JOURNAL OF COMPUTATIONAL PHYSICS 110, 112–133 (1994)

An Implicit Lagrangian Method for Solving One- and Two-Dimensional Gasdynamic Equations

F. LIU,* A. C. MCINTOSH, AND J. BRINDLEY

University of Leeds, Leeds LS2 9JT, United Kingdom

Received March 26, 1992; revised February 1993

The gasdynamic equations are solved by using an implicit Lagrangian algorithm in one and two dimensions. The evolution equation of energy is replaced by the algebraic isentropic condition for each Lagrangian computational cell. The algorithm is essentially developed for isentropic flows but is also applicable to problems involving weak shocks where the entropy increase across the shock is fairly small. The algorithm can be used to predict shock tube problems provided that the entropy change of the shocked fluid is taken into account by incorporating the Rankine–Hugoniot condition. The present method does not require an added artificial viscosity since it contains a built-in mechanism to damp high-frequency disturbances behind shocks. The solution performance of the algorithm is assessed against the exact solution for two shock tube problems. Contact discontinuities are computed with infinite resolution (the number of cells over which the variation occurs is zero). Finally, the algorithm is applied to several gasdynamic problems. © 1994 Academic Press, Inc.

1. INTRODUCTION

Finite difference methods used to approximate Eulerian partial differential equations of gasdynamics introduce numerical diffusion, dispersion, or both through truncation errors. These errors become pronounced when the methods are used to solve the unsteady gasdynamics equations involving shocks and/or contact discontinuities. First-order schemes, such as the first-order upwind scheme of Steger and Warming [1] and the scheme of Godunov [2], have truncation errors proportional to a second derivative which plays the role of artificial viscosity. Therefore, whilst these schemes are able to dampen the high-frequency disturbances generated at shocks, they also smooth out discontinuities (shocks and contacts) and strong gradients. Second-order schemes, such as schemes of the Lax–Wendroff family, on the other hand, not only smooth out contact discontinuities, but also give rise to spurious oscillations behind shocks. In order to reduce these oscillations, von Neumann and Richtmyer [3] proposed the addition of an explicit artificial viscosity term to the gasdynamics equations. As a result, a shock transition is spread over typically three to four grid cells [4]. Artificial viscosity does not contribute to the smearing of a contact discontinuity.

The random choice method (RCM) [5] is capable of computing shocks and contact discontinuities with infinite resolution. However, the method is inaccurate in smooth regions of the flow and is unable to produce accurate positions of discontinuities [4]. Another method to keep shocks and contact discontinuities sharp has been developed by Harten and Hyman [6]. In this method, the computational grids are adjusted (within the underlying fixed mesh) from time step to time step in order to minimise numerical diffusion. Although the method produces very accurate results for the shock tube problems of Sod [4] and Lax [7], it is very difficult to extend to two dimensions.

Recently several sophisticated finite difference techniques have been developed which are capable of capturing discontinuities more accurately. These include the essentially non-oscillatory (ENO) scheme [8], the weighted average flux (WAF) method [9], and TVD-type (total variation diminishing) high resolution schemes [10]. TVD-type schemes have gained popularity for their applications in compressible flow. These schemes all contain some amount of internal dissipation. In TVD schemes the amount of this inherent numerical dissipation depends on the flux limiter used [10]. When these schemes are applied to a shock tube problem, they produce very high resolution for the shock, especially in TVD-type schemes; however, the contact discontinuity is still spread over typically three to four grid cells. For Eulerian finite difference methods, contact discontinuities are more difficult to compute with high resolution than shocks since they do not have a natural compression mechanism to help their sharp numerical resolution.

In some problems we can tolerate this non-physical smearing of sharp gradients at a contact discontinuity. It is not acceptable, however, in others, in particular the smearing of a density interface in the study of the deformation of a premixed flame front under the action of a pressure disturbance [11], with which we are concerned here.

Our study is motivated by the deformation and acceleration of a curved flame under the action of a pressure

* Present address: Department of Chemical Engineering, Queens University, Kingston, Ontario, Canada K7L 3N6.

gradient. At the very early stage of the interaction between a pressure wave and a premixed flame the chemical reaction and molecular diffusion processes can be neglected and the governing equations reduce to the equations of gasdynamics. The flame front is essentially convected with the flow driven by the pressure wave. Such a simplified gasdynamic model holds true only at the early stage of the interaction since at a later stage reaction and diffusion-dominated physical processes cannot be ignored. Under these circumstances we might expect a Rayleigh–Taylor-type instability to be important. This physical situation is very difficult to simulate numerically since the flame behaves as a contact discontinuity and numerical diffusion is not acceptable.

The Rayleigh–Taylor instability [12] occurs when an inertial or gravitational force accelerates an interface between two incompressible fluids of different density. A similar phenomenon is the Richtmyer–Meshkov instability, which is produced by a shock wave hitting an interface (a contact discontinuity or material boundary) separating fluids of different density [13]. The growth rate of these instabilities depends on the density ratio of the fluids at the interface. When modelling them, therefore, it is crucial to keep the fluid interface sharp, otherwise the physics could be largely suppressed by the smearing of the density interface caused by numerical diffusion. This is particularly true at the early stage of the development of the interface instability which is the main concern of the present study. A TVD scheme, however, would be the best choice to study an interface instability if one is interested in a long time effect. To study the early stage of the interface instability the required numerical algorithm must be able to compute a contact discontinuity with infinite resolution and should be easily extendable to two dimensions. None of the above methods appears to satisfy these requirements and we are led to the investigation of a Lagrangian method.

Lagrangian methods appear to offer substantial advantages over Eulerian methods as far as modelling a contact discontinuity is concerned. Since they do not suffer from numerical diffusion around contact discontinuities and therefore can keep them perfectly sharp, they are suitable methods for computing interface instabilities. In addition, when the physical problem of interest involves free surfaces, interfaces, or discontinuities, it is far more natural to use a Lagrangian method to perform the numerical simulation. Lagrangian calculations are free from numerical diffusion around a contact discontinuity for two reasons. First, there is no explicit dependence of density on the spatial derivatives in the governing equations; second, pressure and velocity are conserved across the contact discontinuity. However, Lagrangian methods still suffer numerical diffusion in regions away from the contact discontinuity because of the presence of convective derivatives of velocity and pressure in the governing equations. Although Lagrangian methods offer some advantages over Eulerian methods, they suffer from the severe drawback that they break down when the flow of interest undergoes large fluid distortions. A Lagrangian method, however, is still suitable to study the pressure wave/premixed flame interaction at the early stage for two reasons. First, it can keep a contact discontinuity perfectly sharp; second, we do not need to worry about the above drawback since the simplified gasdynamic model only holds true at the early stage, i.e., before the method collapses due to severe fluid distortions.

For incompressible fluid problems, Lagrangian methods have been used successfully to study the Rayleigh–Taylor instability [14] and the Kelvin–Helmholtz instability [15].

Several Lagrangian methods for solving unsteady gasdynamics equations have also been developed in the literature. For example, the ALE (arbitrary Lagrangian–Eulerian) methods [16, 17] have been widely used in computational fluid dynamics. In these methods, the vertices of a finite difference mesh can move with the fluid (Lagrangian), be fixed (Eulerian), or be moved in any other prescribed way. More recently Bilbao [18] presented a fully Lagrangian method for unsteady compressible fluids in three dimensions. In these Lagrangian methods, the evolution equations of mass, momentum, and energy in differential form are solved. In addition, they all require the addition of an explicit artificial viscosity term to the equations of momentum and energy to reduce numerical oscillations when the problems of interest involve shocks. Boris [19] had earlier developed another Lagrangian method for one-dimensional compressible flows in which the evolution equation for energy was replaced by using the isentropic condition for each Lagrangian computational cell. However, the numerical results based on this method have not involved problems with shocks and/or contact discontinuities. The present Lagrangian algorithm is based on the method of Boris [19].

In the present algorithm no explicit artificial viscosity terms are required; instead the algorithm can be made internally dissipative to dampen numerical oscillations behind shocks. The evolution equation of energy is eliminated by using the isentropic equation of state, so that the entropy of each computational cell is strictly conserved throughout the computation. In one dimension, the sudden change in entropy of a Lagrangian computational cell across the shock is taken into account by using the Rankine–Hugoniot relation. The algorithm is verified by comparison with the exact solution for two shock tube problems in each of which a shock and a contact discontinuity are formed in the flow. They provide good testbeds, therefore, for the evaluation of the method. Finally to further demonstrate the merit of this algorithm, it is used to study the response of a density discontinuity (which simulates a premixed flame front on an acoustic time scale) to a harmonic pressure input in both one and two dimensions.

2. BASIC EQUATIONS

The equations governing the conservation of mass, momentum, and energy (written in terms of entropy) for an unsteady two-dimensional gasdynamic flow are written in Lagrangian form as

$$\frac{d\rho}{dt} = -\rho\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right), \tag{1}$$

$$\frac{du}{dt} = -\frac{1}{\rho}\frac{\partial p}{\partial x}, \tag{2}$$

$$\frac{dv}{dt} = -\frac{1}{\rho}\frac{\partial p}{\partial y}, \tag{3}$$

$$\frac{dS}{dt} = 0. \tag{4}$$

Equation (4) is the isentropic condition for each Lagrangian fluid element; it replaces the evolution equation of energy when temperature is eliminated from the energy equation using the state equation of an ideal gas. The advantages of using Eq. (4) are twofold. First, we are able to deal with an algebraic equation rather than a differential one. Second, when the isentropic condition replaces the evolution equation of energy, there is no need to evaluate the velocity of a computational cell, which is required by its kinetic energy calculation, from the interpolation over the velocities of its four vertices (this will become clear in Section 4). An appropriate entropy expression for a Lagrangian fluid element is defined as

$$S = p/\rho^{\gamma}, \tag{5}$$

where γ is the specific heat ratio. When the flow involves a shock, Eq. (4) is not valid for a Lagrangian fluid element crossing the shock, and its entropy is subject to an increase according to the Rankine–Hugoniot relation. In one dimension, it is straightforward to make such an entropy modification for a Lagrangian fluid element crossing the shock by using the Rankine–Hugoniot relation as described in the next section. In two dimensions, however, it is not so easy to apply the Rankine–Hugoniot condition and we make no attempt in the present work to conduct the entropy modification for a two-dimensional shocked flow.

Finally, the evolution equations of location of a Lagrangian fluid element are given by

$$\frac{dx}{dt} = u, \tag{6}$$

$$\frac{dy}{dt} = v. \tag{7}$$

3. NUMERICAL ALGORITHM IN ONE-DIMENSION

3.1. *Formulation of the Algorithm*

A typical computational domain for the one-dimensional algorithm, shown in Fig. 1, consists of N cells of volume A_i, $i = 1, 2, ..., N$, bounded on $N+1$ cell interfaces located at x_i, $i = 1, 2, ..., N+1$. The cell volumes, A_i, are the difference between each two sequential cell interfaces such that

$$A_i = x_{i+1} - x_i. \tag{8}$$

At each of these cell interfaces, say the ith, there is stored a location x_i and velocity u_i. The pressure p_i, density ρ_i, and cell volume A_i are stored at the centre of each computational cell bounded by the ith and $(i+1)$th cell interfaces. For the sake of convenience, we introduce cell mass M_i and interface mass m_i which are respectively defined as

$$M_i = \rho_i A_i, \tag{9}$$

$$m_i = \tfrac{1}{2}(M_i + M_{i-1}). \tag{10}$$

The derivative of M_i with respect to time can be written as, based on the mass conservation Eq. (1),

$$\frac{dM_i}{dt} = A_i\left(\frac{d\rho_i}{dt} + \rho_i\frac{\partial u_i}{\partial x}\right) = 0. \tag{11}$$

Equation (11) implies that each cell mass M_i is constant throughout its evolution.

The evolution equation of cell interface, Eq. (6), is discretised to

$$x_i^{n+1} = x_i^n + \Delta t[\varepsilon_x u_i^n + (1-\varepsilon_x)\, u_i^{n+1}]. \tag{12}$$

The superscript n indicates variables at time t, and the superscript $n+1$ indicates variables at time $t+\Delta t$. The quantity ε_x is the explicitness parameter for the interface locations. Conventionally (see [19], for example), ε_x is restricted to $0 \leqslant \varepsilon_x \leqslant 1$. However, in the present work, ε_x is allowed to be smaller than zero. A qualitative discussion of the effect of ε_x and ε_u (see below) on the solution performance of the present algorithm will be given in Section 3.3. When $\varepsilon_x < 1$, the method is implicit; when $\varepsilon_x = \frac{1}{2}$, the method is nominally most accurate; when $\varepsilon_x = 1$, the method is explicit and most unstable.

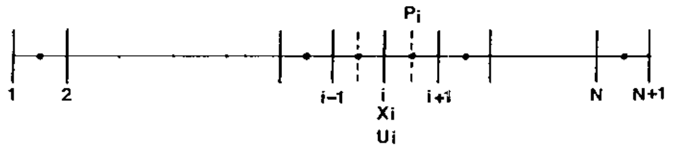


FIG. 1. A typical computational domain for the Lagrangian algorithm.

Integration of Eq. (2) over the domain bounded by the two dashed lines (see Fig. 1) for the ith cell interface yields

$$\frac{du_i}{dt} = -\frac{p_i - p_{i-1}}{m_i}. \tag{13}$$

Equation (13) is in fact Newton's second law written for the ith interface mass m_i. It is worth noting that it is the same equation as that obtained by Boris [19] based on a different physical argument. Equation (13) is then further discretised to

$$u_i^{n+1} = u_i^n - \frac{\Delta t \varepsilon_u}{m_i}(p_i^n - p_{i-1}^n) - \frac{\Delta t(1-\varepsilon_u)}{m_i}(p_i^{n+1} - p_{i-1}^{n+1}), \tag{14}$$

where ε_u is the explicitness parameter for the interface velocity, having the same properties as ε_x. The parameter ε_u is also allowed to be negative.

The way of discretising Eqs. (2) and (6) (see Eqs. (14) and (12)) is very similar to, but less general than that used in the schemes of Lerat and Peyret [20] and Beam and Warming [21].

To simplify the following discussion, we introduce two auxiliary parameters A_i and B_i such that

$$A_i = u_i^n - \frac{\Delta t \varepsilon_u}{m_i}(p_i^n - p_{i-1}^n), \tag{15}$$

$$B_i = \frac{\Delta t(1-\varepsilon_u)}{m_i}. \tag{16}$$

Equation (14) is then written as

$$u_i^{n+1} = A_i - B_i(p_i^{n+1} - p_{i-1}^{n+1}). \tag{17}$$

The ith cell volume at time $t + \Delta t$ is achieved by using Eqs. (8), (12), and (17), viz.

$$\Lambda_i^{n+1} = \Lambda_i^n + \Delta\Lambda_i, \tag{18}$$

where

$$\Delta\Lambda_i = \Delta t\{\varepsilon_x(u_{i+1}^n - u_i^n)\} + \Delta t\{(1-\varepsilon_x)[(A_{i+1} - A_i) - B_{i+1}(p_{i+1}^{n+1} - p_i^{n+1}) + B_i(p_i^{n+1} - p_{i-1}^{n+1})]\}. \tag{19}$$

The present numerical algorithm is built on the fact that at time $t + \Delta t$ the density of the ith cell computed from the mass conservation equation, Eq. (9), must be equal to that calculated from the isentropic condition, Eq. (5), given by

$$\frac{M_i}{\Lambda_i^{n+1}} = \left(\frac{p_i^{n+1}}{S_i}\right)^{1/\gamma}. \tag{20}$$

Equation (20) can also be written as

$$p_i^{n+1} = S_i\left(\frac{M_i}{\Lambda_i^{n+1}}\right)^{\gamma}. \tag{21}$$

This system (Eq. (21)) comprises $N-2$ nonlinear algebraic equations for pressures p_i^{n+1} ($i = 2, 3, ..., N-1$) and is used in the present numerical algorithm to obtain the pressure field at time $t + \Delta t$.

Equations (21) and (17) are subject to appropriate boundary conditions. Those used in this work to test shock tube problems are the simple transmissive boundary conditions at both ends of the tube. The $N-2$ nonlinear algebraic equations for pressure are then solved iteratively by using the Newton–Raphson method, stopping the iteration once the prescribed convergence condition is satisfied. This is taken to be

$$\max\left|\frac{p_i^{M+1} - p_i^M}{p_i^M}\right| \leqslant 10^{-7}, \qquad i = 2, 3, ..., N-1, \tag{22}$$

where the superscript M represents variables at the Mth iteration.

3.2. *Entropy Modification at a Shock*

The equations of inviscid gasdynamics admit discontinuous solutions which cannot satisfy the differential equations but are valid solutions to these equations in integral form. Although several such discontinuous solutions can exist, not all of them are physically acceptable. In the present algorithm, it is straightforward to obtain different discontinuous solutions by making different assumptions for the entropy change of each Lagrangian cell crossing a shock front. Physical arguments, however, require that the correct solution is obtained when the entropy change across the shock is evaluated by the Rankine–Hugoniot condition.

When a shock is present in the flow it is therefore necessary to apply the Rankine–Hugoniot relation for each Lagrangian computational cell crossing the shock in order to modify its entropy increase. To illustrate how to do this, we consider here only the simplest case. More complicated situations may require more sophisticated methods to detect the shock in order to incorporate the Rankine–Hugoniot condition into the algorithm, according to the characteristics of the flows considered.

Consider therefore a situation where a diaphragm initially at the Ith cell interface separates two regions, and the shock formed propagates towards the right-hand side (see Fig. 1). The iteration of the system of Eqs. (21) starts by using the initial values of entropy for all the computational cells to obtain the updated pressure distribution, p_i^{n+1} ($i = 1, 2, ..., N$). Rather than using the isentropic condition (see Eq. (20)), the Rankine–Hugoniot relation is applied to calculate the density of all the computational cells on the right-hand side of the diaphragm such that

$$\rho_i^{n+1}=\rho_i^0\frac{(\gamma+1)(p_i^{n+1}/p_i^0)+(\gamma-1)}{(\gamma+1)+(\gamma-1)(p_i^{n+1}/p_i^0)}, \qquad \text{for} \quad i\geqslant I, \tag{23}$$

where the superscript zero denotes initial values. Next the current entropy of a computational cell on the right-hand side of the diaphragm is modified according to

$$S_i=\frac{p_i^{n+1}}{(\rho_i^{n+1})^\gamma}. \tag{24}$$

This new entropy is then used in the calculation for the next time step. Note that the use of Eq. (23) does not affect the density of cells ahead of the shock. It is also clear that it does not require too much extra cpu time to carry out this entropy modification since the implementation of the Rankine–Hugoniot relation in the algorithm is non-iterative.

3.3. *The Properties of* ε_x *and* ε_u

We now explore the properties of ε_x and ε_u, which are analysed through the conservation equations of mass and momentum. We first consider the mass conservation equation. For the sake of convenience, we denote the difference of any physical quantity ξ (such as u or du/dt) between point $(i+1)$ and point i as

$$\Delta\xi_i=\xi_{i+1}-\xi_i. \tag{25}$$

Based on Eqs. (8) and (9), the mass conservation equation (1) can also be written as

$$\frac{d\rho_i}{dt}=-\frac{M_i}{A_i^2}\frac{dA_i}{dt}. \tag{26}$$

The ith cell volume, A_i, at time $(t+\Delta t)$ can be evaluated, using Eq. (12), as

$$A_i^{n+1}=A_i^n+\Delta t[\varepsilon_x\,\Delta u_i^n+(1-\varepsilon_x)\,\Delta u_i^{n+1}]. \tag{27}$$

We can expand A_i^{n+1} and Δu_i^{n+1} in terms of a Taylor series as

$$A_i^{n+1}=A_i^n+\Delta t\frac{dA_i}{dt}+\frac{1}{2}\Delta t^2\frac{d^2A_i}{dt^2}+O(\Delta t^3), \tag{28}$$

$$\Delta u_i^{n+1}=\Delta u_i^n+\Delta t\frac{d\,\Delta u_i}{dt}+O(\Delta t^2). \tag{29}$$

Substitution of Eqs. (28) and (29) into Eq. (27) then leads to

$$\frac{dA_i}{dt}=\Delta u_i+\left(\frac{1}{2}-\varepsilon_x\right)\Delta t\frac{d\,\Delta u_i}{dt}+O(\Delta t^2). \tag{30}$$

Note that $d^2A_i/dt^2=d\,\Delta u_i/dt$ has been used in obtaining Eq. (30). In order to analyse the effect of ε_x on the numerical scheme, it is desirable to write $d\,\Delta u_i/dt$ in terms of spatial derivatives such that

$$\begin{aligned}\frac{d\,\Delta u_i}{dt}&\approx\frac{d}{dt}\left(A_i\frac{\partial u_i}{\partial x}\right)\\&=\frac{dA_i}{dt}\frac{\partial u_i}{\partial x}+A_i\frac{\partial}{\partial x}\frac{du_i}{dt}\\&\approx A_i\left(\frac{\partial u_i}{\partial x}\right)^2-A_i\frac{\partial}{\partial x}\left(\frac{1}{\rho}\frac{\partial p}{\partial x}\right)_i.\end{aligned} \tag{31}$$

By using Eqs. (30) and (31), Eq. (26) now can be re-written as

$$\frac{d\rho_i}{dt}=-\left(\rho\frac{\partial u}{\partial x}\right)_i+\left(\frac{1}{2}-\varepsilon_x\right)\Delta t\left[\rho\frac{\partial}{\partial x}\left(\frac{1}{\rho}\frac{\partial p}{\partial x}\right)-\rho\left(\frac{\partial u}{\partial x}\right)^2\right]_i+O(\Delta t^2). \tag{32}$$

Next we consider the momentum conservation equation. Equation (14) can be written in the form

$$\frac{u_i^{n+1}-u_i^n}{\Delta t}=\varepsilon_u\left.\frac{du}{dt}\right|_i^n+(1-\varepsilon_u)\left.\frac{du}{dt}\right|_i^{n+1}, \tag{33}$$

where u_i^{n+1} and $(du/dt)_i^{n+1}$ can be expanded as

$$u_i^{n+1}=u_i^n+\Delta t\frac{du_i}{dt}+\frac{1}{2}\Delta t^2\frac{d^2u_i}{dt^2}+O(\Delta t^3), \tag{34}$$

$$\left.\frac{du}{dt}\right|_i^{n+1}=\frac{du_i}{dt}+\Delta t\frac{d^2u_i}{dt^2}+O(\Delta t^2). \tag{35}$$

Substitution of Eqs. (34) and (35) into Eq. (33) results in

$$\frac{du_i}{dt}=\left(-\frac{1}{\rho}\frac{\partial p}{\partial x}\right)_i+\left(\frac{1}{2}-\varepsilon_u\right)\Delta t\left(\frac{d^2u_i}{dt^2}\right)+O(\Delta t^2). \tag{36}$$

Finally by writing d^2u/dt^2 as

$$\begin{aligned}\frac{d^2u}{dt^2}&=\frac{d}{dt}\left(-\frac{1}{\rho}\frac{\partial p}{\partial x}\right)\\&=\frac{d}{dt}\left(-\frac{1}{\rho}\right)\frac{\partial p}{\partial x}+\left(-\frac{1}{\rho}\right)\frac{\partial}{\partial x}\frac{dp}{dt}\\&=\frac{1}{\rho^2}\frac{d\rho}{dt}\frac{\partial p}{\partial x}-\frac{1}{\rho}\frac{\partial}{\partial x}\frac{d}{dt}(S\rho^\gamma)\\&=-\frac{1}{\rho}\frac{\partial u}{\partial x}\frac{\partial p}{\partial x}+\frac{1}{\rho}\frac{\partial}{\partial x}\left(S\gamma\rho^\gamma\frac{\partial u}{\partial x}\right),\end{aligned} \tag{37}$$

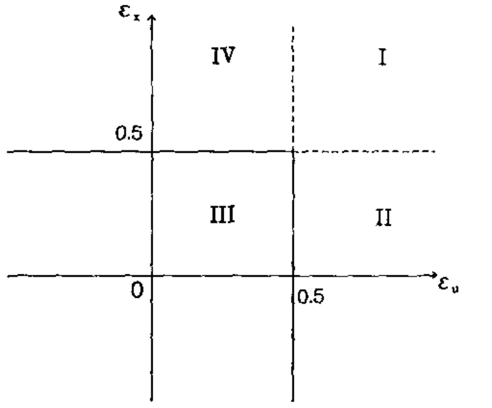


FIG. 2. Four regions in the $(\varepsilon_u, \varepsilon_x)$ plane that characterise the numerical performance of the algorithm. Region I: negative diffusion. Region III: positive diffusion. Regions II and IV: negative and positive diffusion co-exist.

Eq. (36) takes the form

$$\frac{du_i}{dt} = \left(-\frac{1}{\rho}\frac{\partial p}{\partial x}\right)_i + \left(\frac{1}{2} - \varepsilon_u\right)\Delta t\left[-\frac{1}{\rho}\frac{\partial u}{\partial x}\frac{\partial p}{\partial x} + \frac{1}{\rho}\frac{\partial}{\partial x}\left(S\gamma\rho^\gamma\frac{\partial u}{\partial x}\right)\right]_i + O(\Delta t^2). \tag{38}$$

The properties of ε_x and ε_u can be clearly seen from Eqs. (32) and (38). Straight lines $\varepsilon_x = \frac{1}{2}$ and $\varepsilon_u = \frac{1}{2}$ divide the $(\varepsilon_x, \varepsilon_u)$ plane into four regions as shown in Fig. 2. When $\varepsilon_x = \varepsilon_u = \frac{1}{2}$, the algorithm is second-order accurate in time and there is no built-in dissipation to eliminate high-frequency disturbances (or oscillation) generated by discontinuities or sharp gradients. In region I, see Fig. 2, where both ε_x and ε_u are greater than $\frac{1}{2}$, there exists a negative numerical viscosity term in both Eq. (32) and Eq. (38); this causes numerical instability or oscillation around discontinuities or strong gradients. In region III of Fig. 2, where ε_x and ε_u are both smaller than $\frac{1}{2}$, Eqs. (32) and (38) both have truncation errors proportional to a second derivative in space of positive coefficient, which acts as an added numerical viscosity. Therefore, the scheme in this case has a built-in mechanism to dampen the high-frequency components generated around discontinuities or strong gradients. It should be noted that the larger the time step Δt, or the smaller the values of ε_x and ε_u, the stronger the damping effect of the inherent numerical diffusion. This internal dissipation has been a common feature shared by many popular numerical schemes for capturing shocks, such as Lax–Wendroff [22], Lerat and Peyret [20], and Beam and Warming [21]. It can also be shown (see Pulliam [23], for example) that recently developed high resolution schemes (monotone and total variation diminishing) are all equivalent to a central differencing scheme plus some form of implicit dissipation. In regions II and IV, where one of ε_x and ε_u is greater than $\frac{1}{2}$ but the other is smaller than $\frac{1}{2}$, the scheme contains a negative numerical diffusion term in Eq. (32) and a positive numerical diffusion term in Eq. (38), or vice versa. The net overall effect, similar to either region I or region III, depends on which mechanism is dominant. In order to simplify the following discussion, we constrain the values of ε_x and ε_u to vary together and to remain in region III, i.e., $\varepsilon_x = \varepsilon_u \leqslant 0.5$.

3.4. *Accuracy and Stability*

The accuracy of the present algorithm is only first order in time, although it can be made second-order accurate in time when using $\varepsilon_x = \varepsilon_u = \frac{1}{2}$.

A strict stability analysis is very difficult because of its non-linear nature. Following the analysis presented by Bilbao [18], it can be shown that the explicit version of the present algorithm ($\varepsilon_x = \varepsilon_u = 1$) is always unstable and the fully implicit version of the algorithm ($\varepsilon_x = \varepsilon_u = 0$) is unconditionally stable. In order to perform a stable calculation as discussed by Oran and Boris [24, Chap. 10], we still need a Courant-like condition to be satisfied for the flow velocities, u_i.

All Lagrangian methods assume that the interface position x_i increases monotonically with increasing i. The non-physical crossing of cell interfaces is a large potential source of instability. This can occur for large time steps even though the fully implicit algorithm is nominally unconditionally stable. The condition that is necessary and sufficient to prevent interface crossings can be written as

$$\Delta_i^{n+1} = x_{i+1}^{n+1} - x_i^{n+1} > 0, \tag{39}$$

where Δ_i^{n+1} is calculated from Eq. (27). Employing equation (14), Δu_i^{n+1} can be written as

$$\Delta u_i^{n+1} = \Delta u_i^n + \varepsilon_u \Delta t\, \Delta \left.\frac{du}{dt}\right|_i^n + (1-\varepsilon_u)\Delta t\, \Delta \left.\frac{du}{dt}\right|_i^{n+1}. \tag{40}$$

The last term on the right-hand side of Eq. (40) may be approximated as

$$\Delta \left.\frac{du}{dt}\right|_i^{n+1} \approx \Delta \left.\frac{du}{dt}\right|_i^n + \Delta t\, \Delta \left.\frac{d^2u}{dt^2}\right|_i^n. \tag{41}$$

By virtue of Eqs. (27), (40), and (41), Eq. (39) can be written as

$$\Delta_i^n + \Delta t\left[\Delta u_i^n + (1-\varepsilon_x)\Delta t\, \Delta\left(\frac{du}{dt}\right)_i^n + (1-\varepsilon_x)(1-\varepsilon_u)\Delta t^2\, \Delta\left(\frac{d^2u}{dt^2}\right)_i^n\right] > 0. \tag{42}$$

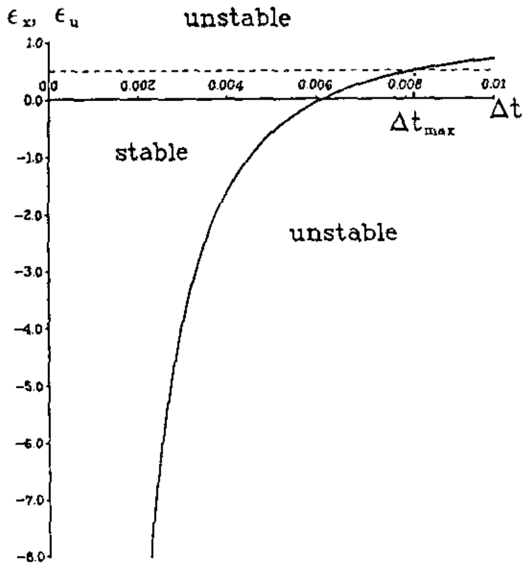


FIG. 3. Stability limit based on the monotonicity of interface positions and non-negative numerical diffusion of the algorithm.

The maximum stable time step is then limited by the condition stated in Eq. (42). This equation also suggests that ε_x affects the maximum stable time step to a larger extent than ε_u. In fact, a good estimate of the maximum stable time step may be made by neglecting the last term on the right-hand side of Eq. (42), i.e.,

$$A_i^n + \Delta t \left[\Delta u_i^n + (1 - \varepsilon_x)\, \Delta t\, \Delta \left(\frac{du}{dt}\right)_i^n \right] > 0, \qquad (43)$$

where $\Delta(du/dt)_i^n$ can be calculated from Eq. (13). Although Eq. (43) provides a means to control the time step in order to prevent cell interface crossings, it is preferable to use a fixed time step in the calculation since it can be somewhat time consuming to estimate the maximum stable time step using Eq. (43).

The condition imposed by Eq. (43) can also be considered as a restriction on the value of ε_x for a given time step Δt. Substitution of $\min[\Delta(du/dt)_i^n]$, which is negative, into Eq. (43) yields

$$\varepsilon_x > 1 + \frac{A_i^n + \Delta t\, \Delta u_i^n}{\Delta t^2 \min[\Delta(du/dt)_i^n]}. \qquad (44)$$

Equation (44) gives the stability limit of ε_x (or ε_u) for a given time step Δt. Bearing in mind that we have constrained our discussion to region III of Fig. 2, the stability limit of ε_x (or ε_u) can now be shown in Fig. 3. This figure is for a specific set of parameters of $A_i^n = 0.01$, $\Delta u_i^n = -0.4$, and $\Delta(du/dt)_i^n = -200.0$, but the general trend will always be the same. In this figure $\Delta t_{\max}$ denotes the maximum stable time step. It is seen that the larger the time step Δt, the narrower the stability region for ε_x.

4. NUMERICAL ALGORITHM IN TWO DIMENSIONS

4.1. *Discretisation of the Basic Equations*

The finite difference mesh used in the present study consists of a network of quadrilateral cells with vertices labelled by integer pairs (i, j), denoting column i and row j. The assignment of variables about a typical computational cell is shown in Fig. 4a, where pressure (p), density (ρ), cell volume (A), cell mass (M), and cell entropy (S) are all assigned to the cell centre; while coordinates (x, y), velocity components (u, v), and vertex mass (m, defined below) are assigned to cell vertices.

The mass conservation equation, Eq. (1), implies that each cell mass $M_{i,j}$ is constant throughout its evolution. $M_{i,j}$ is the product of density and cell volume and written as

$$M_{i,j} = \rho_{i,j} A_{i,j}. \qquad (45)$$

The cell volume $A_{i,j}$ can be calculated by

$$A_{i,j} = \frac{1}{2} \begin{vmatrix} x_{i,j} - x_{i+1,j+1} & y_{i,j} - y_{i+1,j+1} \\ x_{i+1,j} - x_{i,j+1} & y_{i+1,j} - y_{i,j+1} \end{vmatrix}. \qquad (46)$$

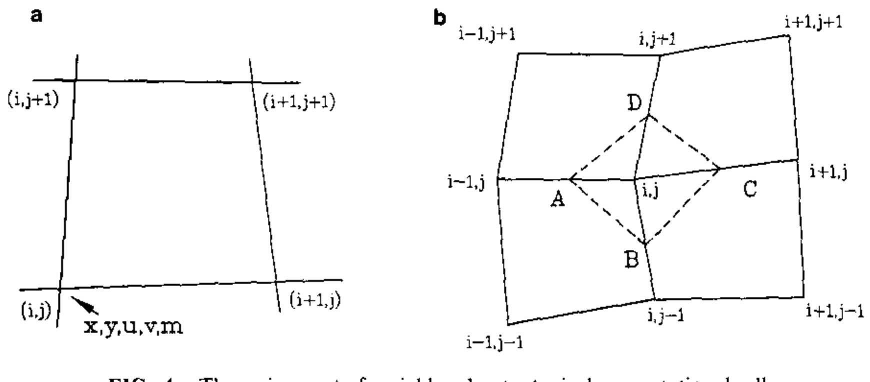


FIG. 4. The assignment of variables about a typical computational cell.

To calculate velocities it is necessary to integrate the momentum equations (2) and (3). Figure 4b shows a typical computational volume $ABCD$ for momentum integration, where A, B, C, and D are mid-points, for the vertex (i, j). In the present algorithm the mass of vertex (i, j), $m_{i,j}$, is taken to be the average among the masses of its four neighbouring cells and therefore is also a constant given by

$$m_{i,j}=\tfrac{1}{4}(M_{i-1,j-1}+M_{i-1,j}+M_{i,j-1}+M_{i,j}). \quad (47)$$

Integration of Eqs. (2) and (3) over the volume enclosed by the dashed lines for the vertex mass $m_{i,j}$ shown in Fig. 4b yields

$$\frac{du_{i,j}}{dt}=\frac{F_{xi,j}}{m_{i,j}}, \quad (48)$$

$$\frac{dv_{i,j}}{dt}=\frac{F_{yi,j}}{m_{i,j}}, \quad (49)$$

where F_x and F_y are components of pressure force acting on the vertex mass $m_{i,j}$ in the x and y directions, respectively. F_x and F_y are calculated from

$$F_{xi,j}=\tfrac{1}{2}[p_{i,j}(y_{i+1,j}-y_{i,j+1})+p_{i-1,j}(y_{i,j+1}-y_{i-1,j}) + p_{i-1,j-1}(y_{i-1,j}-y_{i,j-1})+p_{i,j-1}(y_{i,j-1}-y_{i+1,j})], \quad (50)$$

$$F_{yi,j}=\tfrac{1}{2}[p_{i,j}(x_{i,j+1}-x_{i+1,j})+p_{i-1,j}(x_{i-1,j}-x_{i,j+1}) + p_{i-1,j-1}(x_{i,j-1}-x_{i-1,j})+p_{i,j-1}(x_{i+1,j}-x_{i,j-1})]. \quad (51)$$

The isentropic condition, Eq. (4), means that the entropy of each computational cell remains constant throughout its evolution,

$$S_{i,j}=\frac{p_{i,j}}{(\rho_{i,j})^{\gamma}}. \quad (52)$$

The evolution equations of cell vertices, Eqs. (6) and (7), are discretised to

$$x_{i,j}^{n+1}=x_{i,j}^{n}+\Delta t[\varepsilon_x u_{i,j}^{n}+(1-\varepsilon_x)\,u_{i,j}^{n+1}], \quad (53)$$

$$y_{i,j}^{n+1}=y_{i,j}^{n}+\Delta t[\varepsilon_x v_{i,j}^{n}+(1-\varepsilon_x)\,v_{i,j}^{n+1}]. \quad (54)$$

The quantity ε_x is the explicitness parameter for vertex locations which has properties similar to those discussed in one dimension.

The semi-discretised momentum equations (48) and (49) are further discretised to

$$u_{i,j}^{n+1}=u_{i,j}^{n}+\Delta t\,\frac{\varepsilon_u F_{xi,j}^{n}+(1-\varepsilon_u)\,F_{xi,j}^{n+1}}{m_{i,j}}, \quad (55)$$

$$v_{i,j}^{n+1}=v_{i,j}^{n}+\Delta t\,\frac{\varepsilon_u F_{yi,j}^{n}+(1-\varepsilon_u)\,F_{yi,j}^{n+1}}{m_{i,j}}, \quad (56)$$

where ε_u is the explicitness parameter for vertex velocities. Although it is possible to assign different values to ε_x and ε_u, we constrain the values of ε_x and ε_u to be equal and in the region III, see Fig. 2, as we have done in one dimension.

In one dimension, vertex coordinates do not appear explicitly in the momentum equation, see Eq. (14). For two-dimensional flows, however, the problem is complicated by the explicit dependence of vertex velocities on their coordinates through F_x and F_y. Even so, the underlying idea of constructing the one-dimensional implicit Lagrangian algorithm presented in the last section is still applicable in two dimensions. The algorithm is again based on the fact that the density of cell (i, j) at time $t+\Delta t$, computed from the mass conservation equation, Eq. (45), must be equal to that calculated from the isentropic condition, Eq. (52), so that

$$\rho_{i,j}^{n+1}=\frac{M_{i,j}}{A_{i,j}^{n+1}}=\left(\frac{p_{i,j}^{n+1}}{S_{i,j}}\right)^{1/\gamma}. \quad (57)$$

From Eq. (57) we can write

$$p_{i,j}^{n+1}=S_{i,j}\left(\frac{M_{i,j}}{A_{i,j}^{n+1}}\right)^{\gamma}. \quad (58)$$

Equation (58) is used to obtain the pressure fields at time $t+\Delta t$.

4.2. *Solution Procedure*

Since both pressures $p_{i,j}^{n+1}$ and coordinates $(x_{i,j}^{n+1}, y_{i,j}^{n+1})$ are unknown a priori and they are strongly coupled through the momentum equations, they have to be calculated by iteration. The solution procedure employed in the present work can be summarised as:

1. Assign updated coordinates of vertex (i, j) to $(\bar{x}_{i,j}^{n+1}, \bar{y}_{i,j}^{n+1})$.
2. Assign updated pressure of cell (i, j) to $\bar{p}_{i,j}^{n+1}$.
3. Calculate velocity components $u_{i,j}^{n+1}$ and $v_{i,j}^{n+1}$ from Eqs. (55) and (56).
4. Calculate coordinates of vertex (i, j), $(x_{i,j}^{n+1}, y_{i,j}^{n+1})$, from Eqs. (53) and (54) using updated velocity components.
5. Calculate cell volumes $A_{i,j}^{n+1}$ from Eq. (46) using updated vertex coordinates.
6. Update pressures $p_{i,j}^{n+1}$ using Eq. (58).
7. Check the convergence of pressures. If inequality

$$\left|\frac{\bar{p}_{i,j}^{n+1}-p_{i,j}^{n+1}}{\bar{p}_{i,j}^{n+1}}\right|\leqslant\omega_1, \quad (59)$$

is satisfied, go to step 8. Otherwise, go to step 2.

8. Check the convergence of coordinates. If inequalities

$$\left|\frac{\bar{x}^{n+1}_{i,j} - x^{n+1}_{i,j}}{\bar{x}^{n+1}_{i,j}}\right| \leqslant \omega_2, \tag{60}$$

$$\left|\frac{\bar{y}^{n+1}_{i,j} - y^{n+1}_{i,j}}{\bar{y}^{n+1}_{i,j}}\right| \leqslant \omega_2, \tag{61}$$

are satisfied, go to step 9. Otherwise, go to step 1.

9. Advance the solution in time.

Here ω_1 and ω_2 are two prescribed small quantities, assumed to be 10^{-4} and 10^{-6}, respectively, in this study.

5. NUMERICAL EXPERIMENTS FOR ONE-DIMENSIONAL FLOWS

In this section we apply the Lagrangian algorithm of one dimension described in Section 3 to four gasdynamic flows of air ($\gamma = 1.4$). The first two are typical Riemann problems, namely the shock tube problems of Sod [4] and Lax [7]. The third one simulated the response of a planar premixed flame front, treated as a contact discontinuity on an acoustic time scale, to a sinusoidal pressure input; we consider a situation where the initial pressure and velocity are continuous but the density is not. In the last one we consider an initial pressure disturbance breaking into two travelling pressure waves; all the physical quantities are continuous in this example.

In Sod's shock tube problem, the initial discontinuity breaks into a weak shock wave followed by a contact discontinuity and a rarefaction wave; whilst in Lax's problem, the initial discontinuity breaks into a moderately strong shock followed by a density level far higher than its initial value. The contact discontinuity lowers the density, which is then rebuilt slowly by the rarefaction wave. Numerical results for the two shock tube problems of Sod and Lax are compared with exact solutions which are available, for example, from [25].

Since it can be rather time consuming to estimate the maximum stable time step from the Courant-type condition

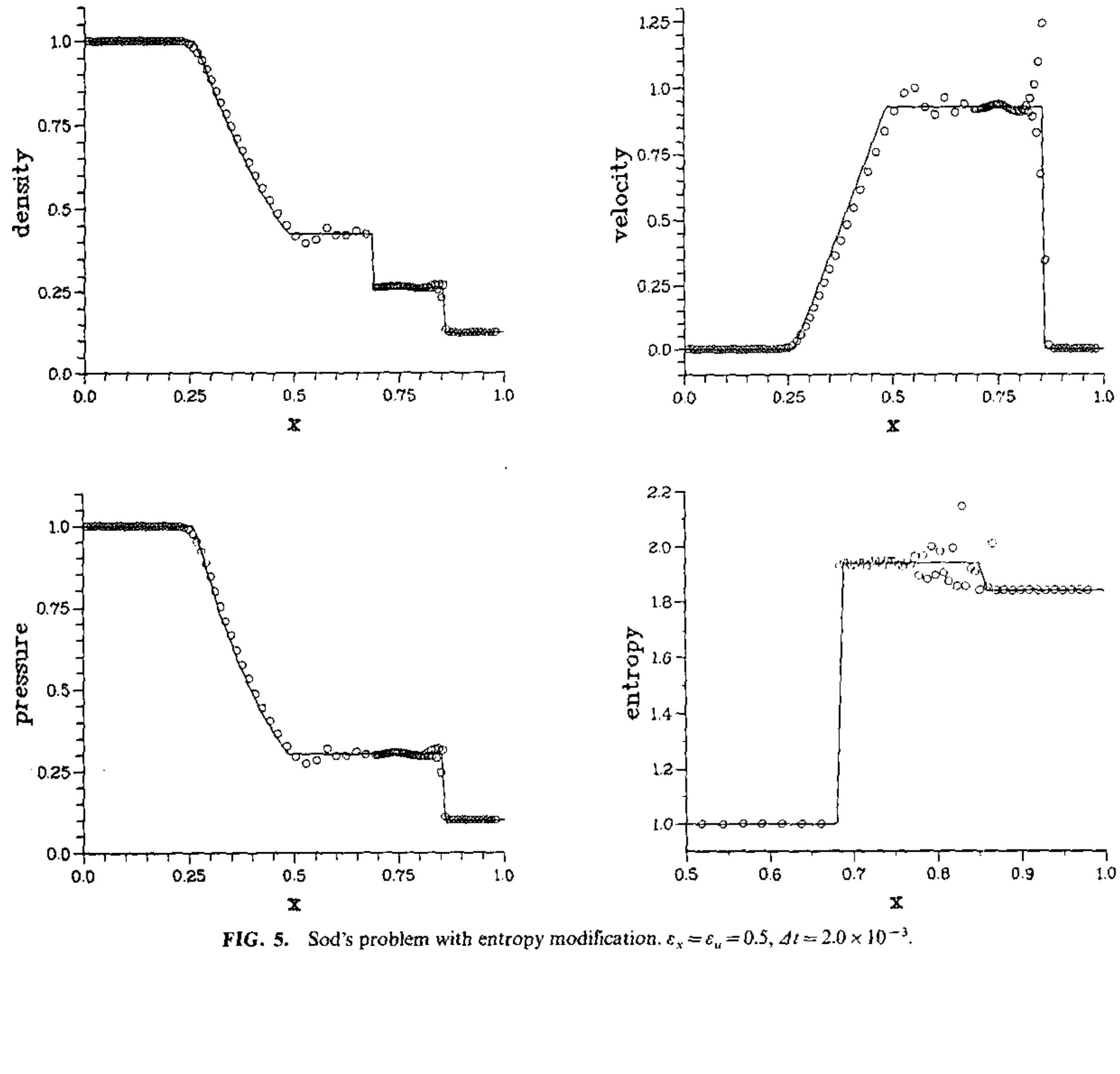


FIG. 5. Sod's problem with entropy modification. $\varepsilon_x = \varepsilon_u = 0.5$, $\Delta t = 2.0 \times 10^{-3}$.

of Eq. (43), we have used constant time steps to compute the evolution of physical variables as presented below. In all the four examples considered in this section, uniform initial cell sizes of 0.01 are used in the calculations. All the calculations were performed on a SUN SPARC workstation. In the first two problems, the exact solution is plotted as a solid line and the computed results are denoted by ◦. The CFL condition numbers mentioned below, for Sod's and Lax's shock tube problems, are calculated using the initial cell size and the exact solutions, i.e.,

$$\mathrm{CFL} = \frac{\Delta t \times (a+v)_{\max}}{\Delta_i^0},$$

where a is the local sound speed.

5.1. *Sod's Shock Tube Problem*

In Sod's problem, the initial condition (at $t = 0.0$) is defined by

	$0 \leqslant x \leqslant \frac{1}{2}$	$\frac{1}{2} < x \leqslant 1$
ρ	1.0	0.125
u	0.0	0.0
p	1.0	0.1

The numerical results are shown at $t = 0.2$. Numerical experiments show that for Sod's problem the maximum time step $\Delta t_{\max}$ (see Fig. 3) is 2.4×10^{-3}, i.e., the CFL condition number is 0.52.

Figure 5 shows the results calculated by using $\Delta t = 2 \times 10^{-3}$, $\varepsilon_x = \varepsilon_u = 0.5$. The corresponding CFL condition number is 0.44. It is seen that enormous numerical oscillations occur between the shock front and the right endpoint of the rarefaction, especially behind the shock. The constant states between the right endpoint of the rarefaction and the shock can be only partially realised for pressure and density and cannot be realised at all for velocity and entropy. However, the rarefaction wave is calculated accurately. The reason for these numerical results is that when $\varepsilon_x = \varepsilon_u = 0.5$

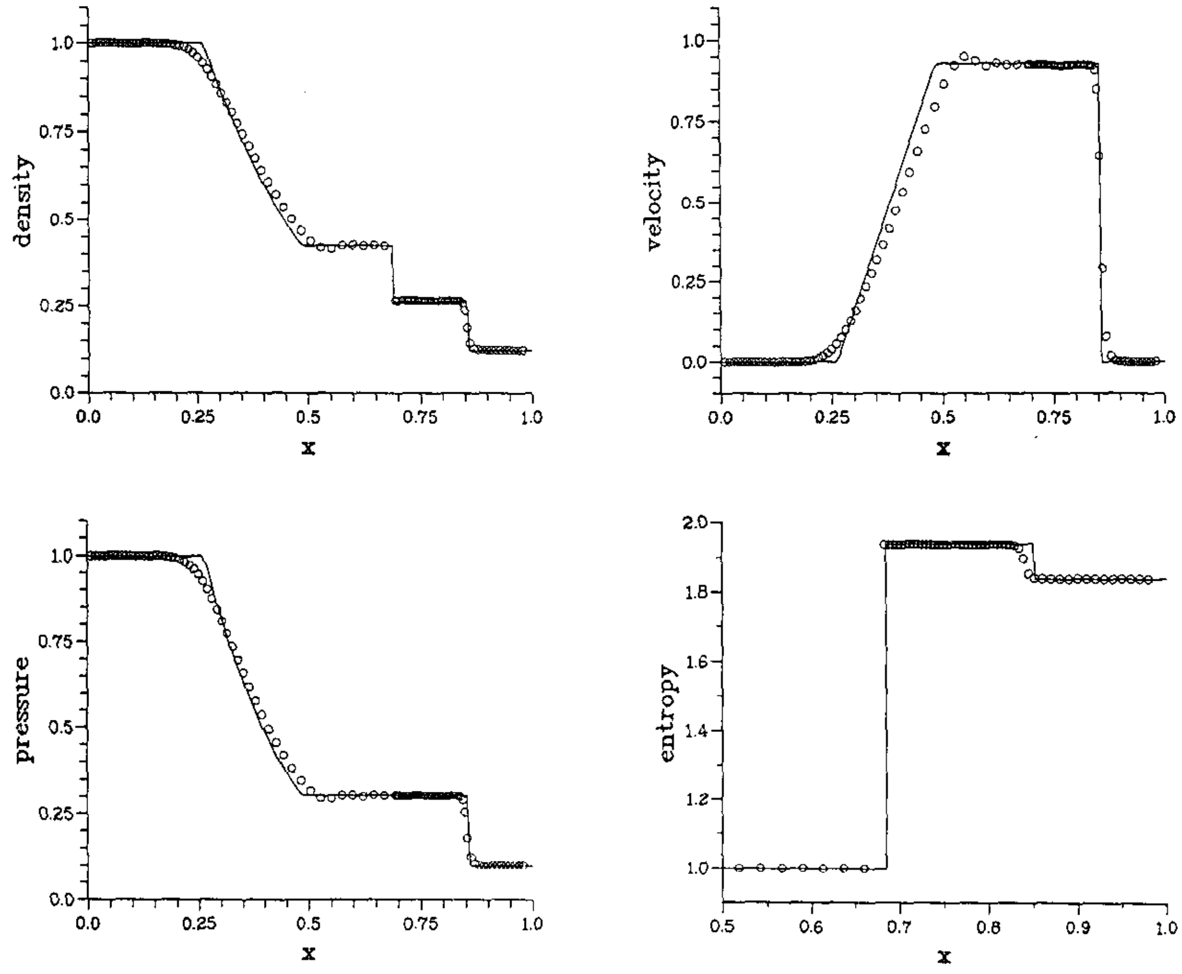


FIG. 6. Sod's problem with entropy modification. $\varepsilon_x = \varepsilon_u = -1.2$, $\Delta t = 1.0 \times 10^{-3}$.

is used there is no internal dissipation to dampen the high-frequency components generated at the shock and to round the endpoints of the rarefaction. This oscillatory behaviour behind the shock is very similar to the two-step Lax–Wendroff method with an added artificial viscosity as shown by Sod [4]. Note that the Lagrangian algorithm still gives infinite resolution of the contact discontinuity.

It is clear that numerical dissipation is required in order to dampen these oscillations. There are several ways to achieve this, as can be seen from Eqs. (32) or (38). Ideally we would wish to proceed either by increasing the time step Δt with fixed ε_x and ε_u, or by lowering ε_x and ε_u but keeping the time step Δt fixed. Unfortunately, these two ways are not feasible when the time step Δt is close to Δt_{max}, since there is only a very narrow region in which these parameters may vary within the required stability criterion (see Fig. 3). A careful analysis of Fig. 3 suggests that the decrease of time step Δt gives rise to a rapid increase in the stability region that ε_x and ε_u can vary within. Therefore, the effective way to gain more numerical dissipation is to lower the time step Δt and consequently use very small (even negative) ε_x and ε_u. The penalty paid by obtaining sufficient numerical dissipation to dampen high-frequency oscillations, however, is the cost of more computing time.

Results calculated by using $\Delta t = 1 \times 10^{-3}$ and $\varepsilon_x = \varepsilon_u = -1.2$ are shown in Fig. 6. The corresponding CFL condition number is 0.22. These results clearly demonstrate the improvement in accuracy at the discontinuities. Only very slight numerical oscillations can be observed in the velocity profile. However, as a consequence of achieving sufficient numerical dissipation for the discontinuities, the shock is spread over four to five cells instead of one to two cells (see Fig. 5) and the endpoints of the rarefaction are noticeably rounded. Note that the algorithm gives the correct entropy increase across the shock and the correct values of physical variables at constant states. We also observe that the numerical dissipation does not alter the positions of the shock and the contact discontinuity. We stress that Eulerian calculations, including the high resolution schemes, cannot achieve this infinite resolution at the contact discontinuity.

Figure 7 shows results calculated by using the same parameters as Fig. 6 but without taking into account the entropy change across the shock. These results are very

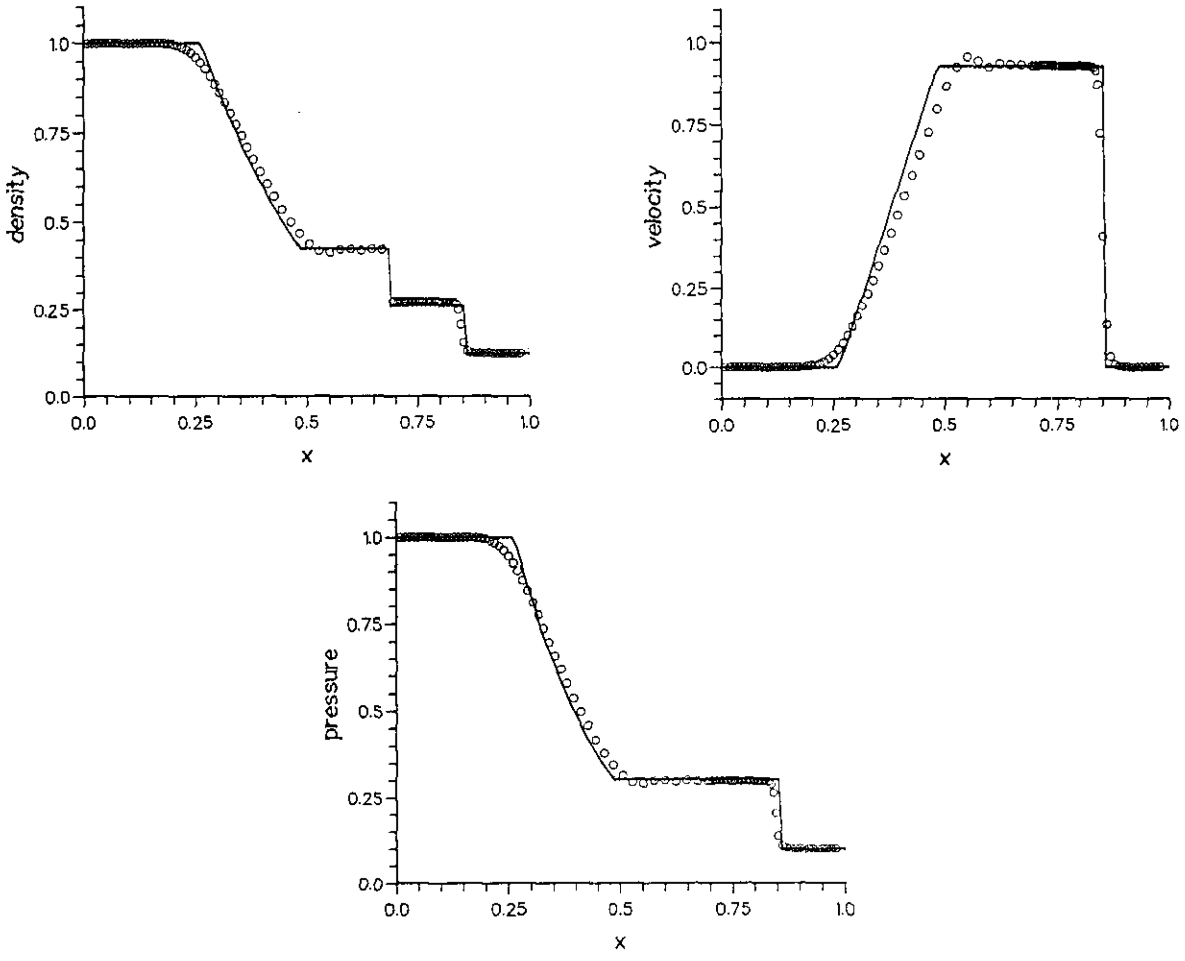


FIG. 7. Sod's problem without entropy modification. $\varepsilon_x = \varepsilon_u = -1.2$, $\Delta t = 1.0 \times 10^{-3}$.

TABLE I

Comparison of CPU Time

	Fig. 6 (with entropy modification)	Fig. 7 (without entropy modification)
Cpu time	3.6 s	3.1 s

similar in general to those presented in Fig. 6. However, a close look at the results in Fig. 7 reveals that the values of physical variables at constant states are slightly inaccurate. Density and velocity are overpredicted and the pressure is underpredicted. A calculation based on the predicted values of physical variables at constant states indicates that the velocity of the contact discontinuity is slightly higher than the exact solution, whilst the velocity of the shock front is slightly lower than the exact value. A practical conclusion drawn from these results is that the isentropic assumption is reasonably good for weak shocks.

To demonstrate the effect of entropy modification on the computing time, the cpu times for obtaining numerical results shown in Figs. 6 and 7 are compared in Table I. It is seen that there is about 16% increase in cpu time when the entropy modification is carried out.

5.2. *Lax's Shock Tube Problem*

The initial condition (at $t = 0.0$) for Lax's problem is defined as

	$0 \leqslant x \leqslant \frac{1}{2}$	$\frac{1}{2} < x \leqslant 1$
ρ	0.445	0.5
u	0.698	0.0
p	3.528	0.571.

The numerical results are shown at $t = 0.1$. For this problem, our numerical experiments show that the maximum time step $\Delta t_{\max}$ is 1.45×10^{-3}, which corresponds to a CFL condition number of 0.45.

Numerical results shown in Fig. 8 are obtained using $\Delta t = 1.0 \times 10^{-3}$ (CFL number is 0.32), $\varepsilon_x = \varepsilon_u = 0.0$, and

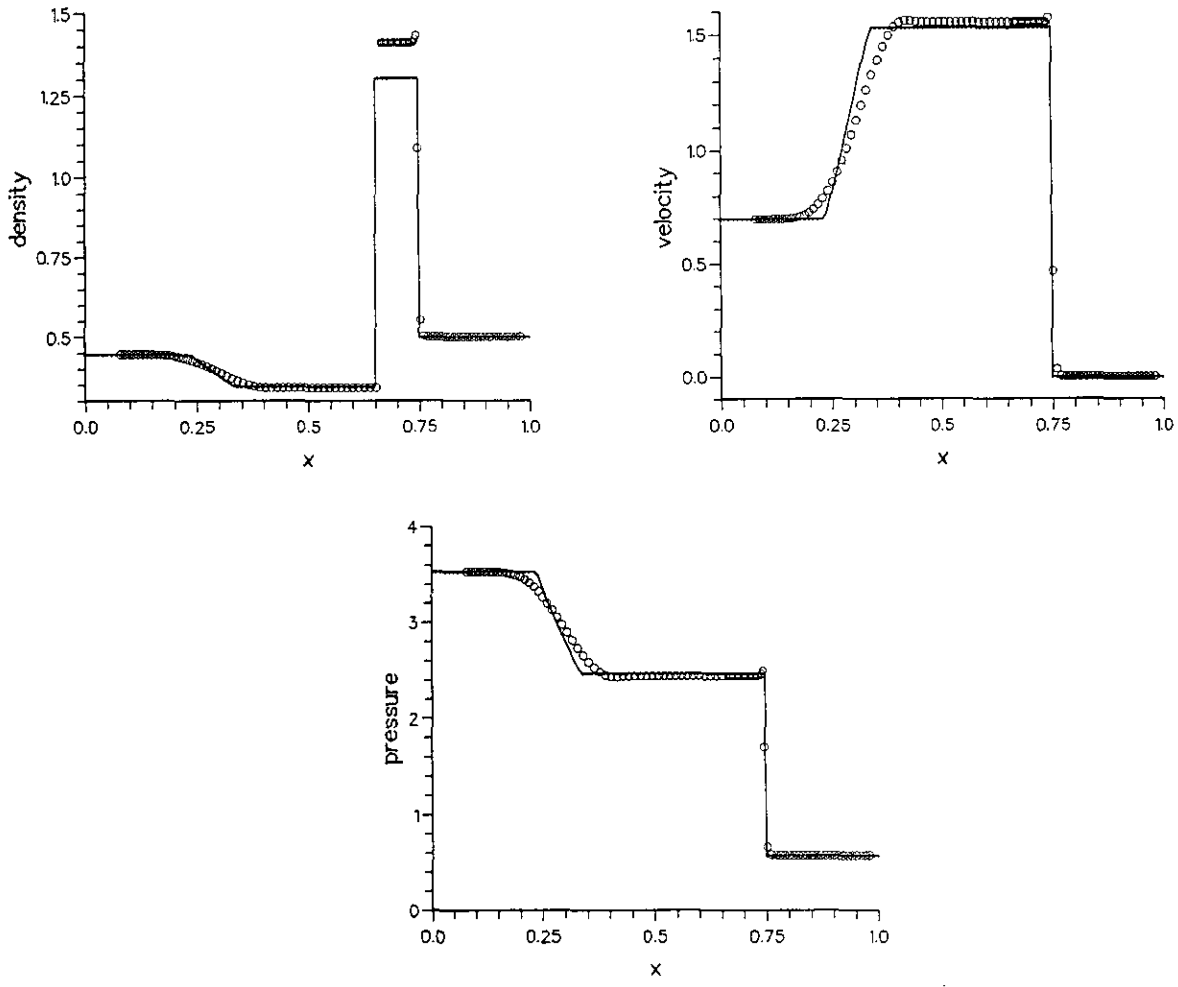


FIG. 8. Lax's problem without entropy modification. $\varepsilon_x = \varepsilon_u = 0.0$, $\Delta t = 1.0 \times 10^{-3}$.

without entropy modification. Because of the presence of the inherent numerical dissipation, there are only slight overshoots just behind the shocks and the endpoints of the rarefaction are noticeably rounded. The shock transition, however, occupies only one to two cells and this is believed to be caused by the stronger natural compression mechanism of the shock than that in Sod's problem (see Fig. 7). The constant states can be clearly realised; however, the values of physical variables at constant states differ from the exact values, especially for density, when using the isentropic approximation. Based on mass conservation across these discontinuities and the computed values of density and velocity, it can be shown that the velocity of the contact discontinuity is slightly overpredicted and the velocity of the shock front is slightly underpredicted, as in Sod's problem.

The numerical results computed using $\Delta t = 2.0 \times 10^{-4}$ and $\varepsilon_x = \varepsilon_u = 0.0$ and without entropy modification show that violent numerical oscillations occur behind the shock (they are not depicted here). These results are expected since the use of a much smaller time step with a fixed ε_x (or ε_u), compared with Fig. 8, reduces the magnitude of the built-in numerical dissipation and leaves the numerical oscillations generated by the shock insufficiently damped.

Results shown in Fig. 9 are calculated by using the same time step and explicitness parameters as those used in Fig. 8, but with the entropy of the shocked flow modified using the Rankine–Hugoniot condition. These results are very similar to those presented in Fig. 8; the shock is spread over one to two cells, the rarefaction is rounded to the same extent, the constant states are clearly realised, and there are slight overshoots/undershoots behind the shock. Unlike the results of Fig. 8, however, the values of physical variables at constant states are now in excellent agreement with the exact solutions and the correct entropy change of the shocked flow is predicted. We observe that the magnitude of the overshoots/undershoots is slightly greater than that found in Fig. 8.

We have shown in the calculation of Sod's problem that it is feasible to achieve sufficient numerical dissipation in order to eliminate numerical oscillations behind the shock

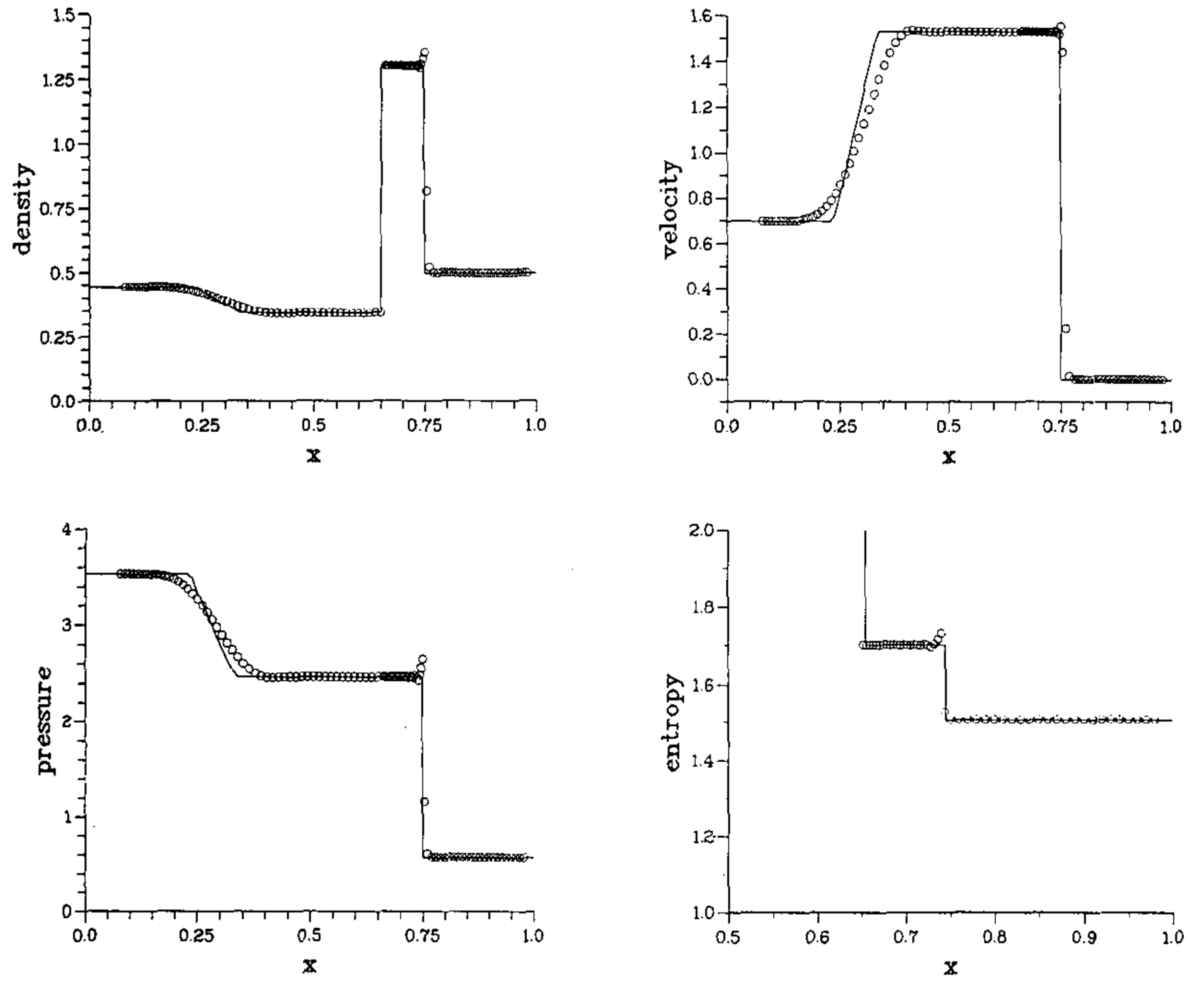


FIG. 9. Lax's problem with entropy modification. $\varepsilon_x = \varepsilon_u = 0.0$, $\Delta t = 1.0 \times 10^{-3}$.

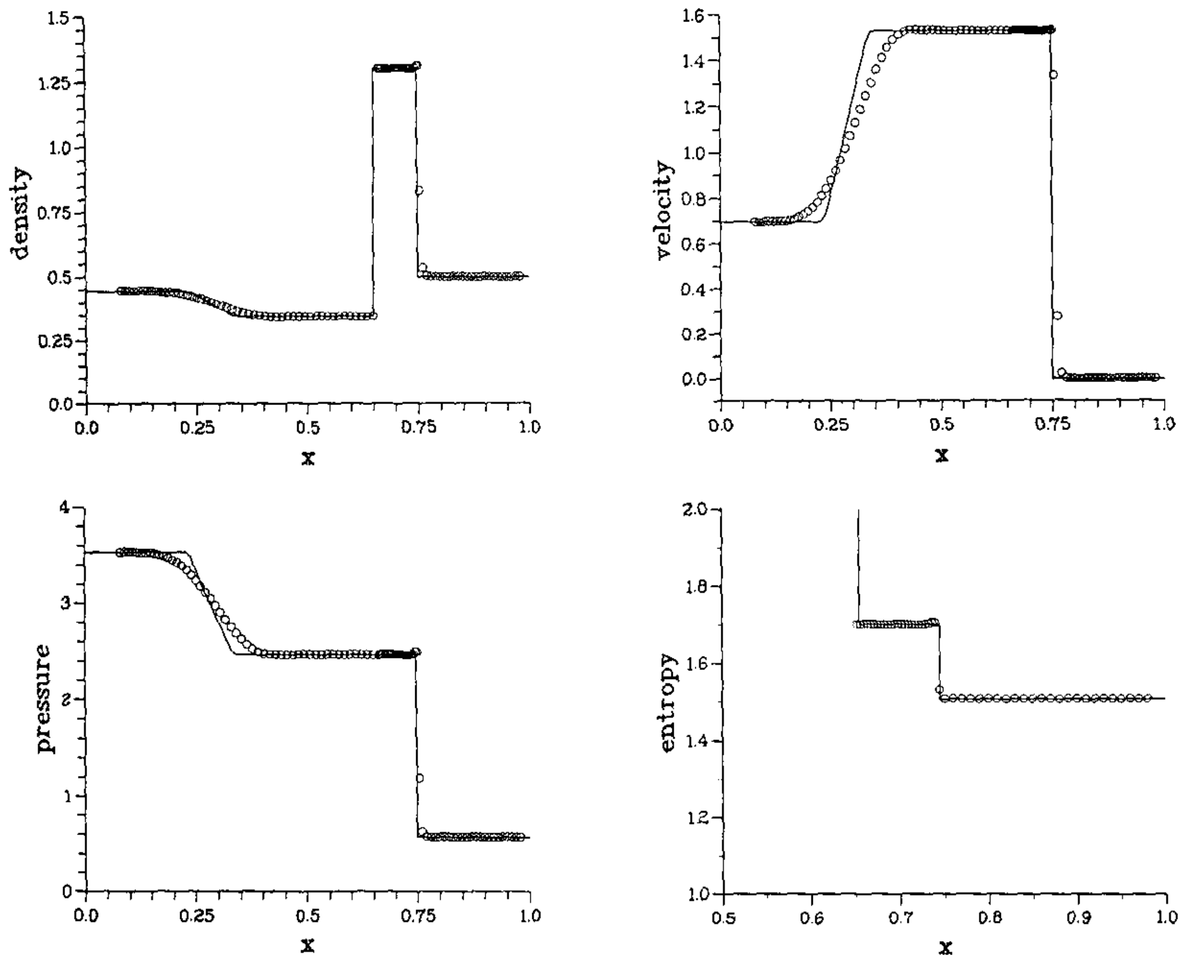


FIG. 10. Lax's problem with entropy modification. $\varepsilon_x = \varepsilon_u = -1.0$, $\Delta t = 5.0 \times 10^{-4}$.

by using a smaller time step and smaller (even negative) explicitness parameters. For Lax's problem, we have also performed a calculation using smaller Δt and ε_x and ε_u, aiming to eliminate the numerical oscillations observed in Fig. 9. The results of this calculation are shown in Fig. 10, where the parameters used are $\Delta t = 5.0 \times 10^{-4}$ and $\varepsilon_x = \varepsilon_u = -1.0$; the corresponding CFL number is 0.16. It is seen that the numerical oscillations are further damped and can hardly be observed. The endpoints of the rarefaction and the shock front are also slightly further smoothed out, compared to Fig. 9.

5.3. *Response of a Planar Flame to a Harmonic Pressure Wave*

In this example we consider the response of a density interface to an initially sinusoidal pressure disturbance. The problem arises from the study of the interaction at the very early stage between a planar premixed flame and a long length-scale pressure wave.

On an acoustic time scale, the planar flame front can be essentially treated as a density discontinuity and is convected with the gas flow driven by a pressure disturbance [26]. Under these circumstances the governing equations reduce to the equations of gasdynamics.

The initial conditions (at $t = 0.0$) are shown in Fig. 11. The pressure of the burned gas side (low density) is

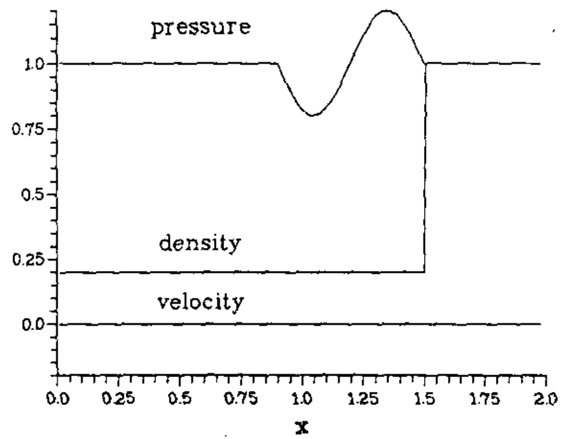


FIG. 11. Initial conditions for studying the response of a density discontinuity to a harmonic pressure wave.

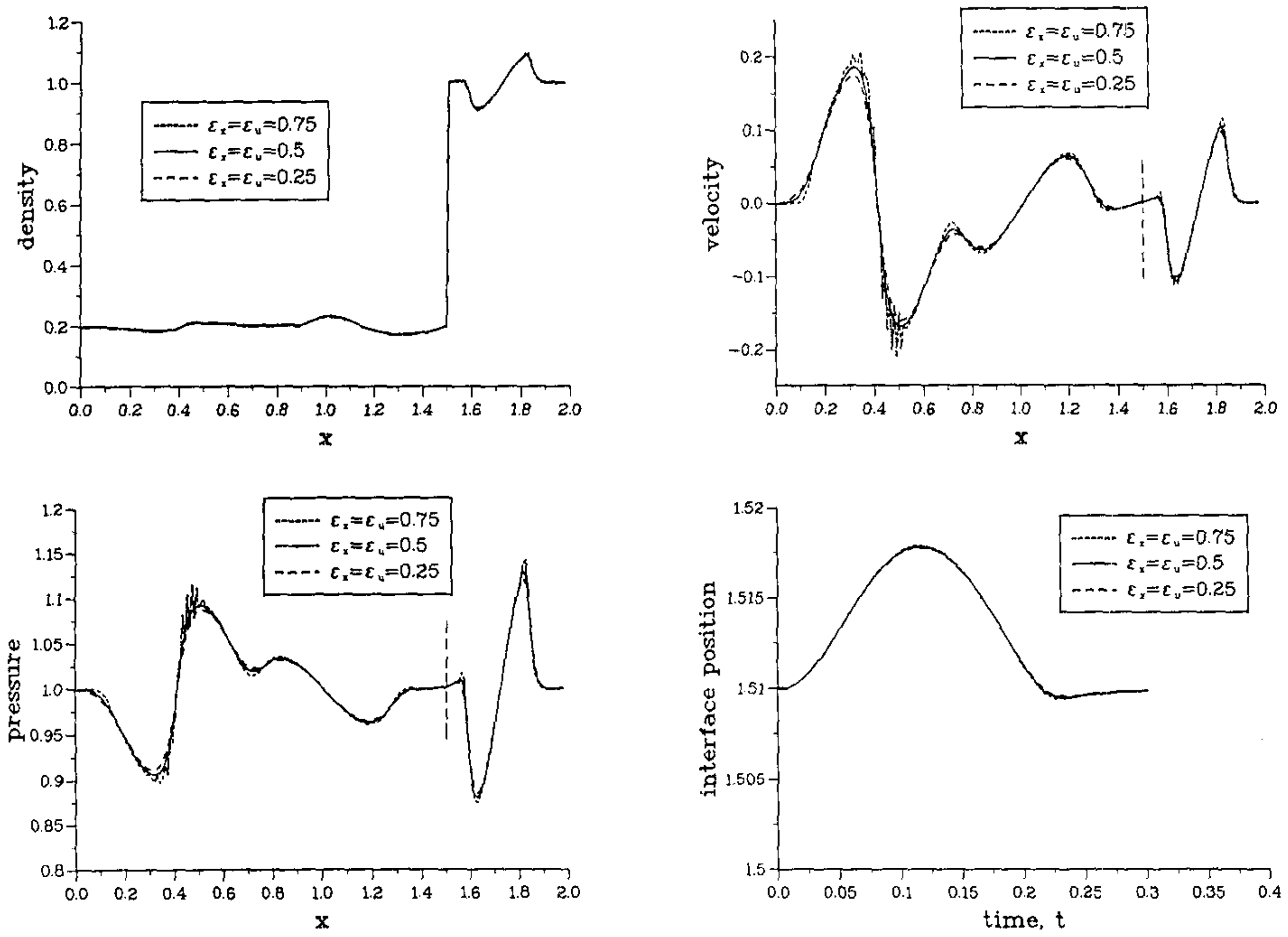


FIG. 12. Physical variables at $t = 0.3$ and the time history of the flame front. $\Delta t = 1.5 \times 10^{-3}$.

disturbed by a long length-scale sinusoidal distribution of one cycle. The time step Δt used in the calculations is 1.5×10^{-3}.

We have used three different values of explicitness parameters ε_x (or ε_u) in the calculation of this example to investigate its effect on the numerical results. Numerical results are shown in Fig. 12, where the physical variables, i.e., density, velocity, and pressure, are shown at $t = 0.3$. The oscillatory behaviour of the algorithm when $\varepsilon_x = \varepsilon_u = 0.75$ is used can be clearly seen at strong gradients. In the first two examples, which contain a shock in the flow, the use of explicitness parameters greater than 0.5 leads to numerical instabilities very rapidly. In this example, however, the numerical instability caused by the negative numerical dissipation grows gradually with time since there is no non-linear discontinuity (shock) present in the flow. The figure also clearly shows the smoothing out of strong gradients caused by the built-in numerical dissipation when $\varepsilon_x = \varepsilon_u = 0.25$ is used. The value of the explicitness parameters, however, has a negligible effect on the variation of the interface position with time for the values used here. The results of this example suggest that the use of $\varepsilon_x = \varepsilon_u = 0.5$ yields the most accurate solutions and should be used for problems which do not contain non-linear discontinuities (i.e., shocks).

5.4. *Propagation of a Planar Pressure Pulse*

In this test problem the initial pressure disturbance will break into two propagating waves in the positive and negative directions, respectively. The initial conditions are defined by $\rho = 1.0$, $u = 0.0$, $p = 1.0 + 0.5 \times \operatorname{sech}(z)$, where $z = 50 \times (x - 0.5)$ and $0 \leqslant x \leqslant 1.0$. Based on the conclusion of the last example, it is preferable to use $\varepsilon_x = \varepsilon_u = 0.5$ to perform the calculation of the Lagrangian algorithm.

The results of the Lagrangian calculation at $t = 0.2$ are compared with those calculated by the first-order upwind scheme of Steger and Warming [1] in Fig. 13. The time step

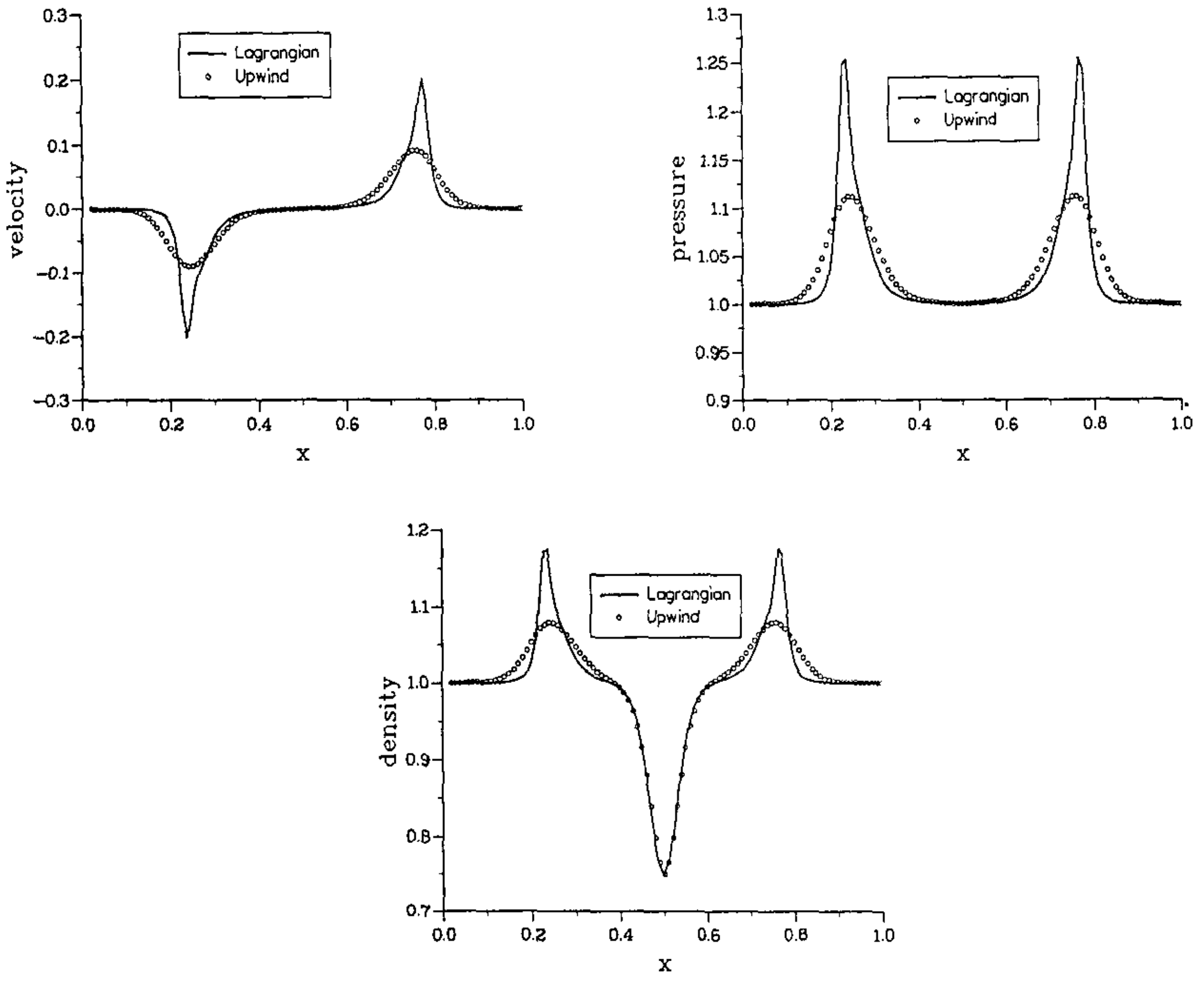


FIG. 13. Profiles of density, velocity, and pressure at $t = 0.2$.

used in these calculations is $\Delta t = 2.0 \times 10^{-3}$. As can be expected, the first-order upwind scheme greatly smooths out the strong gradients, while the Lagrangian algorithm can maintain these strong gradients.

6. APPLICATIONS OF THE TWO-DIMENSIONAL LAGRANGIAN ALGORITHM

In this section we first apply the two-dimensional Lagrangian algorithm described in Section 4 to two gasdynamic problems in order to evaluate the capability and accuracy of the method. The first test is the propagation of a circular pressure pulse. The second is the shock problem of Sod with a circular setup of initial conditions. Lastly, at the end of this section, we use the method to study the early stage interaction between a pressure wave and a curved flame. The expected Rayleigh–Taylor instability is clearly demonstrated by our Lagrangian method.

The specific heat ratio γ is assumed to be 1.4. Unless otherwise stated, the solution domains are divided into a

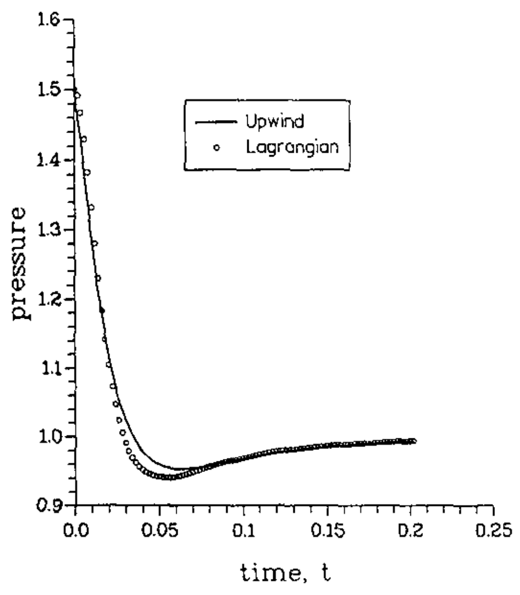


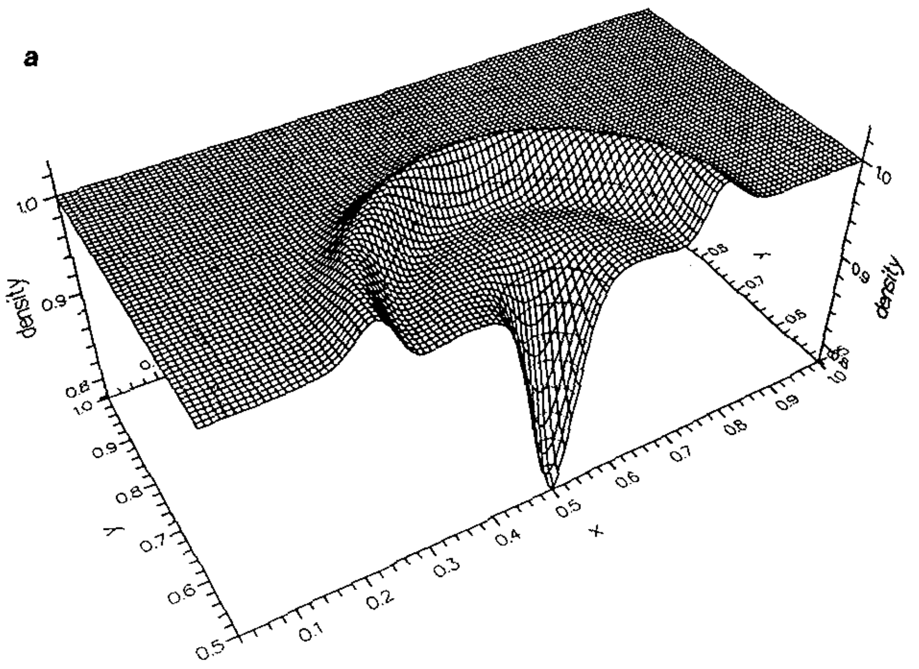
FIG. 14. Evolution of pressure at the pulse centre with time.

number of uniform squares with $\Delta x = \Delta y = 0.01$ and the time increment $\Delta t = 2 \times 10^{-3}$.

6.1. *Propagation of a Circular Pressure Pulse*

The solution domain $[0, 1] \times [0, 1]$ is divided into 100×100 uniform squares. A circular pressure pulse is set up around point $(0.5, 0.5)$ in terms of $p = 1 + 0.5 \times \operatorname{sech}(z)$, where $z = 50 \times \sqrt{(x_{i,j} - x_{50,50})^2 + (y_{i,j} - y_{50,50})^2}$. Initial conditions for density and velocities are such that $\rho = 1$, $u = v = 0$ everywhere. Explicitness parameters of $\varepsilon_x = \varepsilon_u = 0.5$ are used in the calculation to make the algorithm non-diffusive and to achieve higher accuracy.

The evolution of the pressure at the centre, point $(0.5, 0.5)$, with time is shown in Fig. 14. It is interesting that the pressure at the pulse centre drops rapidly below unity and then recovers gradually. Again the result of the first-order upwind scheme of Steger and Warming is diffusive



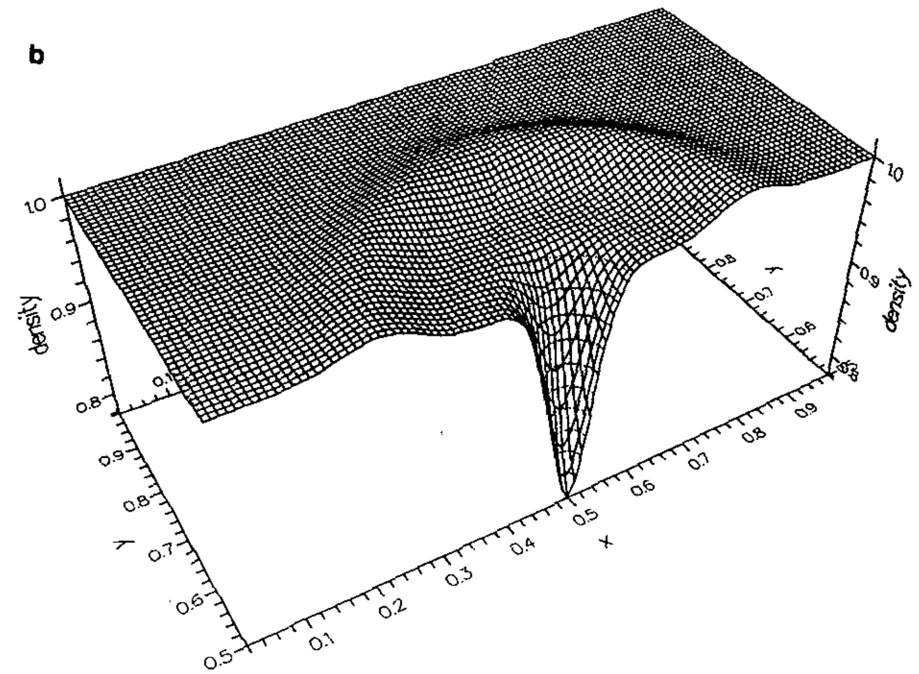


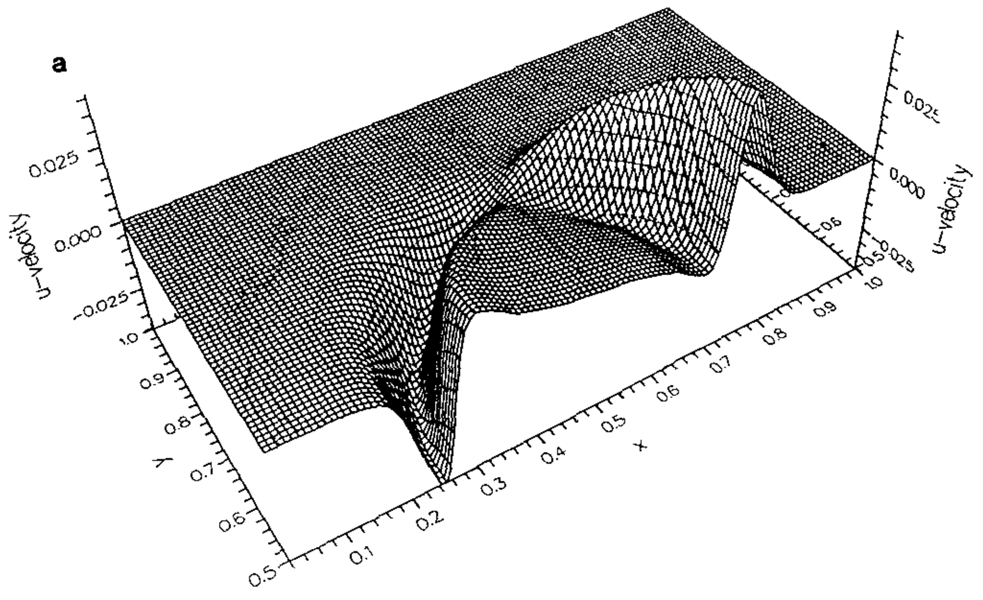
FIG. 15. Comparison of density profiles at $t = 0.2$: (a) Lagrangian; (b) upwind.

near the minimum of the centre pressure, i.e., $t = 0.05$; elsewhere the results of the two methods are in very good agreement.

The results of density, u-velocity, and pressure at $t = 0.2$ are compared with those of the upwind scheme in Figs. 15, 16, and 17, respectively. In general, the profiles of these physical quantities calculated by the Lagrangian method follow very closely the results of the upwind scheme. There are, however, still some noticeable differences between the results of the two methods. In Fig. 15, the Lagrangian algorithm predicts a rather strong density gradient at the wave front; however, the upwind scheme predicts a very weak density gradient at the wave front as the consequence of numerical diffusion. The smearing of the velocity and pressure gradients at the wave front in the results of the first-order upwind scheme can also be clearly observed in Figs. 16 and 17.

6.2. *Circular Version of Sod's Problem*

For this test problem the solution domain [0, 1] × [0, 1] is uniformly divided into 100 × 100 squares. The initial conditions are those of the planar version given in Section 5.1 with $v = 0$ everywhere; the region of high pressure and high



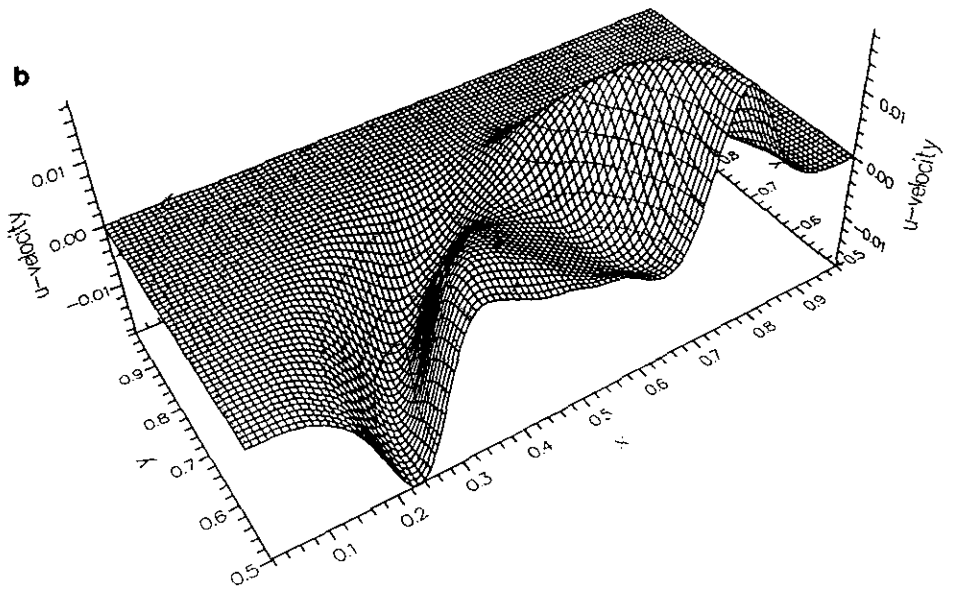
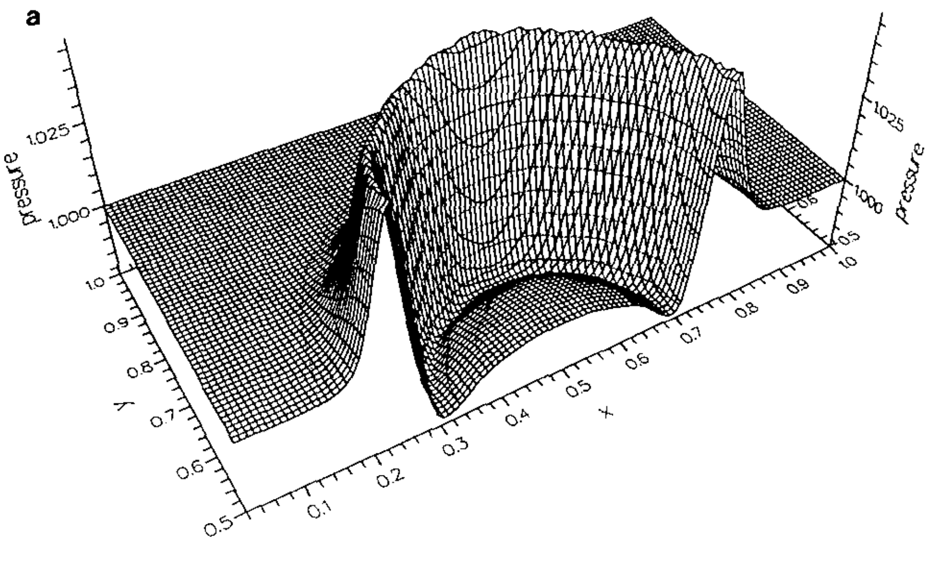


FIG. 16. Comparison of u-velocity profiles at $t = 0.2$: (a) Lagrangian; (b) upwind.



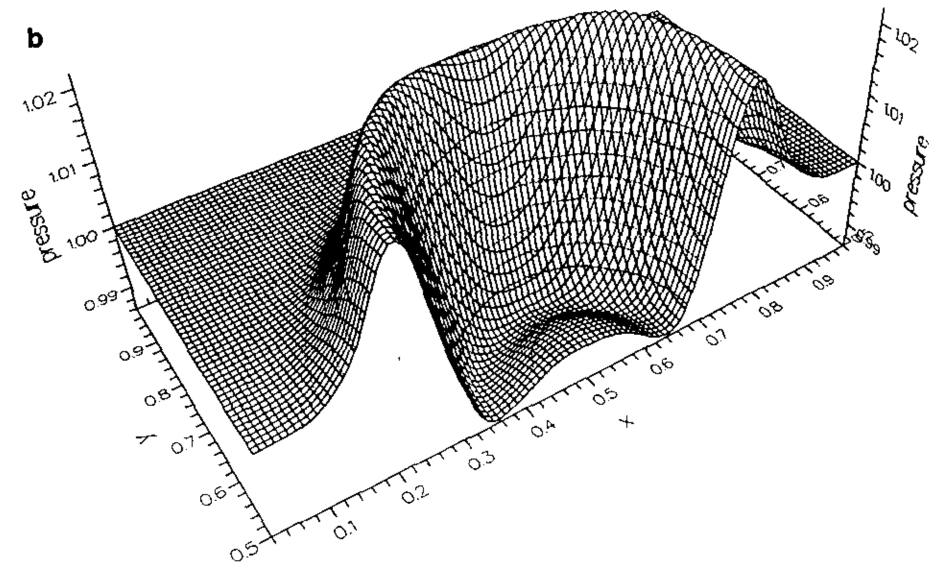


FIG. 17. Comparison of pressure profiles at $t = 0.2$: (a) Lagrangian; (b) upwind.

density in this case is a circle of radius 0.15 centred at (0.5, 0.5). This problem consists of three waves, an outward travelling shock followed by a contact discontinuity and a rarefaction travelling toward the centre. The following parameters are used in the calculation of this problem: $\Delta t = 2 \times 10^{-3}$, $\varepsilon_x = \varepsilon_u = 0.0$. Figure 18a shows the density profile in a quarter of the domain at $t = 0.1$ and Fig. 18b shows the pressure profile at the same time. Note that the slight asymmetry of the circular waves is not unexpected because the initial circular boundary of the high pressure region is approximated by cartesian step gridlines. Both the shock and the contact discontinuity are sharply resolved. These results are comparable with those presented by Toro [9] using the weighted average flux (WAF) method. Note that the isentropic condition is used in obtaining the results of Fig. 18. The values of variables behind the shock are expected to be slightly inaccurate.

6.3. *Rayleigh–Taylor Instability in Flame Propagation*

On an acoustic time scale, the equations governing the interaction between a pressure wave and a premixed flame reduce to the equations of gasdynamics except that the momentum equations are modified to

$$\frac{du}{dt} = -\frac{1}{\rho\gamma}\frac{\partial p}{\partial x}, \tag{62}$$

$$\frac{dv}{dt} = -\frac{1}{\rho\gamma}\frac{\partial p}{\partial y}. \tag{63}$$

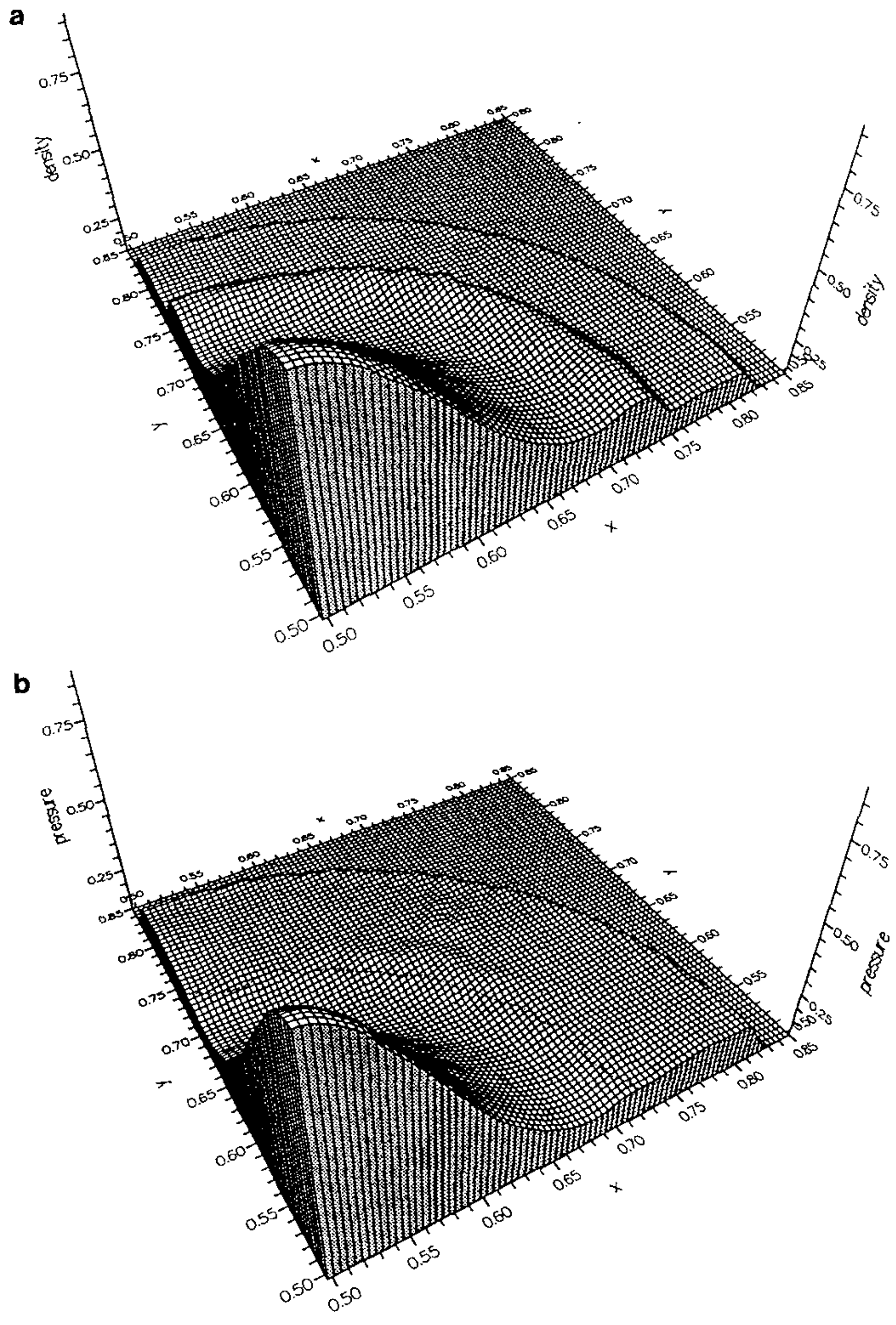


FIG. 18. Density (a) and pressure (b) contours of circular version of Sod's problem at $t = 0.1$.

In these equations, u and v are non-dimensionalised by the sound speed of cold gas a, coordinates x and y by a typical acoustic length scale L, time t by the corresponding time scale L/a, and pressure p and density ρ by the values for cold gas, respectively.

The solution domain $[0.0, 2.5] \times [0.0, 0.85]$ is uniformly divided into 250×85 squares. The region $[0.0, 2.0] \times [0.0, 0.85]$ is filled with hot gas with density $\rho = 0.2$ while the region $[2.01, 2.5] \times [0.0, 0.85]$ is filled with cold gas with density $\rho = 1.0$. A planar linear pressure disturbance is

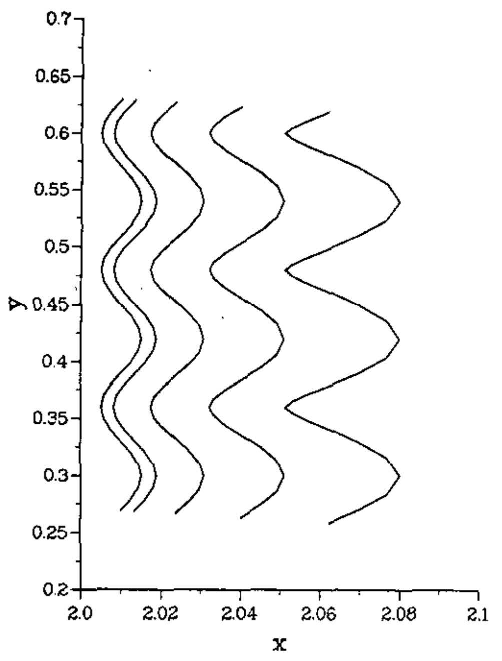


FIG. 19. Outline of the flame front at five different times: $t = 0.0, 0.16, 0.32, 0.48, 0.64$.

set up in the hot gas side in terms of $p = 1 + \nabla p \times (x - 2.0)$, where $\nabla p = -0.25$ and $0 \leqslant x \leqslant 2.0$. The initial pressure of cold gas is unity everywhere. Both hot and cold gases are stationary at time $t = 0$. This assumption reflects the fact that the burning velocity of a laminar flame is much slower than the sound speed [26]. In order to investigate the growth of a flame front disturbance under the action of this pressure wave, the density interface at $x = 2.01$ is disturbed to a sinusoidal distribution with wavelength $\lambda = 0.12$ and amplitude $a_0 = 0.005$. The time increment and explicitness parameters used in this calculation are such that $\Delta t = 2 \times 10^{-3}$ and $\varepsilon_x = \varepsilon_u = 0.5$.

Figure 19 shows a portion of the density interface (flame front) at time $t = 0$, 0.16, 0.32, 0.48, and 0.64. The acceleration and deformation of the flame front under the action of the pressure disturbance are clearly represented by the numerical results. The computation cannot go further because of the large fluid distortions at the flame front. On the other hand, at a longer time the numerical results do not have any physical significance since this simplified gasdynamic model no longer holds true. This work is developed in references [11] and [27].

7. CONCLUDING REMARKS

A simple and flexible implicit Lagrangian algorithm that can predict one- and two-dimensional gasdynamic problems involving shocks and contact discontinuities has been presented. Because of the way that the present method is constructed, it is capable of strictly conserving the mass and entropy of each Lagrangian computational cell throughout its evolution.

When the flow of interest contains a shock, it is necessary to make the algorithm dissipative in order to dampen the high-frequency components of oscillations generated by the shock. The magnitude of the built-in numerical dissipation can be controlled easily by varying the values of the time step and explicitness parameters, within the stability limit. The use of smaller (or even negative) explicitness parameters leads to more numerical dissipation but also reduces the time step allowed by the stability criterion. The maximum stable time step clearly depends on the shock strength, due to shock compression; the stronger the shock, the smaller the maximum stable time step.

Numerical experiments for Sod's and Lax's problems show that sufficient numerical dissipation can be achieved to eliminate numerical oscillations behind the shocks when the time step used corresponds to a CFL condition number of one-third to one-fourth, and some negative value for ε_x and ε_u is used. Under these circumstances, the numerical dissipation is excessive for the rarefaction since their endpoints are considerably rounded. The present Lagrangian method does not give particularly good results for the Riemann problems considered here when compared to high resolution schemes (such as TVD schemes). However, it is the suitable method to study problems where contact discontinuities are of prime interest such as examples 5.3 and 6.3 with which we are concerned in this study.

For problems which do not involve non-linear discontinuities (shocks), it is preferable to use $\varepsilon_x = \varepsilon_u = 0.5$ in the present algorithm. The method has been successfully applied to predict the response of a planar and a curved density discontinuity, which simulates a premixed flame front on an acoustic time scale, to a pressure disturbance.

Although in this work we have used a constant explicitness parameter ε_x (or ε_u) throughout a computation, it is possible to vary ε_x and ε_u from cell to cell or from time step to time step. The algorithm can also be extended to one-dimensional cylindrical and spherical coordinates with ease.

The method in two-dimensions, like any other Lagrangian methods, suffers from the severe problem of grid distortions. But the method can still find its application in problems where a contact discontinuity and short time effects are of particular interest, which is the physical situation we are concerned with here. An Eulerian calculation is certainly preferable if one is interested in long time effects.

Finally we point out that it is possible to overcome the two main shortcomings of Lagrangian methods, namely the necessity to run at a relatively small CFL number, and the numerical instability caused by interface crossings in one dimension and grid distortions in two and three dimensions. This can be done by employing a Lagrangian calculation locally around a contact discontinuity, since it can predict excellent resolution nearby; regions away from the contact discontinuity can then be solved by an Eulerian method [17, 28].

ACKNOWLEDGMENTS

We are very grateful for referees' constructive criticism and comments on the preliminary version of this paper. The financial support of this work by the Science and Engineering Research Council of the U.K. is gratefully acknowledged.

REFERENCES

1. J. L. Steger and R. W. Warming, *J. Comput. Phys.* **40**, 263 (1981).
2. S. K. Godunov, *Math. Sb.* **47**, 271 (1959) [Russian]; translated by U.S. Joint Publ. Res. Service, JPRS 7226, 1969.
3. J. von Neumann and R. D. Richtmyer, *J. Appl. Phys.* **21**, 232 (1950).
4. G. A. Sod, *J. Comput. Phys.* **27**, 1 (1978).
5. A. J. Chorin, *J. Comput. Phys.* **22**, 517 (1976).
6. A. Harten and J. M. Hyman, *J. Comput. Phys.* **50**, 235 (1983).
7. P. D. Lax, *Commun. Pure Appl. Math.* **7**, 159 (1954).
8. C. W. Shu and S. Osher, *J. Comput. Phys.* **83**, 32 (1989).
9. E. F. Toro, *J. Comput. Phys.*, submitted (1990).
10. P. K. Sweby, *SIAM J. Numer. Anal.* **21**, 995 (1984).
11. F. Liu, A. C. McIntosh, and J. Brindley, *Combust. Sci. Technol.* **91**, 373 (1993).
12. G. Taylor, *Proc. R. Soc. London A* **201**, 192 (1950).
13. C. L. Gardner, J. Glimm, J. W. Grove, O. McBryan, Q. Zhang, R. Menikoff, and D. H. Sharp, *Nucl. Phys. B* (*Proc. Suppl.*) **2**, 441 (1987).
14. C. W. Hirt, J. L. Cook, and T. D. Butler, *J. Comput. Phys.* **5**, 103 (1970).
15. R. K.-C. Chan, *J. Comput. Phys.* **17**, 311 (1975).
16. A. A. Amsden and C. W. Hirt, Los Alamos Scientific Laboratory Report LA-3425, 1966 (unpublished).
17. C. W. Hirt, A. A. Amsden, and J. L. Cook, *J. Comput. Phys.* **14**, 227 (1974).
18. L. Bilbao, *J. Comput. Phys.* **91**, 361 (1990).
19. J. P. Boris, NRL Memorandum Report No. 4022, Naval Research Laboratory, Washington, DC, 1979 (unpublished).
20. A. Lerat and R. Peyret, *Lecture Notes in Physics*, Vol. 35 (Springer-Verlag, New York/Berlin), p. 251.
21. R. M. Beam and R. F. Warming, *J. Comput. Phys.* **22**, 87 (1976).
22. P. D. Lax and B. Wendroff, *Commun. Pure Appl. Math.* **13**, 217 (1960).
23. T. H. Pulliam, *AIAA J.* **24**, 1931 (1986).
24. E. S. Oran and J. P. Boris, *Numerical Simulation of Reaction Flow* (Elsevier, New York, 1987).
25. H. W. Liepmann and A. Roshko, *Elements of Gasdynamics* (Wiley, New York, 1957).
26. A. C. McIntosh, *Combust. Sci. Technol.* **75**, 287 (1991).
27. N. R. Edwards, A. C. McIntosh, and J. Brindley, *Combust. Sci. Technol.*, submitted (1993).
28. H. S. Zhang and R. M. C. So, *Comput. Fluids* **20**, 421 (1991).