggles" into the surrounding flow (see Fig. 4). A way to remove these errors is to modify the algorithm according to the data to be processed, such that discontinuous data prompt the use of an algorithm more suited to discontinuities. That is, we make the algorithm compatible with the data.

For equally spaced grids considerable effort has been applied to this problem, and even for this simple case there are many matters of detail which are still to be

settled. The most successful methods depend upon monitoring the smoothness of the data. Van Leer [4] first suggested the use of the ratio of successive gradients in the data (i.e., $\Delta U_i/\Delta X$ and $\Delta U_{i-1}/\Delta X$) to assess when the algorithm should be modified. Later Roe [12] adopted this same ratio as a basis for the method of which the present work is a generalisation. A straightforward extension of Roe's algorithm to irregular grids is to set $\gamma = 0$, $\alpha_i = \frac{1}{2}(1 - v_i)$ and $\beta_i = \frac{1}{2}(1 + v_i)$. This algorithm, as we have already seen, is first order exact on irregular grids. However, for nonlinear data, oscillations will appear close to shock waves. We can suppress these oscillations by a simple application of the compatibility technique, which consists of reducing the value of α whilst retaining $\beta = 1 - \alpha$. This straightforward approach however, cannot lead to an algorithm which is second order accurate on irregular grids. To achieve second order accuracy we shall need $\gamma \neq 0$ and it will be necessary for compatibility to include mechanisms which reduce the value of γ as well as α.

To prevent new oscillations forming in the flow under the action of the algorithm, it is sufficient to require that monotonic data remain monotonic. One way of ensuring this is to insist that the value of the data at any point after the application of the algorithm lies between its original value and that of its upstream neighbour. That is, with U^i the updated value of U_i, we require

$$(U^i - U_i)(U^i - U_{i-1}) \leqslant 0. \tag{45}$$

This requirement is stronger than that necessary to maintain monotonicity, but it is easier to analyse than trying to use a weaker restriction based on the updated value of the upstream point. Note that the inequality (45) also ensures that the Total Variation (TV) of the data, defined by

$$\mathrm{TV} = \sum_i |U_{i+1} - U_i|$$

continuously diminishes (see Harten [6]). This Total Variation Diminishing (TVD) property has the advantage, when dealing with non-linear equations, that shock waves can weaken. However, schemes which enforce TVD also tend to attenuate extreme values of the solution. If the data contains a sharp peak which should not be attenuated, it is probably necessary to use a better criterion than (45), but this is not a problem we are concerned with in this work.

Consider first the upwind algorithm with an (α, β, γ) weighting given by (0, 1, 0), that is, all the change in the cell in time Δt is distributed to the upwind point of the cell. It is clear that this algorithm will satisfy the compatibility condition (45) for equally spaced grids when $0 \leqslant v \leqslant 1$ and similarly for irregular grids when $0 \leqslant v_i \leqslant 1$. Then an algorithm $(\alpha_i, \beta_i, \gamma_i)$ which violates the compatibility condition can always be changed to a compatible algorithm by reducing α_i and γ_i sufficiently. However, the upwind algorithm is only first order accurate for irregular grids. Hence, for algorithms of higher order accuracy, the imposition of the compatibility constraint

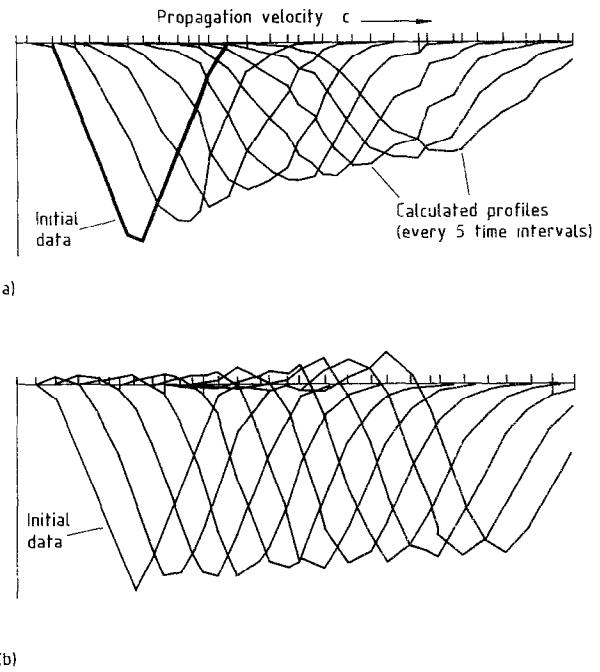

FIG. 4. The propagation of initial "V" data on a random grid for the linear wave equation. (a) Upwind algorithm (Eq. (4)), (b) Lax–Wendroff (first order, conservative).

must be expected to cause a loss of accuracy. Thus it is essential only to modify the α_i and γ_i to satisfy compatibility where it is necessary.

To simplify the problem, we initially consider the compatibility restriction on α_i for a fixed value of $\gamma_i = 0$. We suppose that the compatibility condition requires that α_i is reduced by some factor λ_i where $0 \leqslant \lambda_i \leqslant 1$. Then the total change in U_{i+1} in time Δt is given, as for Eq. (15), by

$$\delta U_i = \frac{-2C\,\Delta t(\lambda_i \alpha_i \Delta U_i + \beta_{i-1}\Delta U_{i-1} + (1-\lambda_{i-1})\alpha_{i-1}\Delta U_{i-1})}{\Delta X_i + \Delta X_{i-1}}, \tag{46}$$

where the final term in the numerator comes from the compatibility condition applied at X_{i-1}. For ease of analysis, we restrict ourselves first to monotonic increasing data. The inequalities to be satisfied are then from Eq. (45),

$$U^i - U_i \leqslant 0 \leqslant U^i - U_{i-1} = U^i - U_i + \Delta U_{i-1}. \tag{47}$$

Substituting for $U^i - U_i$ $(= \delta U_i)$ from Eq. (46) we obtain

$$\lambda_i \alpha_i \Delta U_i + \beta_{i-1}\Delta U_{i-1} + (1-\lambda_{i-1})\alpha_{i-1}\Delta U_{i-1} \geqslant 0 \tag{48}$$

and

$$\lambda_i\alpha_i\Delta U_i+\beta_{i-1}\Delta U_{i-1}+(1-\lambda_{i-1})\alpha_{i-1}\Delta U_{i-1}$$
$$\leqslant \Delta U_{i-1}(\Delta X_i+\Delta X_{i-1})/2C\,\Delta t. \tag{49}$$

These inequalities can be further simplified by using Eq. (11) with $\gamma_i=0$ to give

$$(\lambda_{i-1}\alpha_{i-1}-1)\,\Delta U_{i-1}\leqslant\lambda_i\alpha_i\,\Delta U_i\leqslant\lambda_{i-1}\alpha_{i-1}\,\Delta U_{i-1}$$
$$+\tfrac{1}{2}(1/v_i+1/v_{i-1}-2)\,\Delta U_{i-1}. \tag{50}$$

For $v_i v_{i-1}\leqslant 1$ the left hand inequality is satisfied for all $0\leqslant\lambda_{i-1}\leqslant 1$ and the [illegible]

$$\lambda_i=\text{minimum}\{1,\,(1/v_i+1/v_{i-1}-2)\,\Delta U_{i-1}/2\alpha_i\,\Delta U_i\}. \tag{51}$$

For monotonic decreasing data the same result is obtained.

For data which is not monotonic (i.e., $\Delta U_i\,\Delta U_{i-1}<0$) we obtain by a similar process that the largest allowable value of λ_i is

$$\lambda_i=\text{minimum}\{1,\,-\Delta U_{i-1}/\alpha_i\,\Delta U_i\}. \tag{52}$$

Equations (51) and (52) are used to ensure compatibility when $\gamma=0$, as, for example, for the irregular grid extension of Roe's algorithm discussed earlier.

Compatibility conditions involving three point weightings $(\alpha_i, \beta_i, \gamma_i)$ are more complicated because the contributions to each point involve four compatibility factors: λ_i, λ_{i-1} for the α's and the corresponding factors μ_{i-1}, μ_{i-2} for the γ's. The total change at the point X_i is, as for Eq. (15),

$$\delta U_i=-2C\,\Delta t(\mu_{i-2}\gamma_{i-2}\,\Delta U_{i-2}+(1-\lambda_{i-1})\,\alpha_{i-1}\,\Delta U_{i-1}$$
$$+\beta_{i-1}\,\Delta U_{i-1}+(1-\mu_{i-1})\,\gamma_{i-1}\,\Delta U_{i-1}+\lambda_i\alpha_i\,\Delta U_i)/(X_{i+1}-X_{i-1}). \tag{53}$$

Again we consider first monotonic increasing data. The compatibility inequality (45) and Eq. (53) then give

$$0\leqslant\mu_{i-2}\gamma_{i-2}\,\Delta U_{i-2}+(1-\lambda_{i-1})\,\alpha_{i-1}\,\Delta U_{i-1}+\beta_{i-1}\,\Delta U_{i-1}$$
$$+(1-\mu_{i-1})\,\gamma_{i-1}\,\Delta U_{i-1}+\lambda_i\alpha_i\,\Delta U_i\leqslant\Delta U_{i-1}(\Delta X_i+\Delta X_{i-1})/2C\,\Delta t. \tag{54}$$

To extract simple results from this complicated condition, it is necessary to make some simplifying assumptions. If $v_i\leqslant 1$ we find in most cases that α_i is positive and γ_i is negative. A typical example is the second order algorithm of Eqs. (42)–(44). For this algorithm

$$\alpha_i=\frac{1-v_i}{2}\left(1-\frac{1+v_i}{1+\Delta X_{i-1}/\Delta X_i}\right)=\frac{1-v_i}{2}\left(1-\frac{\Delta X_i+C\,\Delta t}{\Delta X_i+\Delta X_{i-1}}\right)$$
$$\geqslant 0 \qquad \text{for} \quad v_i, v_{i-1}\leqslant 1, \tag{55}$$

and

$$\gamma_i = \frac{v_i^2 - (\Delta X_{i+1}/\Delta X_i)^2}{2(1 + \Delta X_{i+1}/\Delta X_i)} = \frac{C^2\,\Delta t^2 - (\Delta X_{i+1})^2}{2\Delta X_i(\Delta X_i + \Delta X_{i+1})}$$
$$\leqslant 0 \qquad \text{for} \quad v_{i+1} \leqslant 1. \tag{56}$$

and the assumption is justified, for this example. With γ_i negative and μ_i and ΔU_i positive, the righthand inequality of (54) is satisfied if

$$(1 - \lambda_{i-1})\,\alpha_{i-1}\,\Delta U_{i-1} + \beta_{i-1}\,\Delta U_{i-1} + \lambda_i \alpha_i\,\Delta U_i$$
$$\leqslant \Delta U_{i-1}(\Delta X_i + \Delta X_{i+1})/2C\,\Delta t. \tag{57}$$

This inequality has already appeared as Eq. (49) for compatibility, hence if we first apply compatibility tests to α using Eqs. (51) and (52), the critical inequality for compatibility will be the left-hand inequality of (54).

With α_i and ΔU_i positive and $0 \leqslant \lambda_i \leqslant 1$, the left hand inequality of (54) is satisfied if

$$\mu_{i-2}\gamma_{i-2}\,\Delta U_{i-2} + \beta_{i-1}\,\Delta U_{i-1} + (1 - \mu_{i-1})\,\gamma_{i-1}\,\Delta U_{i-1} \geqslant 0$$

which leads, after some argument, to

$$(1 - \alpha_{i+1})\,\Delta U_{i+1}/\Delta U_i \geqslant -\mu_i \gamma_i. \tag{58}$$

Hence we have for γ a relationship similar to Eq. (51) for α, that is, the largest value μ_i can take is

$$\mu_i = \text{minimum}\left\{1, \frac{1 - \alpha_{i+1}}{-\gamma_i}\frac{\Delta U_{i+1}}{\Delta U_i}\right\}. \tag{59}$$

The same result holds for monotonic decreasing data. At a maximum or a minimum in the data (i.e., $\Delta U_i \Delta U_{i-1} < 0$) the analysis is complicated, and to enhance the stability we put

$$\mu_i = \gamma_i = 0.$$

In view of the assumptions made in analysing the compatibility for γ, it was considered expedient to limit γ more strongly. We recall that the algorithm (0, 1, 0) is compatible with all data, and we therefore restrict γ_i by a factor at least equal to λ_i. The compatibility procedure can thus be summarized to first reducing α_i as necessary to conform to Eqs. (51) and (52), reducing γ_i by the same factor, then further reducing γ_i to conform to Eq. (59) or put $\gamma_i = 0$ if $\Delta U_{i+1}/\Delta U_i < 0$. The value to β_i is then calculated from Eq. (11) to preserve conservation. This compatibility process is shown to be highly effective in the results of the next section. The complete algorithm is summarized in Appendix A.

5. Test Problems for the Algorithms

The algorithms are tested on the linear wave equation, Burgers equation, and the one-dimensional Euler equation using initial data for which the exact solution is known. These equations form a set progressively increasing in complication from the linear wave equation

$$U_t + CU_X = 0, \tag{60}$$

where C is a constant, through the non-linear Burgers equation

$$U_t + UU_X = 0, \tag{61}$$

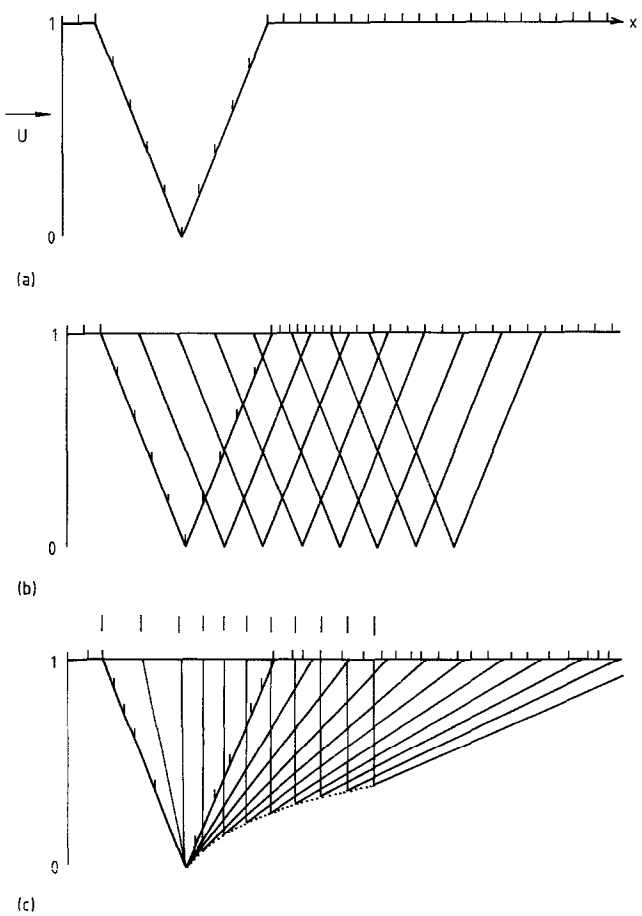

Fig. 5. Exact solutions and calculation grids for the linear wave equation and Burgers equation. (a) starting data, (b) analytic solution linear wave equation, (c) analytic solution Burgers equation.

to the non-linear system of Euler equations

$$\mathbf{U}_t + \mathbf{F}_X = 0, \tag{62}$$

where $\mathbf{U} = (\rho, \rho u, \rho v, \rho e)^T$ and $\mathbf{F} = (\rho u, \rho u^2 + p, \rho uv, \rho u(p + \rho e))^T$. The examples give the results shown in Figs. 5–14 in which increasing complexity in the equations is combined with increasing irregularity in the grid, for algorithms which are zero, first, and second order accurate. By comparing the results with an exact solution, we are able to identify the errors and discuss their reduction.

The initial data used for both the linear wave equation and Burgers equation is in the form of a "V" as shown in Fig. 5(a), and the exact solutions for the linear wave equation and Burgers equation are shown in Figs. 5(b) and (c), respectively, for $t > 0$. We see that the exact solution for the linear wave equation in Fig 5(b) remains identical to the solution at $t = 0$, except that the solution has been transported a distance Ct in the X direction. The time interval at which the solution is shown is $2.25\Delta X/C$, where ΔX is the distance between the tick marks in Fig. 5(a). Using Burgers equation, the same initial V data are transported in a similar manner, but become deformed in the process (Fig. 5(c)). The right-hand (expansion) side of the V gets progressively less steep whilst the left-hand (compression) side steepens until it becomes a discontinuity. This discontinuity continues to propagate to the right, but it slowly weakens as it interacts with the expansion side of the V. The locus of the minimum value at the base of the discontinuity is shown by the dotted line. It is important that the compatibility conditions of the algorithm do not inhibit the change in the calculated value of this minimum.

The vertical lines above the solution denote the X position of the left-hand extremity of the V after each time interval. Note that even though the solution is shown at equal time intervals ($\Delta t = 2.25X/U_0$), the distance moved by the left-hand edge of the disturbance is not constant for Burgers equation.

The algorithms are used to compute the test flows discussed above on grids of progressively increasing irregularity. The grid spacings used are indicated by the tick marks on the initial V data in Figs. 5(a)–(c) and subsequently along the line $U = 1$ in Figs. 6–11. Figure 5(a) shows an equally spaced grid with a constant interval ΔX and 10 intervals spanning the V. Figure 5(b) shows the same grid with a region of double density points embedded in the equally spaced grid, and Fig. 5(c) shows a grid with intervals which vary randomly in size between $\frac{1}{2}$ and $1\frac{1}{2}$ times those of the equally spaced grid. These grids are shown in the same order in Figs. 6–11. These six figures show the results from applying zero, first and second order accurate algorithms for propagation of the V data, first for the linear wave equation, then for Burgers equation.

The zero order accurate algorithm used has (α, β, γ) weightings of $(0, 1, 0)$. That is the whole of the increment generated in an interval is used to change the U value at the upwind end of the interval. The algorithm is conservative which ensures that the area of the V remains unchanged, but on equally spaced grids (when the algorithm is first order accurate) the algorithm is known to attenuate the high fre-

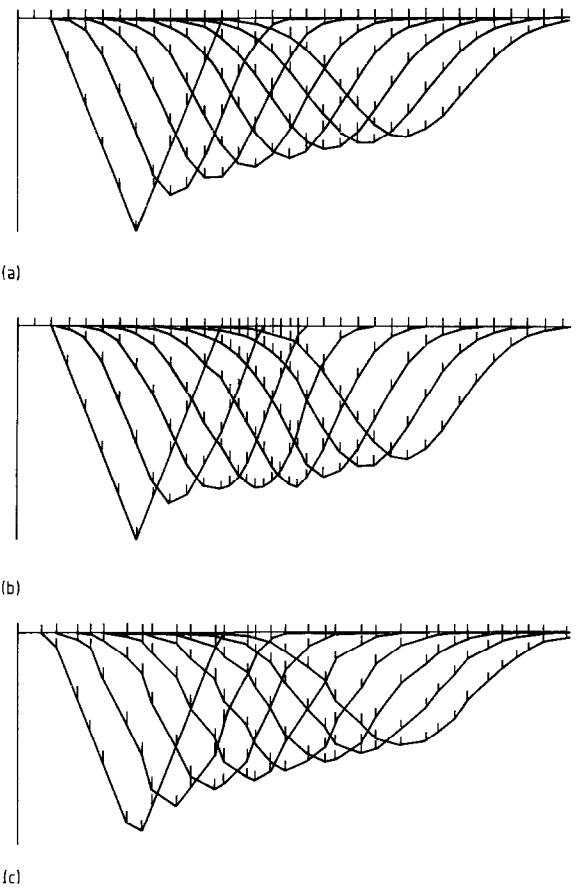

FIG. 6. Linear wave equation V test on a zero order accurate algorithm $(\alpha, \beta, \gamma) = (0, 1, 0)$. (a) Equally spaced grid, (b) embedded fine grid, (c) random grid.

quency components of the solution, so we should expect the shape of the V to change. This attenuation is well shown in Fig. 6(a), where the algorithm is used to propagate the initial V data on the equally spaced grid with $\Delta t = 0.45\Delta X/C$ (i.e., $\nu = 0.45$) with the results being shown every 5 time steps. The same value of Δt is used on the grids of Figs. 6(b) and (c), where we see that whatever the grid the effect of this algorithm is to attenuate the high frequency components so as to significantly degrade the solution. Similar results are obtained for other values of Δt less than $0.5\Delta X/C$. Above this value, in the smallest intervals in the grids of Figs. 6(b) and (c) the CLF stability limit is exceeded and the computation process becomes unstable.

The first order accurate algorithm which produces the results shown in Fig. 7, has (α, β, γ) weightings of $((1-\nu_i)/2, (1+\nu_i)/2, 0)$ as described by Eqs. (28) and

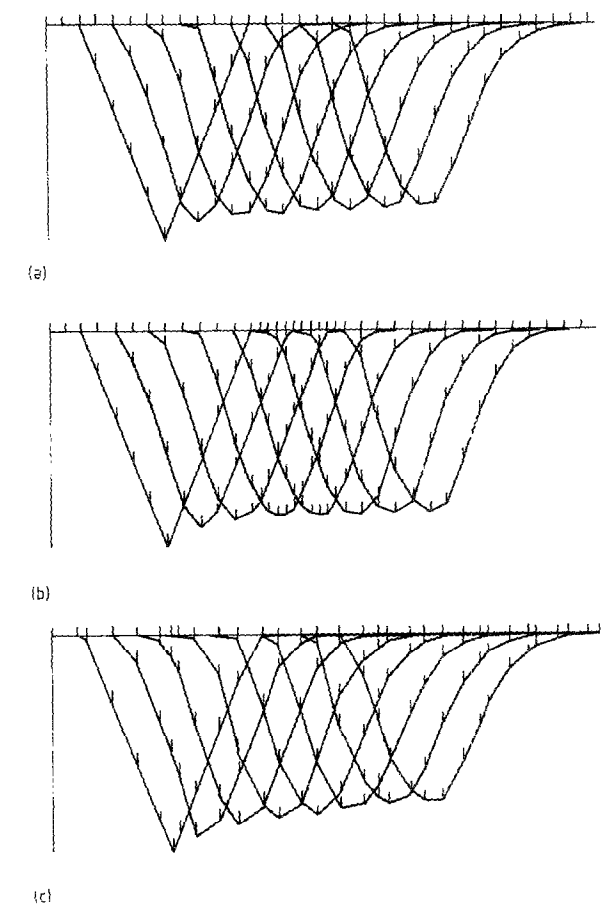

FIG. 7. Linear wave equation V test on a first order accurate algorithm $(\alpha, \beta, \gamma) = ((1 - \nu_i)/2, (1 + \nu_i)/2, 0)$. (a) Equally spaced grid, (b) embedded fine grid, (c) random grid.

(29), with compatibility conditions based on Eqs. (51) and (52). On an equally spaced grid this algorithm is the second order accurate Lax–Wendroff algorithm, with a compatibility requirement obtained from Eq. (51) with $\nu_i = \nu_{i+1}$. This restricts α to a maximum of $2\Delta U_{i-1}/\nu_i \Delta U_i$. The solution computed by this algorithm is a considerable improvement on that computed by the zero order accurate algorithm as can be seen by comparing Fig. 7 with Fig. 6. The attenuation of the point of the V is much less in Fig. 7, but there is some deterioration in this aspect of the solution between the equally spaced grid of Fig. 7(a) and the random grid of Fig. 7(c).

The second order accurate algorithm which produces the results shown in Fig. 8, has (α, β, γ) weightings given by Eqs. (42)–(44), with compatibility as described in Section 4 using Eqs. (51), (52), and (59). This algorithm has been constructed for

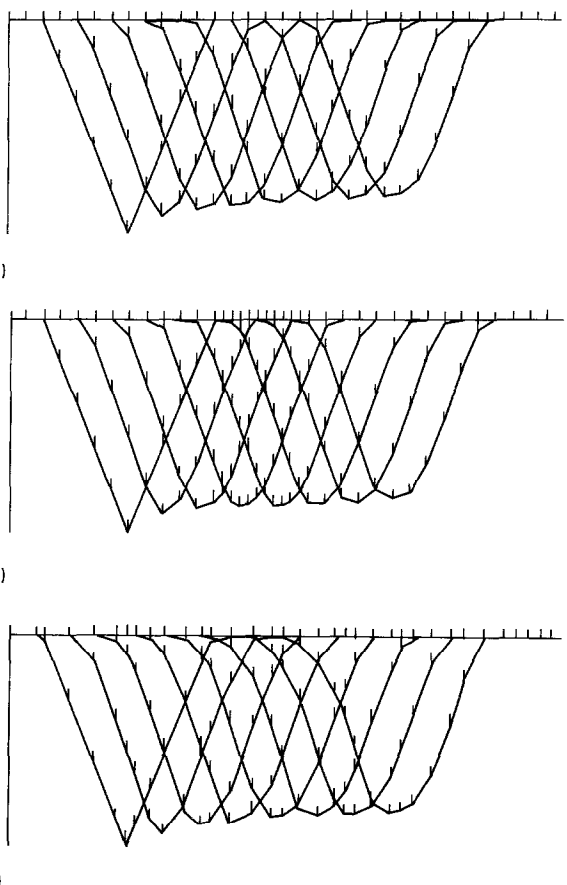

FIG. 8. Linear wave equation V test on a second order accurate algorithm (α, β, γ) from Eqs. (42)-(44). (a) Equally spaced grid, (b) embedded grid, (c) random grid.

its effectiveness on irregular grids and is still only second order accurate for equally spaced grids. It is to be expected then, that the solution on the equally spaced grid of Fig. 8(a) is similar in quality to the solution of Fig. 7(a). The main difference is that the introduction of the forward γ component seems to have improved the leading edge of the V. Note, however, that between the equally spaced grid of Fig. 8(a) and the random grid of Fig. 8(c) there is now no deterioration in the representation of the point of the V.

The effectiveness of the compatibility procedure, which permits algorithms to propagate discontinuities without producing errors is shown in Figs. 9–11 when the algorithms are applied to Burgers equation. The poor performance of the zero order accurate algorithm is shown again by the results of Fig. 9, where the discontinuity is

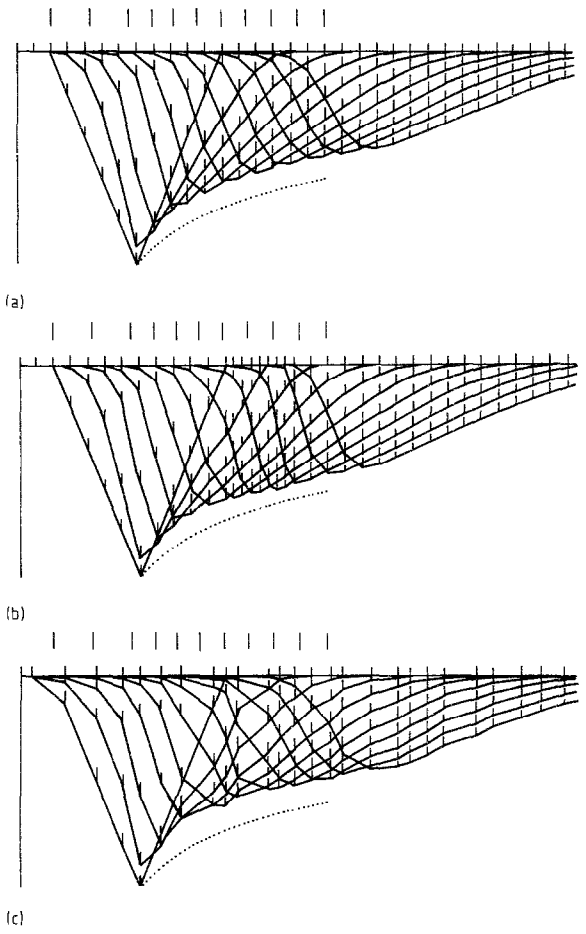

FIG. 9. Burgers equation V test on a zero order accurate algorithm $(\alpha, \beta, \gamma) = (0, 1, 0)$. (a) Equally spaced grid, (b) embedded grid, (c) random grid.

excessively smoothed in a similar manner to the results of Fig. 6 for the linear equation.

The solutions obtained by using the first order accurate algorithm are shown in Fig. 10 to be a considerable improvement on those of Fig. 9, with the discontinuity contained within at most three intervals and often occupying only the theoretical minimum of two intervals. A feature of the solution on the random grid (Fig. 10(c)) which causes some concern, however, is the appearance of "wiggles" in the expansion, particularly when the expansion gradient is steep. The effect appears to be associated with the non-linear nature of Burgers equation, because it is much less apparent for the linear case in Fig. 7(c). This type of error may be important for

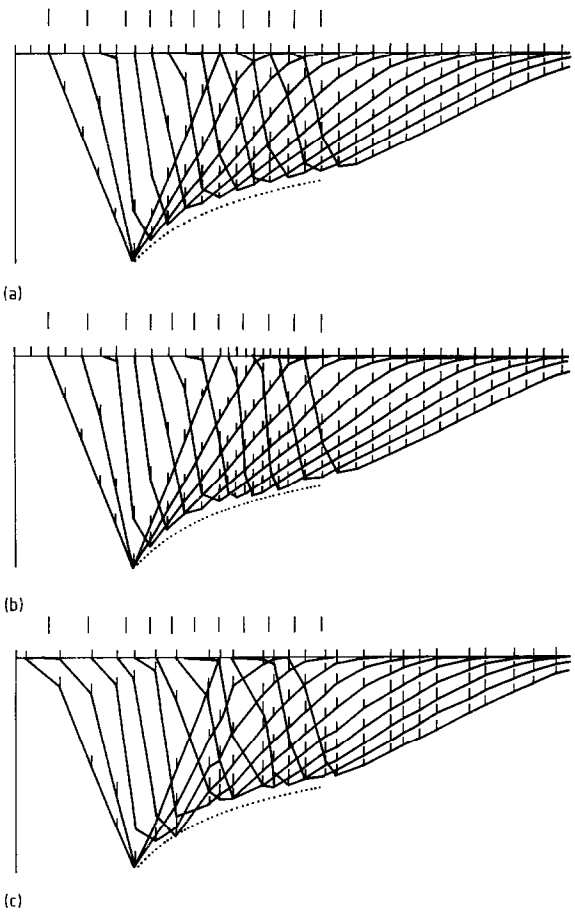

FIG. 10. Burgers equation V test on a first order accurate algorithm $(\alpha, \beta, \nu) = ((1-\nu_t)/2, (1+\nu_t)/2, 0)$. (a) Equally spaced grid, (b) embedded grid, (c) random grid.

non-linear systems of equations, because errors in one characteristic wave can cause errors in the other waves of the system, which may not even be propagating in the same direction.

The results from the second order accurate algorithm in Fig. 11, show the discontinuity is as well represented as in Fig. 10. Moreover, if we compare Fig. 11(c) with Fig. 11(a) there is almost no observable deterioration in the solution, even though the results shown in Fig. 11(c) are obtained using a random grid and those of Fig. 11(a) an equally spaced grid. With this objective achieved for Burgers equation the final test is to calculate a solution of the Euler equations.

When using the algorithms on a system of equations, it is more difficult to interpret the errors because of the increased complication of the problem. The test

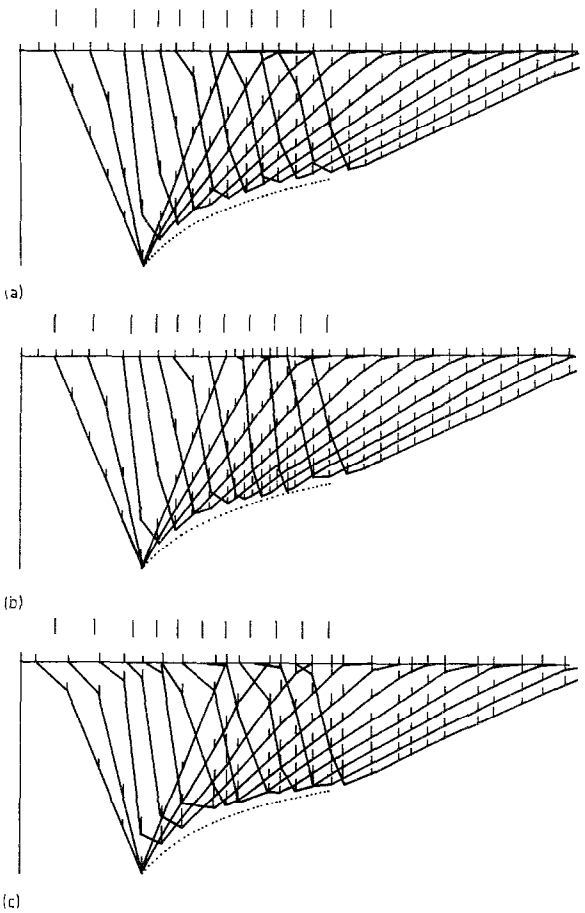

FIG. 11. Burgers equation V test on a second order accurate algorithm (α, β, γ) from Eqs. (42)–(44). (a) Equally spaced grid, (b) embedded grid, (c) random grid.

problem for the one-dimensional Euler equations is a problem suggested by Sod [13]. At $t=0$ the gas is stationary with a discontinuity in pressure (at $X=50$). After a time $24.5C/\Delta X$, where C is the speed of sound in the high pressure region, exact solutions for the density, velocity, and pressure are shown by the lines in Fig. 12, together with the internal energy $p/(\gamma-1)\rho$, which is often found to be particularly sensitive to errors. We see that the initial discontinuity has separated into an expansion fan, a contact discontinuity (near $X=63$) and a shock wave (near $X=76$). The flow remains undisturbed for $X<33$ and $X>76$. For equally spaced grids, accurate solutions can be calculated [3, 5] using Roe's method by resolving the equations into eigenvectors and applying the algorithm to each of the resulting characteristic waves. Here we apply the same technique, but use the three

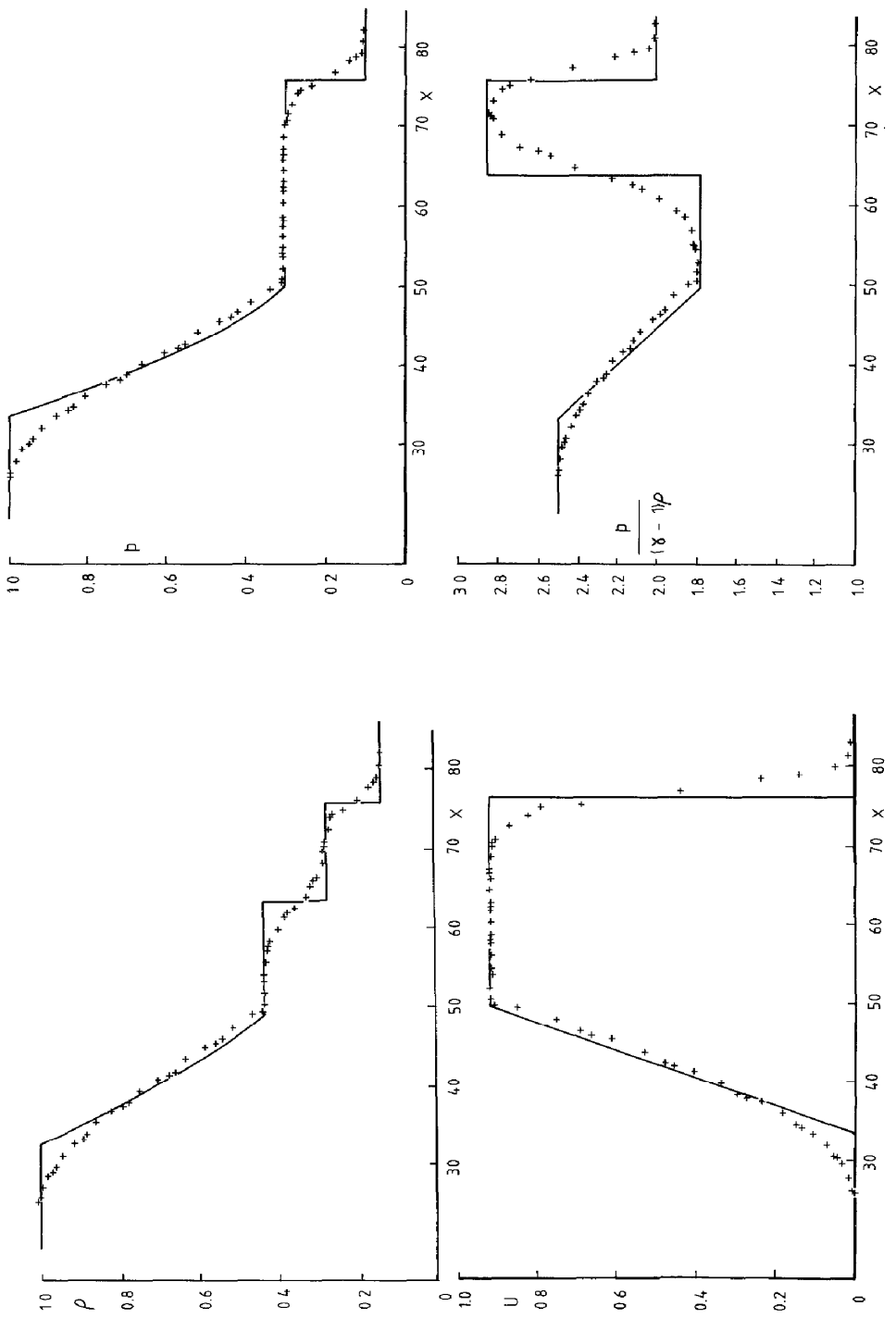

FIG. 12. Sod's problem solved with the zero order accurate algorithm $(\alpha, \beta, \gamma) = (0, 1, 0)$.

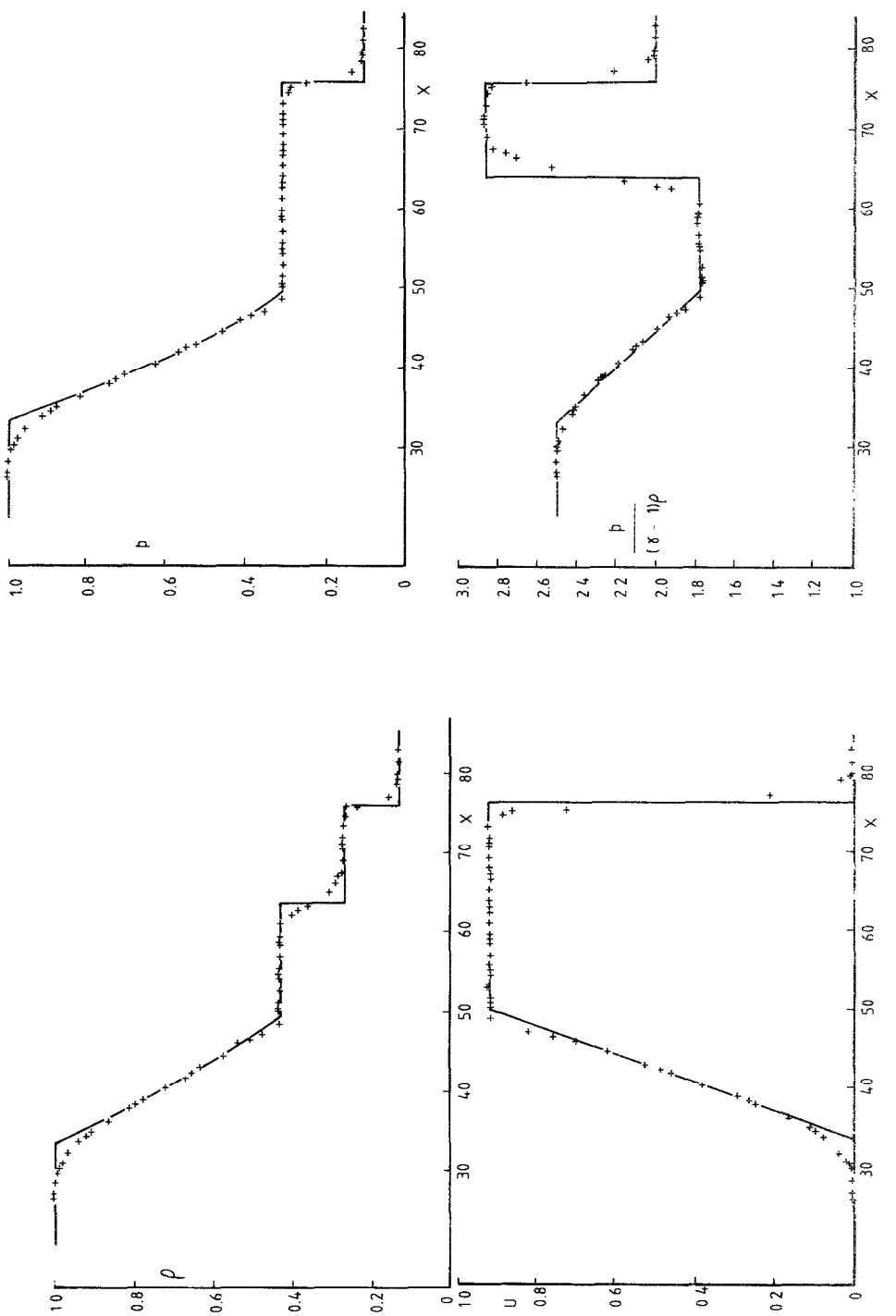

FIG. 13. Sod's problem solved with a first order accurate algorithm $(\alpha, \beta, \gamma) = ((1 - v_t)/2, (1 + v_t)/2, 0)$.

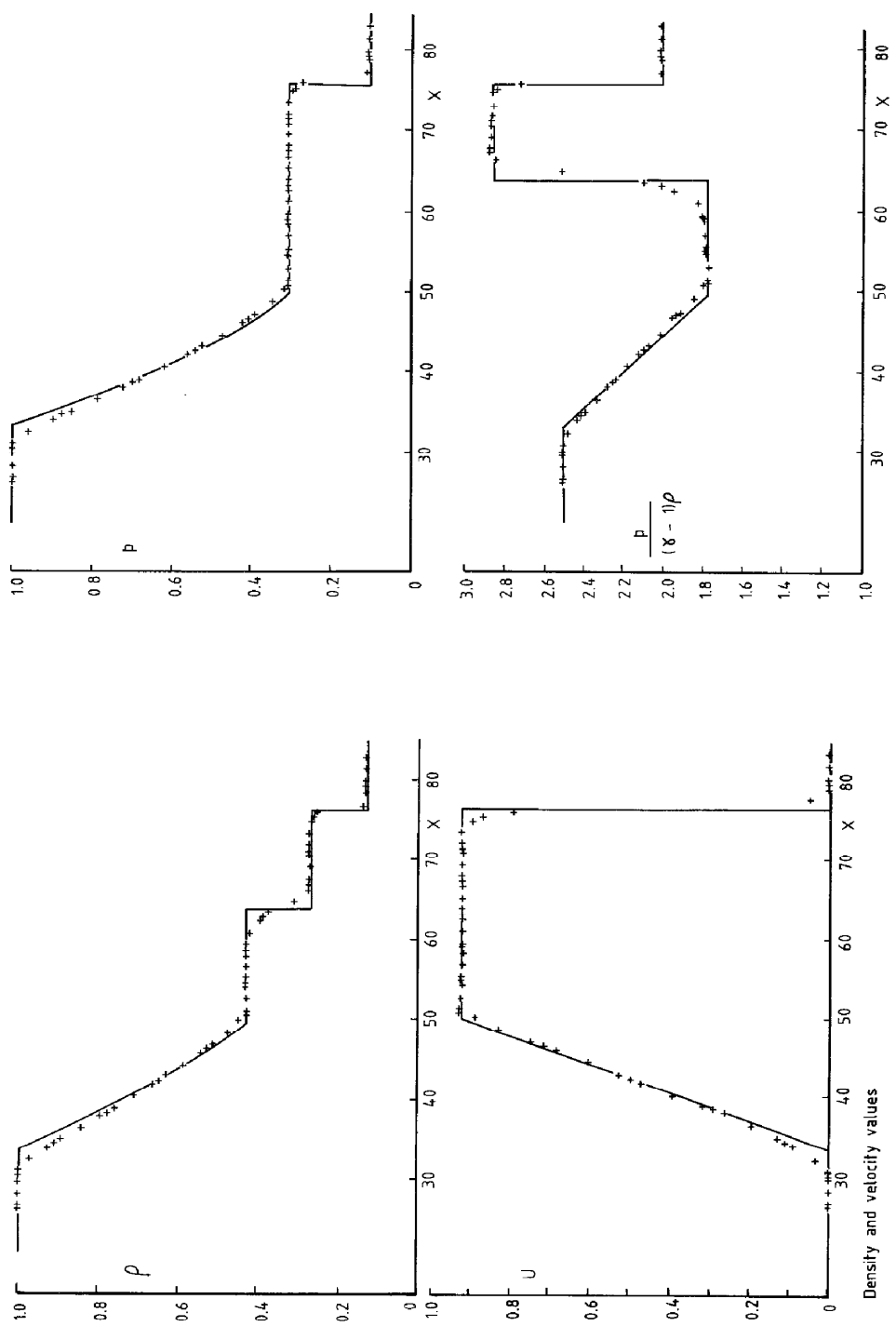

FIG. 14. Sod's problem solved with a second order accurate algorithm (α, β, γ) from Eqs. (39)–(41).

algorithms developed for irregular grids. The grid spacing used is $\frac{1}{2}$, $\frac{1}{2}$, $1\frac{1}{2}$, $1\frac{1}{2}$ (repeating), because this type of spacing tends to exhibit the errors in the solution. The solution on this grid using 70 time-steps is shown for the zero order accurate algorithm by the crosses in Fig. 12. We see how the excessive smoothing found in Fig. 6 and 9 for this algorithm is also apparent in the Euler solution. The solution for the first order accurate algorithm shown in Fig. 13 is a considerable improvement on that shown in Fig. 12. However, there is still inadequate representation of the discontinuities, particularly the contact discontinuity, and a raggedness about the representaton of the expansion. Both these features are improved by using the second order accurate algorithm as can be seen by the solution in Fig. 14. The results from this algorithm (Appendix A) give results at the standard expected from a good algorithm on an equally spaced grid, in particular, they compare well with those of reference [3], demonstrating that good quality solutions are achievable even on grids where large changes of grid spacing occur.

6. Conclusions

Algorithms have been developed for the solution of hyperbolic conservation laws in one dimension on grids with random spacing. The algorithms are constructed to be conservative stable and accurate up to second order in the average grid spacing. It is shown how standard second order accurate algorithms for an equally spaced grid may be interpreted as first order algorithms on an irregular grid. To obtain second order accuracy information from one more data point must be included. A technique is presented for modifying these algorithms to meet the monotonicity condition. The results for test problems using the linear wave equation, Burger's equation, and the Euler equation show that the results for the second order accurate algorithm on highly irregular grids are nearly as good as the results on an equally spaced grid.

APPENDIX A: An Algorithm for Irregular Grids

It is assumed that at some time t, data U_i is known at the points X_i. To evaluate the solution to Eq. (1) at time $t+\Delta t$, the data is updated by adding contributions originating from each interval ΔX_i, where ΔX_i is the interval $X_i - X_{i-1}$ with the points numbered such that $CX_i > CX_{i-1}$. The magnitude of the contribution from the interval ΔX_i is $v_i \Delta U_i$, where v_i is the Courant number $C\,\Delta t/\Delta X_i$ and ΔU_i is $U_i - U_{i-1}$.

The contributions from the intervals are distributed to update the adjacent data. That is, the contribution $v_i \Delta U_i$ from the interval ΔX_i is distributed as an upwind contribution $\mu_i \alpha_i v_i \Delta U_i$ to the point X_{i-1}, a downwind contribution to the point $\lambda_i \gamma_i v_i \Delta U_i$ to the point X_{i+1} and the remainder as a contribution $(1-\mu_i\alpha_i-\lambda_i\gamma_i)\, v_i \Delta U_i$ to the point X_i.

The distribution coefficients α_i and γ_i are given by

$$\alpha_i = \frac{1-v_i}{2}\left(1 - \frac{1+v_i}{1+\Delta X_{i-1}/\Delta X_i}\right)$$

$$\gamma_i = \frac{v_i^2 - (\Delta X_{i-1}/\Delta X_i)^2}{1+\Delta X_{i+1}/\Delta X_i}$$

and the compatibility factors to maintain stability are positive and are given for monotonic data ($\Delta U_i \Delta U_{i-1} \geqslant 0$) by

$$\lambda_i = \text{minimum}\left\{1, \left(\frac{1}{2v_i} + \frac{1}{2v_{i-1}} - 1\right)\frac{\Delta U_{i-1}}{\alpha_i \Delta U_i}\right\}$$

$$\mu_i = \text{minimum}\left\{\lambda_i, \frac{1-\alpha_{i+1}}{-\gamma_i} \cdot \frac{\Delta U_{i+1}}{\Delta U_i}\right\}$$

and for $\Delta U_i \Delta U_{i-1} < 0$ by

$$\lambda_i = \text{minimum}\left\{1, \frac{-\Delta U_{i-1}}{\alpha_i \Delta U_i}\right\}$$

$$\mu_i = 0.$$

APPENDIX B: NOMENCLATURE

C	constant of linear wave equation (Eq. (1))
C_1, C_2	parameters of 1st and 2nd order accurate algorithms (Eqs. (24) and (31))
e	specific energy $(p/(\gamma-1)\rho + \frac{1}{2}(u^2+v^2+w^2))$
$\mathbf{F}, \mathbf{G}, \mathbf{H}$	vectors of the Euler equations (Eqs. (3))
K_j	constants of polynomial data (Eq. (16))
K_1, K_2	parameters of grid adaptive algorithms (Eq. (39))
p	pressure
t	time
u, v, w	velocity components in X, Y, Z directions
U	variable of the linear wave equation (Eq. (1)) and Burgers equation (61)
$\mathbf{U}$	vector of conserved variables for Euler equation (Eq. (3))
X, Y, Z	rectangular cartesian coordinates

Greek Symbols

α, β, γ	distribution weightings for Φ (Eqs. (11)–(14))
δU	change of U in time Δt (Eq. (9))
Δt	time increment
ΔX_i	cell length $(X_{i+1} - X_i)$
ΔX_R	reference cell length
ΔU	change of U in distance ΔX (Eq. (17))
λ, μ	compatibility constants (Eqs. (46) and (53))
ν	CFL number
ρ	density
Φ	increase of U in cell in time Δt (Eq. (10))

REFERENCES

1. S. OSHER AND R. SANDERS, *Math. Comput.* **41**, **164**, 321 (1983).
2. J. D. HOFFMAN, *J. Comput Phys.* **46**, 469 (1982).
3. P. L. ROE, *J. Comput. Phys.* **43**, 357 (1981).
4. B. VAN LEER, *J. Comput. Phys.* **23**, 276 (1977).
5. P. L. ROE AND J. PIKE, in *Computing Methods in Applied Sciences and Engineering*, edited by R. Glowinski, J. L. Lions (North-Holland, Amsterdam, 1984), p. 499.
6. A. HARTEN, *J. Comput. Phys.* **49**, 357 (1983).
7. S. CHAKRAVATHY AND S. OSHER, in *Proceedings, 15th Summer Seminar on Applied Maths, LaJolla, CA, Pt. I* (Amer. Math. Soc., Providence, RI, 1985), p. 57.
8. P. K. SWEBY, *Comput. Methods in Appl. Sci. and Eng. VI*, (INRIA, 1984).
9. P. L. ROE, *Proc. 7th Int. Conf. Num. Methods in Fluid Dyn.* (Springer. New York, 1981).
10. P. K. SWEBY, Ph.D. thesis, Univ. of Reading, December 1982.
11. P. K. SWEBY, *Reading Univ. Num. Anal. Rept. 6* (1981).
12. P. L. ROE, RAE TR 81047, 1981 (unpublished).
13. G. SOD, *J. Comput. Phys.* **27**, 1 (1978).
14. L. G. MARGOLIN, H. M. RUPPEL, AND R. B. DEMUTH, *4th Int. Conf. in Turb Flow* (Univ. of Wales, 1985).
15. P. COLELLA AND P. WOODWARD, *J. Comput. Phys.* **54**, 174 (1984).
16. E. TURKEL, ICASE Report No. 85-43, 1985 (unpublished).