JOURNAL OF COMPUTATIONAL PHYSICS 104, 69–74 (1993)

Unconditionally Stable Explicit Algorithms for Nonlinear Fluid Dynamics Problems

JOHN L. RICHARDSON AND ROBERT C. FERRELL

Thinking Machines Corporation, 245 First Street, Cambridge, Massachusetts 02142

AND

LYLE. N. LONG

Pennsylvania State University, Department of Aerospace Engineering, University Park, Pennsylvania 16802

Received January 18, 1991; revised February 21, 1992

This paper describes novel explicit algorithms that are unconditionally stable. The algorithms are applied to some 1D convection and diffusion problems, including nonlinear problems. Algorithms such as these are of particular interest for massively parallel computers, where one is trying to minimize communications while at the same time maintain the stability properties normally associated with implicit schemes. It is shown how these stable algorithms can be applied in higher spatial dimensions and how they can be extended to problems defined on unstructured meshes. © 1993 Academic Press, Inc.

INTRODUCTION

There continues to be a great deal of debate over explicit versus implicit computational fluid dynamics (CFD) algorithms. Implicit schemes often exhibit unconditional stability for model equations, but involve more complex code and more computer time per iteration. Explicit algorithms are usually conditionally stable, but are relatively easy to program and require less time per iteration.

In discussing explicit versus implicit, one must differentiate between unsteady and steady-state problems. Usually the criteria that sets the maximum time step for explicit methods is the same criteria for time accuracy. That is, the time step must be small to obtain a time-accurate solution and small can mean of the same order as is required to maintain stability of an explicit scheme. So the larger time step possible in implicit methods often cannot be used if one is interested in time accurate solutions. Thus there is often little reason to use implicit methods for time accurate CFD problems.

However, when one is interested only in the steady-state solutions (large time behavior), implicit methods may offer advantages due to their ability to use large time steps. This is easy to demonstrate on model problems, but is often not realized on complicated problems. Typically, the more complicated the physics or the geometry of the problem the more often people will choose explicit methods. Industrial CFD codes have often been explicit, in order to minimize coding time and maximize the flexibility of the code.

Most stability analyses are for linear problems and ignore boundary condition effects. The effect of either nonlinearities or boundary conditions usually force the implicit algorithms to have time step restrictions that are not very different than modern explicit methods such as multi-grid Runge–Kutta with residual smoothing and pseudo-time marching. For unsteady problems the amplitude and phase errors are a better indication of accuracy than the truncation error.

Also, the ability to take larger time steps does not ensure that the solution will converge in fewer iterations. The amplification factor of the scheme and the work per time step is a better indication of the time to convergence. While one would like to have an amplification factor near unity for unsteady problems, to rapidly reach a steady state it is desirable to have amplification factors that are small (less than one). This causes the transient effects to rapidly decay.

The algorithm described below is an explicit scheme that is unconditionally stable [1, 2]. Explicit schemes exhibiting unconditional stability have been developed [8, 5], such as Saul'yev's alternating direction explicit (ADE) and the Dufort–Frankel algorithms. However, these techniques are unconditionally stable only for the heat equation. In addition, the ADE method is not readily parallelized due to its data dependencies. The scheme described below is unconditional stable when applied to convection and/or diffusion. It is also highly parallel.

This is an important development because new types of algorithms may well be required for the next generation of computers in order to take maximum advantage of their architectures. In particular, on massively parallel computers one would like to minimize the amount of long-range communication, which is required when using implicit methods [4].

DISCUSSION

In order to illustrate the behavior of these new unconditionally stable explicit methods, they will be applied to a number of model problems. The basic equations that will be solved will be the various forms of the nonlinear, viscous Burgers equation [3]:

$$\frac{\partial \mathbf{u}(\mathbf{x}, t)}{\partial t} + \mathbf{u}(\mathbf{x}, t) \cdot \nabla \mathbf{u}(\mathbf{x}, t) = \nu\, \nabla^2 \mathbf{u}(\mathbf{x}, t), \tag{1}$$

where $\nu \geqslant 0$. In one spatial dimension with $u(x, t) \rightarrow u(x, t) + c$ this reduces to

$$\frac{\partial u(x, t)}{\partial t} + [u(x, t) + c]\, \frac{\partial u(x, t)}{\partial x} = \nu\, \frac{\partial^2 u(x, t)}{\partial x^2}. \tag{2}$$

This equation includes a number of features found in fluid dynamic problems, including nonlinear convection (hyperbolic) and viscous diffusion (parabolic). By modifying the constants c and ν in the equation, one can investigate problems dominated by nonlinear effects, linear convection, or diffusion. One can also look at problems where the convection terms and the diffusion terms are both important. This equation also can simulate unsteady problems or steady-state problems. For example, with oscillatory boundary conditions one can investigate nonlinear wave propagation with or without diffusion. For steady boundary conditions, one can investigate phenomena such as long-time behavior representative of steady-state heat conduction problems, steady-state boundary layer-type problems, or steady-state shock wave problems.

When discussing the suitability of algorithms, it is important to address unsteady and steady problems separately, as well as convection and diffusion separately. The advantages and disadvantages of particular schemes will depend on which of these four cases one is investigating.

DIFFUSION

The diffusion equation in one dimension is given by

$$\frac{\partial u(x, t)}{\partial t} = \nu\, \frac{\partial^2 u(x, t)}{\partial x^2}. \tag{3}$$

If we discretize the spatial part to second order in the grid spacing Δx on a uniform periodic grid with an even number of sites, and approximate $u_j(t) \approx u(x = j\, \Delta x, t)$, we obtain

$$\frac{du_j}{dt} = \frac{\nu}{\Delta x^2}\,(u_{j+1} - 2u_j + u_{j-1}). \tag{4}$$

In matrix form with $\mathbf{u}(t) = [u_1(t), u_2(t), \ldots]^{\mathrm{T}}$, this becomes

$$\frac{d\mathbf{u}(t)}{dt} = \frac{\nu}{\Delta x^2} \begin{pmatrix} -2 & 1 & 0 & \cdots & 0 & 1 \\ 1 & -2 & 1 & 0 & \cdots & 0 \\ 0 & 1 & -2 & 1 & \cdots & 0 \\ \cdots & & & & & \end{pmatrix} \mathbf{u}(t) \tag{5}$$

Let

$$L = \begin{pmatrix} -2 & 1 & 0 & \cdots & 0 & 1 \\ 1 & -2 & 1 & 0 & \cdots & 0 \\ 0 & 1 & -2 & 1 & \cdots & 0 \\ \cdots & & & & & \end{pmatrix}. \tag{6}$$

Then, this can be split into two pieces $L = L_e + L_o$, where

$$L_e = \begin{pmatrix} -1 & 1 & 0 & 0 & \cdots & 0 \\ 1 & -1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & -1 & 1 & \cdots & 0 \\ 0 & 0 & 1 & -1 & \cdots & 0 \\ \cdots & & & & & \end{pmatrix} \tag{7}$$

and

$$L_0 = \begin{pmatrix} -1 & 0 & 0 & 0 & \cdots & 1 \\ 0 & -1 & 1 & 0 & \cdots & 0 \\ 0 & 1 & -1 & 0 & \cdots & 0 \\ 0 & 0 & 0 & -1 & \cdots & 0 \\ \cdots & & & & & \end{pmatrix}. \tag{8}$$

Now, the exact solution to the spatially discretized diffusion equation is

$$\mathbf{u}(t + \Delta t) = \exp(\alpha L)\, \mathbf{u}(t), \tag{9}$$

where $\alpha = \nu\, \Delta t/\Delta x^2$. It is the approximation of this expression that determines the integration algorithm. The first naive explicit algorithm is

$$\exp(\alpha L) = 1 + \alpha L + \mathcal{O}(\alpha^2), \tag{10}$$

which is stable provided that $\alpha \leqslant \frac{1}{2}$. The first implicit algorithm approximates

$$\exp(\alpha L) = [1 - \alpha L]^{-1} + \mathcal{O}(\alpha^2), \tag{11}$$

which is unconditionally stable but requires the expensive matrix inverse operation. The higher order implicit or Crank–Nicholson schemes are simply approximating

$$\exp(\alpha L) = \left[1 + \frac{\alpha}{2} L\right]\left[1 - \frac{\alpha}{2} L\right]^{-1} + \mathcal{O}(\alpha^3). \tag{12}$$

The exponentiated matrix $\exp(\alpha L)$ is dense, since it involves all possible powers of the tri-diagonal matrix L. However, we can approximate the exponential by splitting L into $L = L_e + L_o$ and writing

$$\exp(\alpha L) = \exp(\alpha L_e)\exp(\alpha L_o) + \mathcal{O}(\alpha^2). \tag{13}$$

The approximate time stepping operator,

$$\mathscr{P} \equiv \exp(\alpha L_e)\exp(\alpha L_o), \tag{14}$$

forms the basis for our unconditionally stable integration algorithm for the diffusion equation.[1] The splitting part of our algorithm is similar in spirit to the splitting up method of Marchuk [7]; however, his techniques are not explicit and require the solution of large sparse sets of linear equations at each time step.

To compute the matrix exponentials, note that both L_e and L_o are block diagonal and we need only consider the exponential of the two by two matrix

$$A = \alpha\begin{pmatrix} -1 & 1 \\ 1 & -1 \end{pmatrix}. \tag{15}$$

By using the series definition of the exponential function, we see that

$$\exp(A) = \left(\frac{1}{2}\right)\begin{pmatrix} 1 + e^{-2\alpha} & 1 - e^{-2\alpha} \\ 1 - e^{-2\alpha} & 1 + e^{-2\alpha} \end{pmatrix}. \tag{16}$$

Since both L_e and L_o are block diagonal, exponentiating them amounts to exponentiating A. Of course, both $\exp(\alpha L_e)$ and $\exp(\alpha L_o)$ are also block diagonal. Since the eigenvalues of $\exp(A)$ are 1 and $e^{-2\alpha}$, it follows that the matrix L_2 norms of $\|\exp(\alpha L_e)\| \leqslant 1$ and $\|\exp(\alpha L_o)\| \leqslant 1$. To prove the unconditional stability of the algorithm we need only show that $\|\mathscr{P}\| \leqslant 1$. This follows immediately since $\|\mathscr{P}\| = \|\exp(\alpha L_e)\exp(\alpha L_o)\| \leqslant \|\exp(\alpha L_e)\|\ \|\exp(\alpha L_o)\|$, each of which is less than unity by the preceding argument.

To compute the wave number dependent amplification factor, we need to find the action of $\mathscr{P}$ on a *plane wave*, with wave number k,

$$u(k)_j = e^{ikj\,\Delta x}. \tag{17}$$

[1] For more accuracy one can verify that $\exp(\alpha L) = \exp((\alpha/2)\,L_e)\exp(\alpha L_o)\exp((\alpha/2)\,L_e) + \mathcal{O}(\alpha^3)$.

The square of the amplification factor is given by

$$|\xi(k)|^2 = \frac{|\mathscr{P}\mathbf{u}(k)|^2}{|\mathbf{u}(k)|^2}. \tag{18}$$

We find that

$$|\xi(k)|^2 = [\tfrac{1}{4}(1 + e^{-2\alpha})^2 \tag{19}$$
$$+ \tfrac{1}{2}(1 - e^{-4\alpha})\cos(k\,\Delta x) \tag{20}$$
$$+ \tfrac{1}{4}(1 - e^{-2\alpha})^2\cos(2k\,\Delta x)]^2 \tag{21}$$
$$+ [\tfrac{1}{4}(1 - e^{-2\alpha})^2\sin(2k\,\Delta x)]^2. \tag{22}$$

It can be checked that $|\xi| \leqslant 1$ for all k, α, so the algorithm is indeed unconditionally stable. This is compared with the usual implicit method [6], for which

$$\xi_{\text{implicit}} = \frac{1}{1 + 4\alpha\sin^2(k\,\Delta x/2)} \tag{23}$$

and the exact result, where

$$\xi_{\text{exact}} = \exp(-4\alpha\sin^2(k\,\Delta x/2)) \tag{24}$$

In Fig. 1 we have plotted the solid curves of $|\xi(k)|$ for the method described in this paper and compared it to an implicit solver whose $|\xi(k)|$ is shown in the dashed curves for $\alpha = \frac{1}{10}$, $\alpha = 1$, and $\alpha = 10$.

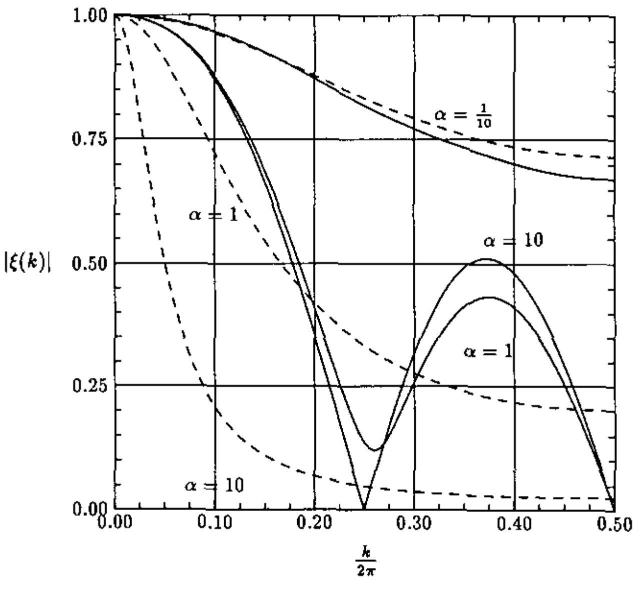

FIG. 1. Amplification factor.

WAVES

The linear wave equation in one dimension is

$$\frac{\partial u(x,t)}{\partial t} + c\frac{\partial u(x,t)}{\partial x} = 0. \qquad (25)$$

If we difference the spatial part to second order in the grid spacing Δx on a uniform periodic grid with an even number of sites, we obtain

$$\frac{du_j}{dt} + \frac{c}{2\Delta x}(u_{j+1} - u_{j-1}) = 0. \qquad (26)$$

In matrix form this is

$$\frac{d\mathbf{u}(t)}{dt} = \frac{c}{2\Delta x}\begin{pmatrix} 0 & -1 & 0 & \cdots & 0 & 1 \\ 1 & 0 & -1 & \cdots & 0 & 0 \\ 0 & 1 & 0 & -1 & \cdots & 0 \\ \cdots \end{pmatrix}\mathbf{u}(t). \qquad (27)$$

We have

$$L = \begin{pmatrix} 0 & -1 & 0 & \cdots & 0 & 1 \\ 1 & 0 & -1 & \cdots & 0 & 0 \\ 0 & 1 & 0 & -1 & \cdots & 0 \\ \cdots & & & & & . \end{pmatrix} \qquad (28)$$

which can again be split into $L = L_e + L_o$, where

$$L_e = \begin{pmatrix} 0 & -1 & 0 & \cdots & 0 & 0 \\ 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & -1 & \cdots & 0 \\ 0 & 0 & 1 & 0 & 0 & \cdots \\ \cdots \end{pmatrix} \qquad (29)$$

and

$$L_o = \begin{pmatrix} 0 & 0 & 0 & \cdots & 0 & 1 \\ 0 & 0 & -1 & \cdots & 0 & 0 \\ 0 & 1 & 0 & 0 & \cdots & 0 \\ \vdots \\ -1 & 0 & 0 & \cdots & 0 & 0 \end{pmatrix}. \qquad (30)$$

Define

$$\alpha = \frac{c}{2\Delta x}\Delta t \qquad (31)$$

Then, the solution is

$$\mathbf{u}(t+\Delta t) = \exp(\alpha L)\,\mathbf{u}(t). \qquad (32)$$

Now, $\exp(\alpha L)$ is not available in closed form, but we can again approximate

$$\exp(\alpha L) = \exp(\alpha L_e)\exp(\alpha L_o) + \mathcal{O}(\alpha^2). \qquad (33)$$

The approximate time stepping operator

$$\mathcal{P} \equiv \exp(\alpha L_e)\exp(\alpha L_o) \qquad (34)$$

again forms the basis for our unconditionally stable integration algorithm for the wave equation.[2]

Since L is tridiagonal, $\exp(\alpha L)$ is dense. However, because both L_e and L_o are block 2×2 diagonal, they can be easily exponentiated to give $\mathcal{P}$. Let

$$A = \alpha\begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}. \qquad (35)$$

We again sum the exponential series to find that

$$\exp(A) = \begin{pmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{pmatrix}. \qquad (36)$$

Since both L_e and L_o are block diagonal, exponentiating them amounts to exponentiating A. Of course, both $\exp(\alpha L_e)$ and $\exp(\alpha L_o)$ are also block diagonal and orthogonal. Since the eigenvalues of $\exp(A)$ are $e^{i\alpha}$ and $e^{-i\alpha}$ which lie on the unit circle, it follows that $\|\exp(\alpha L_e)\| = 1$ and $\|\exp(\alpha L_o)\| = 1$. Since $\mathcal{P} = \exp(\alpha L_e)\exp(\alpha L_o)$ is the product of two orthogonal matrices, $\mathcal{P}$ is also orthogonal and $\|\mathcal{P}\| = 1$ and our algorithm is unconditionally stable. In this case the square of the amplification factor is unity as it should be.

DIFFUSIVE WAVES

In order to solve problems which have both diffusive and advective parts, we must combine the results of the previous two sections. Consider the equation

$$\frac{\partial u(x,t)}{\partial t} + c\frac{\partial u(x,t)}{\partial x} = v\frac{\partial^2 u(x,t)}{\partial x^2} \qquad (37)$$

which has both advective and diffusive terms. We solve this exactly by

$$\mathbf{u}(t+\Delta t) = e^{((c\Delta t/2\Delta x)L^{\text{wave}} + (v\Delta t/\Delta x^2)L^{\text{diffusion}})}\mathbf{u}(t) \qquad (38)$$

[2] For more accuracy one can again verify that $\exp(\alpha L) = \exp((\alpha/2)L_e)\exp(\alpha L_o)\exp((\alpha/2)L_e) + \mathcal{O}(\alpha^3)$.

and approximate it by

$$\begin{aligned}\mathbf{u}(t+\Delta t) &= e^{(c\Delta t/2\Delta x)L_e^{\text{wave}}}\, e^{(c\Delta t/2\Delta x)L_o^{\text{wave}}} \\ &\quad \times e^{(\nu\Delta t/\Delta x^2)L_e^{\text{diffusion}}}\, e^{(\nu\Delta t/\Delta x^2)L_o^{\text{diffusion}}} \\ &\quad \times \mathbf{u}(t) + \mathcal{O}([\Delta t]^2) \\ &= \mathcal{P}^{\text{wave}}\mathcal{P}^{\text{diffusion}}\mathbf{u}(t) + \mathcal{O}([\Delta t]^2). \qquad (39)\end{aligned}$$

The stability proof for this aggregate stepping operator follows from the fact that $\|\mathcal{P}^{\text{wave}}\mathcal{P}^{\text{diffusion}}\| \leqslant \|\mathcal{P}^{\text{wave}}\| \, \|\mathcal{P}^{\text{diffusion}}\| \leqslant 1$ since $\|\mathcal{P}^{\text{wave}}\| = 1$ and $\|\mathcal{P}^{\text{diffusion}}\| \leqslant 1$ by the results of the previous sections.

NONLINEAR TERMS

To solve nonlinear time dependent problems we locally linearize at a given time and use the algorithms described above. Consider first a nonlinear diffusion problem like

$$\frac{\partial \rho(x,t)}{\partial t} = \frac{\partial}{\partial x} D(\rho(x,t),x,t)\frac{\partial \rho(x,t)}{\partial x}. \qquad (40)$$

A second-order differencing of the spatial derivatives leads to

$$\begin{aligned}2\Delta x^2 \frac{d\rho_j}{dt} &= (D_j + D_{j+1})\,\rho_{j+1} \\ &\quad + (D_j + D_{j-1})\,\rho_{j-1} \\ &\quad - (2D_j + D_{j-1} + D_{j+1})\,\rho_j \qquad (41)\end{aligned}$$

which may again be split into even and odd parts which leads to our familiar two by two's which take the form

$$A = \frac{D_j + D_{j+1}}{2\Delta x^2}\begin{pmatrix} -1 & 1 \\ 1 & -1 \end{pmatrix}. \qquad (42)$$

Since D is positive these matrices have one negative and one zero eigenvalue, which means that when exponentiated all eigenvalues are $\leqslant 1$, yielding stability as before.

Consider next a nonlinear wave problem like

$$\begin{aligned}\frac{\partial \rho(x,t)}{\partial t} &= F(\rho(x,t),x,t) \\ &\quad \times \frac{\partial}{\partial x} G(\rho(x,t),x,t)\,\rho(x,t). \qquad (43)\end{aligned}$$

A second-order differencing of the spatial derivatives leads to

$$2\Delta x \frac{d\rho_j}{dt} = F_j[G_{j+1}\rho_{j+1} - G_{j-1}\rho_{j-1}] \qquad (44)$$

which may again be split into even and odd parts which leads again to two by two's which take the form

$$A = \frac{1}{2\Delta x}\begin{pmatrix} 0 & F_j G_{j+1} \\ -F_{j+1}G_j & 0 \end{pmatrix}. \qquad (45)$$

Again the exponential series may be summed to obtain

$$e^{\Delta t A} = \begin{pmatrix} \cos(\theta) & \dfrac{\Delta t\, F_j G_{j+1}\sin(\theta)}{2\Delta x\,\theta} \\ -\dfrac{\Delta t\, F_{j+1} G_j \sin(\theta)}{2\Delta x\,\theta} & \cos(\theta) \end{pmatrix}, \qquad (46)$$

where $\theta = (\Delta t/2\Delta x)\sqrt{F_j G_{j+1} F_{j+1} G_j}$. Note that A is skew symmetric only when $F_j = \pm G_j$ and in this case $e^{\Delta t A}$ will have eigenvalues on the unit circle so $\|e^{\Delta t A}\| = 1$, which leads to stability.

NUMERICAL RESULTS

As a test case of our algorithm we integrate Equation 2 for $x \in [0, 1]$ with the boundary conditions that $u(1,t) = 0$ and $u(0,t) = A\sin(\omega t)$. We have chosen $A = 100$, $c = 300$, $\lambda = 0.2$ with $\omega = 2\pi/\lambda c$ which is a highly nonlinear regime. Here $\nu = 0.1$ and $\Delta x = \frac{1}{511}$. $\Delta t = \text{cfl}\,\Delta x/(c + |A|) = 48 \times 10^{-9}$, where cfl $= 0.01$. These results are displayed in Fig. 2, where we have compared our method to a fourth-order Runge–Kutta scheme. We see that after 40,000 time steps the solid curve and the dashed curve are almost identical. This figure clearly shows nonlinear wave steepening and viscous dissipation.

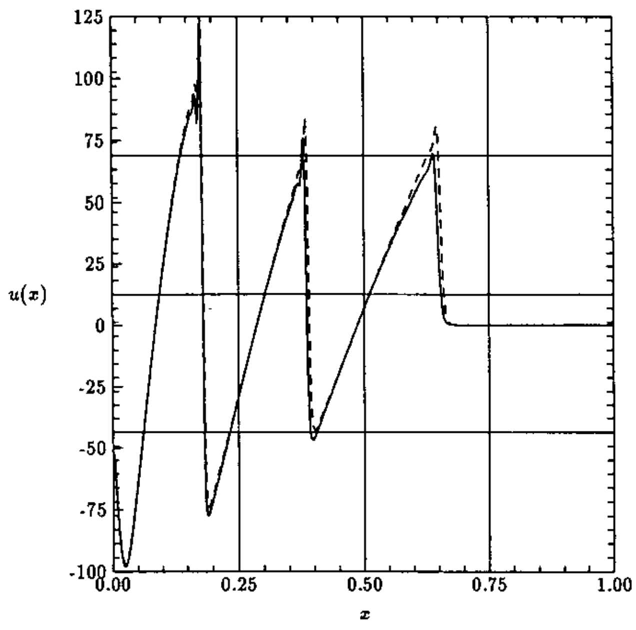

FIG. 2. Solution to Burger's equation after 40,000 time steps.

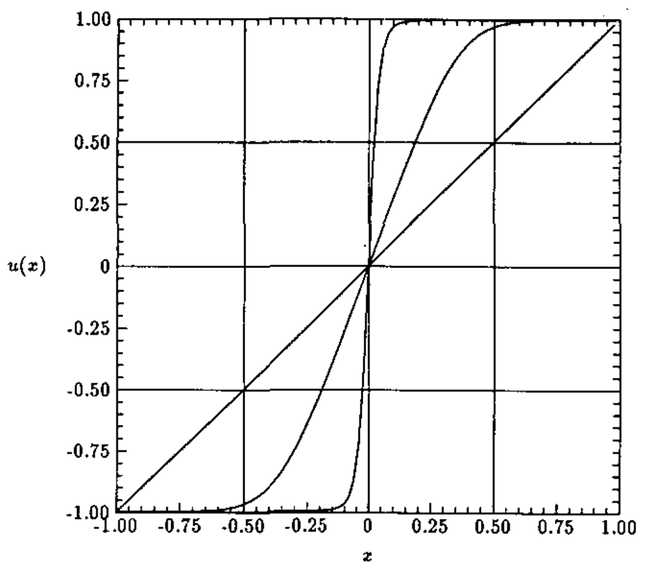

FIG. 3. Solution to Burger's equation after 0, 500, and 1000 time steps.

As a final case to test our algorithm on nonlinear steady-state behavior we again integrate Eq. (2) for $x \in [-1, 1]$ with the boundary conditions that $u(-1, t) = -1$ and $u(1, t) = 1$ and $u(x, 0) = x$, with $\nu = \frac{1}{100}$. We have chosen $\Delta t/\Delta x = \frac{1}{8}$ with $\Delta x = \frac{1}{128}$. The steady state behavior can be compared to the exact solution $u(x, t \to \infty) = \tanh(x/2\nu)$. This evolution of the steady-state shock wave is displayed in Fig. 3 at various time steps. We have also investigated reducing Δx holding Δt fixed and have done the same simulation with $\Delta x = \frac{1}{256}$ and $\frac{1}{512}$ and found results that are almost identical to those shown in Fig. 3 with the added improvement that for long times there is increased steepening due to the finer grid.

HIGHER DIMENSIONS

In order to implement these algorithms in higher dimensions on structured grids we must break the spatial operator L into an even part and an odd part for each spatial direction. If we consider the three-dimensional problem defined on an even three-torus, this amounts to approximating

$$e^{\alpha L} = e^{\alpha L_e^x} e^{\alpha L_o^x} e^{\alpha L_e^y} e^{\alpha L_o^y} e^{\alpha L_e^z} e^{\alpha L_o^z} + \mathcal{O}(\alpha^2) \tag{47}$$

which is straightforward to implement.

UNSTRUCTURED MESHES

To extend our technique to unstructured meshes we must find a splitting of L into m terms. For our one-dimensional examples L was split into two terms L_e and L_o. In the previous section it was shown how to split L into six pieces for a structured three-dimensional problem. If we can find such a splitting $L = \sum_{a=1}^{m} L_a$, we approximate[3]

$$e^{\alpha L} = \prod_{a=1}^{m} e^{\alpha L_a} + \mathcal{O}(\alpha^2). \tag{48}$$

Solving this splitting problem is equivalent to finding an m-coloring of the line graph of L. It can be shown [9] that the line graph of L can be colored in less than or equal to one plus the largest degree of any vertex in the graph of L. This means, for example, that if we use a finite-volume scheme based on tetrahedrons then L can be written as a sum of at most five terms since each tetrahedron has four faces.

CONCLUSION

We have presented new unconditionally stable and explicit algorithms for nonlinear fluid dynamics problems and explored numerically some simple one-dimensional problems to demonstrate the use of our methods. In addition, we have laid a groundwork for handling large unstructured problems in higher dimensions.

Algorithm design needs to be reevaluated with parallel processing in mind. The methods presented in this paper is one way to do this and could be the starting point for a new generation of method.

REFERENCES

1. J. L. Richardson, *Comput. Phys. Comun.* **63**, 84 (1991).
2. H. DeRaedt, *Comput. Phys. Rep.* 7, 1 (1987).
3. J. M. Burgers, *Adv. Appl. Mech.* 1, 171 (1948).
4. L. N. Long, M. M. S. Khan, and H. T. Sharp, to appear in *AIAA J.* **29**, No. 4 (1991).
5. C. Hirsch, *Numerical Computation of Internal and External Flows*, Vols. 1, 2 (Wiley, New York, 1988).
6. W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes* (Cambridge Univ. Press, London, 1986).
7. G. I. Marchuk, *Methods of Computational Mathematics* (Springer-Verlag, Berlin/New York, 1981).
8. D. H. Anderson, J. C. Tannehill, and R. H. Pletcher, *Computational Fluid Mechanics and Heat Transfer* (McGraw-Hill, New York, 1984).
9. F. Harary, *Graph Theory* (Addison-Wesley, Reading, MA, 1971), p. 133.

[3] For an accuracy $\mathcal{O}(\alpha^3)$ we use the formula $e^{\alpha L} = \prod_{a=1}^{m} e^{(1/2)\alpha L_a} \prod_{a=m}^{1} e^{(1/2)\alpha L_a} + \mathcal{O}(\alpha^3)$.