JOURNAL OF COMPUTATIONAL PHYSICS 35, 319–340 (1980)

On the Practical Use of High-Order Methods for Hyperbolic Systems*

ELI TURKEL

Courant Institute of Mathematical Sciences,
New York University, New York 10012

Received December 28, 1978; revised May 16, 1979

A number of high-order methods are tested on a variety of dynamic problems in one, two, and three space dimensions. These problems include wave propagation phenomena as well as an asymptotic approach to a steady state. Both smooth and shocked flows are considered. The methods compared require only minor modifications of many existing second-order schemes. The results show that significant gains can be expected from the use of fourth-order methods. Spectral methods are also considered for some of the problems presented.

I. INTRODUCTION

Within the last few years a number of high-order methods have been proposed for for the numerical solution of hyperbolic equations. It will be demonstrated that the use of schemes with spatial accuracy of at least fourth order offers advantages both in computer speed and computer storage over standard first- and second-order methods.

In this study we shall concentrate on schemes that are minor modifications of standard second-order methods. This requires that the higher-order methods be only second order in time. Hence, these methods are most suitable to stiff equations or to systems where the steady-state solution is of most significance. The results indicate that for many situations of physical significance there is less necessity for increased accuracy in time. The achievement of higher-order methods in time requires more work per time step and more involved algorithms. These methods are most appropriate for problems with rapid temporal oscillations. The methods developed to date for these problems of high-order temporal accuracy are mainly based on Taylor series expansions or Runge–Kutta methods; see Burstein and Mirin [5], Rusanov [32], Abarbanel, Gottlieb, and Turkel [1], Turkel, Abarbanel, and Gottlieb [39], and Steppeler [34] for further details.

* This report was prepared as a result of work performed under NASA Contract No. NAS1-14101 at ICASE, NASA Langley Research Center, Hampton, Va. 23665. Also partial support was given by DOE Contract EY-76-C-02-3077 at New York University.

0021-9991/80/060319–22$02.00/0

II. High Order Methods

We will discuss the various schemes for one-dimensional problems. For leapfrog methods the extension to several dimensions is straightforward. With the other methods to be described the simplest extension to multidimensions is by either a splitting method or an alternating direction scheme.

We consider the equation

$$u_t + f_x = 0 \tag{2.1a}$$

or in quasi-linear form

$$u_t + Au_x = 0. \tag{2.1b}$$

The simplest second-order method is given by leapfrog

$$u_j^{n+1} = u_j^{n-1} - \frac{\Delta t}{\Delta x}(f_{j+1}^n - f_{j-1}^n), \qquad j = 1,..., N-1. \tag{2.2}$$

This method is second order in space and time and is stable if $A\,\Delta t/\Delta x \leqslant 1$. At $j = 0$ we use a one-sided Euler method

$$u_0^{n+1} = u_0^n - \frac{\Delta t}{\Delta x}(f_1^n - f_0^n). \tag{2.3}$$

This has been shown to be stable for a scalar equation by Gustafsson *et al.* [16]. The extension to systems will be described later. A similar algorithm is used at $j = N$. These boundary algorithms are used to supplement the given boundary conditions.

A fourth-order version of this method suggested by Kreiss and Oliger [23] is

$$u_j^{n+1} = u_j^{n-1} - \frac{\Delta t}{6\Delta x}[8(f_{j+1}^n - f_{j-1}^n) - (f_{j+2}^n - f_{j-2}^n)]. \tag{2.4}$$

This method is fourth-order accurate in space and second order in time and is stable if $A\,\Delta t/\Delta x \leqslant 0.72$, the introduction of a larger domain of dependence introduces complications near the boundary. Oliger [27] suggests

$$\begin{aligned} u_0^{n+1} &= u_0^{n-1} - \frac{\Delta t}{3\Delta x}\left[-\frac{11}{2}(f_0^{n+1} + f_0^{n-1}) + 18f_1^n - 9f_2^n + 2f_3^n\right], \\ u_1^{n+1} &= u_1^{n-1} - \frac{\Delta t}{3\Delta x}\left[-2f_0^n - \frac{3}{2}(f_1^{n+1} + f_1^{n-1}) + 6f_2^n - f_3^n\right] \end{aligned} \tag{2.5}$$

with a similar set at the right boundary. Due to the implicit nature of these equations they are less practical for nonlinear equations. One possibility is to linearize these equations (see Gary [10]). Another alternative offered by Oliger [28] is to use a finer

mesh near the boundary with a dissipative method used in the finer mesh. Due to these complications this method was not used in the comparison runs. For equations on the globe, where boundaries do not appear, this method may be useful. Kreiss and Oliger [24] as well as Williamson and Browning [40] and Kalnay–Rivas *et al.* [19] present comparisons for this case.

The leapfrog methods are all nondissipative. In many problems involving large gradients one wishes to use a scheme that is inherently dissipative. A second-order method with this property is the Lax–Wendroff scheme. A two-step implementation of this suggested by MacCormack [25] is

$$\begin{aligned} u_j^{(1)} &= u_j^n - \frac{\Delta t}{\Delta x}(f_{j+1} - f_j), && j = 0,\ldots, N-1 \\ u_j^{n+1} &= \frac{1}{2}\left[u_j^n + u_j^{(1)} - \frac{\Delta t}{\Delta x}(f_j^{(1)} - f_{j-1}^{(1)})\right], && j = 1,\ldots, N. \end{aligned} \tag{2.6}$$

Another variant uses backward differences on the first step and forward differences in the corrector. Both variants are second order in space and time and are stable when $(\Delta t/\Delta x)A \leqslant 1$. At the boundaries we use

$$u_N^{(1)} = u_N^n - \frac{\Delta t}{\Delta x}(f_N^n - f_{N-1}^n) \tag{2.7a}$$

and

$$u_0^{n+1} = \frac{1}{2}\left[u_0^n + u_0^{(1)} - \frac{\Delta t}{\Delta x}(f_1^{(1)} - f_0^{(1)})\right]. \tag{2.7b}$$

A method which is identical to (2.7) is to extrapolate beyond the boundaries for the fluxes, that is, let

$$f_{N+1}^n = 2f_N^n - f_{N-1}^n, \tag{2.7c}$$

$$f_{-1}^{(1)} = 2f_0^{(1)} - f_1^{(1)} \tag{2.7d}$$

and then use (2.6). On serial machines the choice between (2.7a) and (2.7b) and (2.7c) and (2.7d) is determined by programming convenience. The problem on acoustics discussed later was run on the CDC–STAR. In this case the boundary equations are performed over the entire grid in order to use vector operations. The use of (2.7c) and (2.7d) is then considerably more efficient.

A fourth extension of (2.6) is given by Gottlieb and Turkel [12]

$$\begin{aligned} u_j^{(1)} &= u_j^n + \frac{\Delta t}{6\Delta x}(7f_j^n - 8f_{j+1}^n + f_{j+2}^n), && j = 0,\ldots, N-2 \\ u_j^{n+1} &= \frac{1}{2}\left[u_j^n + u_j^{(1)} - \frac{\Delta t}{6\Delta x}(7f_j^{(1)} - 8f_{j-1}^{(1)} + f_{j-2}^{(1)})\right], && j = 2,\ldots, N. \end{aligned} \tag{2.8a}$$

As before there exists another variant

$$u_j^{(1)} = u_j^n - \frac{\Delta t}{6\Delta x}(7f_j^n - 8f_{j-1}^n + f_{j-2}^n), \qquad j = 2,\ldots, N$$
$$u_j^{n+1} = \frac{1}{2}\left[u_j^n + u_j^{(1)} + \frac{\Delta t}{6\Delta x}(7f_j^{(1)} - 8f_{j+1}^{(1)} + f_{j+2}^{(1)})\right], \qquad j = 0,\ldots, N-2. \tag{2.8b}$$

In order to have fourth-order accuracy for nonlinear problems it is necessary to use (2.8a) and (2.8b) at alternate time steps. The extension to several dimensions will be discussed in Section 3.

The predictor in (2.8a) cannot be used at $j = N-1, N$. At these points we substitute

$$u_{N-1}^{(1)} = u_{N-1}^n - \frac{\Delta t}{6\Delta x}(4f_N^n - f_{N-1}^n - 4f_{N-2}^n + f_{N-3}^n),$$
$$u_N^{(1)} = u_N^n - \frac{\Delta t}{6\Delta x}(15f_N^n - 28f_{N-1}^n + 17f_{N-2}^n - 4f_{N-3}^n). \tag{2.9a}$$

Similar for the corrector we substitute

$$u_0^{n+1} = \frac{1}{2}\left[u_0^n + u_0^{(1)} - \frac{\Delta t}{6\Delta x}(4f_3^{(1)} - 17f_2^{(1)} + 28f_1^{(1)} - 15f_0^{(1)})\right],$$
$$u_1^{n+1} = \frac{1}{2}\left[u_1^n + u_1^{(1)} - \frac{\Delta t}{6\Delta x}(-f_3^{(1)} + 4f_2^{(1)} + f_1^{(1)} - 4f_0^{(1)})\right]. \tag{2.9b}$$

Similar formulas can be used with the variant (2.8b). As before these formulas are equivalent to extrapolating the fluxes. This method is stable if $(\Delta t/\Delta x)(\partial f/\partial u) \leqslant \frac{2}{3}$.

For many stiff problems an implicit method has been tried. In this case the accuracy determines the time step as a function of the dominant speeds of propagation. Unconditional stability guarantees that the less important fast waves will not give rise to instabilities though they are not to be accurately computed. An important point is to make sure that these fast waves do not grow in amplitude and destroy the accuracy of the scheme. A standard implicit method for solving (2.1) is the Crank–Nicolson method

$$u_j^{n+1} + \frac{\Delta t}{4\Delta x}(f_{j+1}^{n+1} - f_{j-1}^{n+1}) = u_j^n - \frac{\Delta t}{4\Delta x}(f_{j+1}^n - f_{j-1}^n). \tag{2.10}$$

For linear equations this involves the inversion of a block tridiagonal matrix. For nonlinear problems this can be solved either by iteration or else by a linearization expansion (see Lindemuth and Killeen [24], Beam and Warming [3], and Briley and McDonald [4]).

A fourth-order version of (2.10) was suggested by Collatz [7] and later by Jones

et al. [18] and by Kreiss (see [30]). It can also be derived from linear finite elements (see [31, 37]). To solve (2.1) we use

$$\frac{1}{6}u_{j+1}^{n+1}+\frac{2}{3}u_j^{n+1}+\frac{1}{6}u_{j-1}^{n+1}+\frac{\Delta t}{4\Delta x}(f_{j+1}^{n+1}-f_{j-1}^{n+1})$$
$$=\frac{1}{6}u_{j+1}^{n}+\frac{2}{3}u_j^{n}+\frac{1}{6}u_{j-1}^{n}-\frac{\Delta t}{4\Delta x}(f_{j+1}^{n}-f_{j-1}^{n}). \tag{2.11}$$

This system again is solved by a block tridiagonal inversion and so does not take much more time than (2.10). For nonlinear problems both iteration and linearization techniques can be used as before (see [3, 41]). Use of Galerkin procedures indicates that the boundaries should be treated by

$$\frac{1}{6}u_0^{n+1}+\frac{1}{3}u_1^{n+1}+\frac{\Delta t}{4\Delta x}(f_1^{n+1}-f_0^{n+1})=\frac{1}{6}u_0^{n}+\frac{1}{3}u_1^{n}-\frac{\Delta t}{4\Delta x}(f_1^{n}-f_0^{n})$$

and (2.12)

$$\frac{1}{6}u_{N-1}^{n+1}+\frac{1}{3}u_N^{n+1}+\frac{\Delta t}{4\Delta x}(f_N^{n+1}-f_{N-1}^{n+1})=\frac{1}{6}u_{N-1}^{n}+\frac{1}{3}u_N^{n}-\frac{\Delta t}{4\Delta x}(f_N^{n}-f_{N-1}^{n}).$$

However, this is only first order at the boundary. An alternative is to use the box scheme as suggested by Sköllermo [33],

$$u_0^{n+1}+u_1^{n+1}+\frac{\Delta t}{\Delta x}(f_1^{n+1}-f_0^{n+1})=u_0^{n}+u_1^{n}-\frac{\Delta t}{\Delta x}(f_1^{n}-f_0^{n})$$

and (2.13)

$$u_N^{n+1}+u_{N-1}^{n+1}+\frac{\Delta t}{\Delta x}(f_N^{n+1}-f_{N-1}^{n+1})=u_N^{n}+u_{N-1}^{n}-\frac{\Delta t}{\Delta x}(f_N^{n}-f_{N-1}^{n}).$$

This boundary treatment is now second order in space and time and preserves the tridiagonal nature of the matrix. According to [17] it would be preferable to use a third-order approximation at the boundary. However, this would destroy the tridiagonal structure of the matrix. The results shown later demonstrate that in many problems (2.13) is sufficient. This method can be extended to mixed hyperbolic–parabolic problems by use of operator compact implicit methods (see Ciment *et al.* [6]). For applications to boundary layer flow, see [21, 42].

The use of Padé approximations can also be used for high-order compact explicit schemes, as is discussed in the Appendix.

As an alternative to these fourth-order methods spectral methods are also considered. With the use of a modified Euler time integration these methods are second order in time and "infinite" order in space. When the space domain is periodic a Fourier collocation method is appropriate. As a second-order Runge–Kutta is unstable when used with the Fourier method, the modification of [13] can be used to solve (2.1) on $0 \leqslant x \leqslant 2\pi$.

Let

$$f_k = \sum_{j=-N}^{N} \hat{f}_j e^{\pi i jk/N}, \qquad j = 0,\ldots, 2N-1,$$

then define $z = i\alpha k\, \Delta t$ and choose $\alpha \geqslant 1.5\rho(A)$, $A = \partial f/\partial u$.

$$(\Delta t f_j)_x^{(1)} = \sum_{k=-N}^{N} \frac{1}{6\alpha}(-7 + 8e^z - e^{2z})\hat{f}_k e^{\pi i jk/N}, \tag{2.14a}$$

$$(\Delta t f_j)_x^{(2)} = \sum_{k=-N}^{N} \frac{1}{6\alpha}(7 - 8e^{-z} + e^{-2z})\hat{f}_k e^{\pi i jk/N}. \tag{2.14b}$$

We then integrate (2.1) by

$$\bar{u}_j = u_j^n - (\Delta t f_j)_x^{(1)}$$

and (2.15)

$$u_j^{n+1} = \tfrac{1}{2}[u_j^n + \bar{u}_j^n - (\Delta t \bar{f}_j)_x^{(2)}].$$

This method is now unconditionally stable and second order in time and "infinite" order in space.

For problems with boundaries, Fourier expansions are inappropriate as a Gibbs phenomena will occur at the boundaries. Instead one can expand in a Chebyshev series (see [11]).

Let

$$f_j = \sum_{k=0}^{N} a_k T_k(x), \tag{2.16}$$

then

$$(\Delta t f_j)_x = \sum_{k=0}^{N} b_k T_k(x) \tag{2.17}$$

with

$$b_k = \frac{2}{c_k} \sum_{\substack{l=k+1 \\ l+k \text{ odd}}}^{N} l a_l\, \Delta t. \tag{2.18}$$

$$c_0 = c_N = 2; \qquad c_j = 1 \qquad j = 1,\ldots, N-1.$$

Then we integrate (2.1) by

$$u_j^{n+1/2} = u_j^n - \left(\frac{\Delta t}{2} f_j^n\right)_x,$$

$$u_j^{n+1} = u_j^n - (\Delta t\, f_j^{n+1/2})_x. \tag{2.19}$$

This method is stable if $A\,\Delta t \leqslant 8/N^2$ (see [11]). To improve the stability criterion one can replace (2.18) by

$$z = e^{-2\alpha\Delta t l}, \qquad b_k = \frac{2}{c_k}\sum_l \frac{a_l}{6\alpha}\,[11 - 18z + 9z^2 - 2z^3].$$

Computationally this modification seems to be unconditionally stable (see [13]).

For systems of equations one has to modify the boundary treatment to take into account the characteristic variables (see [14]). This is discussed in more detail in the following sections.

III. One-Dimensional Problems

As the first problem, we consider wave propagation in a bounded domain. In particular we solve the system

$$\begin{aligned} u_t + \tfrac{1}{2}[(c_p + c_m)\,u_x + (c_p - c_m)\,v_x] &= 0, \\ v_y + \tfrac{1}{2}[(c_p - c_m)\,u_x + (c_p + c_m)\,v_x] &= 0, \end{aligned} \qquad 0 \leqslant x \leqslant 1. \tag{3.1}$$

A solution of (3.1) is

$$\begin{aligned} u(x, t) &= F(x + c_p t) + G(x - c_m t), \\ v(x, t) &= F(x + c_p t) - G(x - c_m t). \end{aligned} \tag{3.2}$$

To ensure that (3.2) is the unique solution to (3.1) we add initial conditions

$$\begin{aligned} u(x, 0) &= F(x) + G(x), \\ v(x, 0) &= F(x) - G(x), \end{aligned} \tag{3.3a}$$

and boundary conditions

$$\begin{aligned} u(0, t) &= F(-c_p t) + G(-c_p t), \\ u(1, t) &= F(1 - c_p t) + G(1 - c_p t). \end{aligned} \tag{3.3b}$$

Thus, both boundary conditions are imposed on u. This is a well-posed problem whenever c_p and c_m have opposite signs. We shall always assume $c_m < 0 < c_p$.

The boundary algorithms discussed in the previous section are necessarily stable only for a scalar equation. For systems we must account for the characteristic variables (see [14]). For (3.1) we have

$$\begin{aligned} (u + v)_t + c_p(u + v)_x &= 0, \\ (u - v)_t + c_m(u - v)_x &= 0, \end{aligned} \qquad 0 \leqslant x \leqslant 1. \tag{3.4}$$

where $c_m < 0 < c_p$. At $x = 0$ we wish to use the given algorithm on the quantity $u - v$ which is propagating toward the boundary. We denote by $u_0^c - v_0^c$ the value of $u - v$ calculated by one of the boundary algorithms of the previous section. We then calculate the boundary values by

$$\begin{aligned} u_0^n &= u(0, t) = F(-c_p t) + G(-c_p t), \\ v_0^n &= v_0^c + (u_0^n - u_0^c), \end{aligned} \tag{3.5a}$$

and similarly

$$\begin{aligned} u_N^n &= u(1, t) = F(1 - c_p t) + G(1 - c_p t), \\ v_N^n &= v_N^c + (u_N^c - u_N^n). \end{aligned} \tag{3.5b}$$

This correction was used for many of the results in this section. For some of the methods the scheme was unstable without the use of the correction, while for others the correction was not necessary. For these latter cases the corrections to v did not significantly change the accuracy of the method.

This problem was solved by all the methods discussed in Section 2. The effectiveness of the methods is calculated by comparing the mesh sizes and computer times required to achieve a given error. We define the normalized L_2 error as

$$\text{ERR} = \left[\frac{\sum [u_j^n - u(x, t)]^2 + [v_j^n - v(x, t)]^2}{\sum (u_j^0)^2 + (v_j^0)^2}\right]^{1/2}. \tag{3.6}$$

The error requirement was that ERR be about 5 % at $t = 5.0$. That is, the normalized L_2 error should be 0.05 at a time corresponding to the transition of five waves with a wavelength of one.

To simulate the range of problems that occur naturally we have chosen three sets of problems. The first is

$$\begin{aligned} c_p &= 30, & c_m &= -1, \\ F(x) &= 0.01 \sin 2\pi x, & G(x) &= \cos 2\pi x; \end{aligned} \tag{3.7}$$

the second is

$$c_p = 10, \qquad c_m = -1; \tag{3.8}$$

while the third is

$$\begin{aligned} c_p &= 1, & c_m &= -1, \\ F(x) &= \sin 2\pi x, & G(x) &= \cos 2\pi x. \end{aligned} \tag{3.9}$$

These problems correspond to the various degrees of stiffness found in many physical situations.

TABLE I

System (3.1)–(3.2) with $c_p = 30$, $c_m = -1$,
$F(x) = 0.01 \sin 2\pi x$, $G(x) = \cos 2\pi x$; Periodicity not Used

Method	Order	N	$\Delta t/30\Delta x$	L_2 error ($t = 5$)	*CPU* time
Two step (2.6)	2	75	0.90	0.0490	344.5
Two step (2.8)	4	20	0.65	0.0278	49.2
Implicit (2.10)	2	81	25.00	0.0513	10.9
Implicit (2.11)	4	44	25.00	0.0478	4.7
Implicit (2.11)	4	21	9.00	0.0495	2.8
Chebyshev (2.16)	∞	9	0.50	0.0026	5.6

In Table I we present the results for (3.7) where the fast speed is 30 times larger than the significant speed. The computer times are for the CDC 6600 at New York University. To achieve 5% accuracy the fourth-order extension of the MacCormack method ((2.8–(2.9)) requires about one-seventh of the time required by the second-order method and about one-quarter of the storage. As expected the Crank–Nicolson is most effective when using larger time steps. The fourth-order implicit is about four times faster and requires about one-quarter the storage. It is interesting to note that the fourth-order implicit is most effective with smaller time steps than that used with the Crank–Nicolson method. For large time steps the temporal errors dominates the spatial accuracy and the fourth-order accuracy in space is not being utilized. We wish to choose time steps so that the spatial and temporal errors are about equal. For this problem the Chebyshev method was more efficient than the fourth-order explicit method. Hence, the spectral expansions should be seriously considered for many applications even in bounded domains. The timings for the Chebyshev method depend

TABLE II

System (3.1)–(3.2) with $c_p = 10$, $c_m = -1$,
$F(x) = 0.01 \sin 2\pi x$ $G(x) = \cos 2\pi x$; Periodicity not Used

Method	Order	N	$\Delta t/10\Delta x$	L_2 error ($t = 5$)	*CPU* time
Two step (2.6)	2	66	0.90	0.0489	64.2
Two step (2.8)	4	15	0.65	0.0475	7.8
Implicit (2.10)	2	81	9.00	0.0468	9.4
Implicit (2.11)	4	41	9.00	0.0507	3.4
Implicit (2.11)	4	13	0.90	0.0383	3.0
Chebyshev (2.16)	∞	9	0.50	0.0022	1.9

heavily on the fast Fourier transform that is used. In Table II the results are presented for (3.8) where the fast speed is 10 times faster than the speed of interest. The results are similar to that in Table I. If the total number of points N in each direction is not large, then one can calculate the Fourier series directly without using fast transform methods. The point at which the FFT is significantly faster is machine dependent. All runs in this paper used the FFT for uniformity.

In Table III we present the results for the wave equation where all speeds are of interest. The fourth-order Lax–Wendroff method is still about twice as fast as the second-order version and requires less than one-third the storage. This gain is achieved by choosing a coarse mesh and a time step below that allowed by stability considerations. Choosing a small time step reduces the errors in time and only increases the work linearly. Since the spatial errors are kept small by the fourth-order method, we still gain in efficiency. Similarly the fourth-order implicit is more efficient than the second-order implicit when the fourth-order method is coupled with a small time step. Because of the necessity of reduced time steps the gains in efficiency are not as great as they were for the problems with different time scales.

TABLE III

System (3.1)–(3.2) with $c_p = 1$, $c_m = -1$; Periodicity not Used.

Method	Order	N	$\Delta t/\Delta x$	L_2 error ($t = 5$)	*CPU* time
		$F(x) = \sin 2\pi x$, $G(x) = \cos 2\pi x$			
Two step (2.6)	2	37	0.90	0.0497	2.58
Two step (2.8)	4	41	0.65	0.0489	5.39
Two step (2.8)	4	12	0.25	0.0525	1.26
Implicit (2.10)	2	67	0.90	0.0481	6.41
Implicit (2.11)	4	41	0.90	0.0536	3.38
Implicit (2.11)	4	9	0.25	0.0447	0.61
Chebyshev (2.16)	∞	9	0.20	0.0414	0.49
		$F(x) = \sin 8\pi x$, $G(x) = \cos 8\pi x$			
Two Step (2.6)	2	68	0.90	0.1955	5.33
Two Step (2.8)	4	31	0.30	0.1851	5.07
Two Step (2.6)	2	80	0.90	0.1479	7.13
Two Step (2.8)	4	33	0.30	0.1384	5.68
Two Step (2.6)	2	99	0.90	0.0998	10.69
Two Step (2.8)	4	37	0.30	0.0953	7.07
Two Step (2.6)	2	144	0.90	0.0491	21.582
Two Step (2.8)	4	43	0.30	0.0483	10.860

In Table III we also consider the wave equation but now with four wavelengths within the domain of integration. We compare the efficiency of the second-order and fourth-order two-step methods for a variety of error levels. Choosing the time step for the fourth-order method as $\Delta t = 0.3\Delta x$, then even at the 20% error level the fourth-order method requires less computer time and reduces the storage requirements in half compared with the second-order method. If one does not fine tune the time step for the fourth-order method, then it is less efficient, computer-time-wise, at the 20% level than (2.6). However, for Courant numbers between 0.15 and 0.40, the fourth-order method is already more efficient at the 15% error level. At the 5% error level and a Courant number of 0.30 the error is dominated by the time discretization. Hence, for this error level a Courant number of 0.25 was chosen, for the fourth-order method (2.8), to reduce the temporal error.

Thus, we have shown that for the wave equation fourth-order methods are competitive even at error levels in the range of 15 to 20%. Though the reduction in computer running time is not significant at this error level, the storage requirements are halved. For stiff problems where higher-order methods are more appropriate the fourth-order methods are more efficient even at the 20% error level.

The purpose of the tests is to demonstrate the usefulness of higher-order modifications to existing codes. This table also indicates the advantage of implicit methods. However, the computer time required for an implicit method increases cubically with the number of equations to be solved. Hence, for realistic problems the comparison of explicit and implicit methods has to be done on an indivudial basis. Similar remarks apply to the spectral method.

For the second one-dimensional problem we consider flow in a Laval nozzle. In contrast to the previous case the equations are nonlinear and the question of importance is the steady-state solution. Let $A(x)$ be the area of the nozzle at position x. Then the equations of motion are

$$\begin{aligned} (\rho A)_t + (\rho A u)_x &= 0, \\ (A\rho u)_t + [A(\rho u^2 + p)]_x &= A_x p, \\ (AE)_t + [Au(E + p)]_x &= 0, \end{aligned} \tag{3.10}$$

where $p = (\gamma - 1)(E - \frac{1}{2}u^2)$ and $\gamma = 1.4$. At the inlet the flow is subsonic and we specify E and ρ as the known steady solution. The outlet condition depends on the solution desired. For a smooth solution the flow is supersonic and no boundary conditions are specified. For a shocked solution the pressure is specified at the outlet (see [8]). If $A'(x)$ is given by a second difference, then the accuracy of the higher-order methods deteriorate and hence $A'(x)$ was calculated analytically. In other cases where $A(x)$ is given by data points it is recommended that $A'(x)$ be calculated by differentiating a cubic spline interpolation. Similar situations arise in flows over airfoils when the radius of curvature is required. For the examples given below $A(x)$ was chosen as a hyperbolic cosine.

For smooth profiles it was found that the major difficulty occurred at the throat where the flow is sonic. When an artificial viscosity is not added, many of the methods

TABLE IV

Smooth Solution to System (3.10)

Method	Order	N	CFL	Viscosity factor	L_2 Error steady state	CPU time
MacCormack	2	41	0.90	0	1.93×10^{-3}	4.70
Richtmyer	2	41	0.90	0	1.27×10^{-3}	4.28
Brailaskaya	2	41	0.90	0.35	1.12×10^{-3}	4.55
Two step (2.8)	4	21	0.60	0	8.15×10^{-4}	2.02
Brailaskaya	4	21	0.60	0.50	1.93×10^{-3}	2.32
Fully implicit	2	35	0.90	0.30	1.10×10^{-3}	9.75
Fully implicit	2	35	3.0	0.30	1.10×10^{-3}	3.01
Crank–Nicolson	2	41	0.9	0.40	1.00×10^{-3}	13.09
Crank–Nicolson	2	41	2.0	0.40	1.02×10^{-3}	6.52
Fully implicit	4	33	0.90	0.30	9.90×10^{-4}	7.73
Fully implicit	4	29	3.00	0.30	1.40×10^{-3}	2.20
Crank–Nicolson	4	31	0.90	0.70	1.22×10^{-3}	7.40
MacCormack	2	171	0.90	0.	1.01×10^{-4}	111.51
Two step	4	39	0.60	0.	1.09×10^{-4}	6.79

gave rise to an expansion shock. The solution is considered converged if p changes by less than $\epsilon = 10^{-5}$ in one time step. For finer grids a smaller ϵ was used.

In Table IV we consider the results for a smooth profile. The second-order methods that we considered are the MacCormack method, the Brailaskaya–Matsuno scheme, and implicit methods with both centered and backward time differencing. The two-step Richtmyer method gave results very close to those of the two-step MacCormack method. The second- and fourth-order MacCormack methods required no artificial viscosities. The Brailaskaya and the implicit schemes required the addition of an artificial viscosity to avoid difficulties at the throat. For the implicit methods the artificial viscosity was added explicitly at time t. For this problem the fourth-order explicit methods are more than twice as efficient as the corresponding second-order method. They also required less than half the storage to achieve three significant digits in steady state. For four-digit accuracy the efficiency factor of the fourth-order method increases to 16. These results are all based on an odd number of mesh points so that the throat is located at a mesh point. Using an even number of mesh points the error again decayed monotonically but was considerably worse than the solution based on an odd number of points. This probably depends on whether or not mesh point is located at the throat.

For implicit methods the accuracy and stability depends strongly on the boundary treatment. In [14] it is proved that one does not need to include the given boundary conditions in the implicit method. Instead, after each time step one can correct the

boundary conditions. For (3.10) it was found that this worked only for time step less than three times the Courant limit. It is interesting to note that for larger time steps the nonlinear instabilities manifested themselves as negative pressures near the sonic point. It would have been difficult to trace these negative pressures to an improper boundary treatment. When the given boundary conditions were incorporated in the matrix to be inverted and a box scheme was used for the correct characteristic variables, then time steps greater than 15 times the Courant limit could be used.

For the implicit methods the higher-order methods are only about 30% more efficient. We speculate that in going to a steady state the use of the lower-order box scheme at the boundary causes a deterioration in the accuracy. An alternative for steady-state problems is to solve for $u^{n+1} - u^n$. Then only the explicit right-hand side of the equation need be of higher order since the implicit part converges to zero at the steady state. Hence, one can use a backward implicit method for the implicit part, while the derivatives on the right-hand side are approximated by fourth-order differences or else by a Chebyshev expansion.

We next consider (3.10) with a shocked solution profile. The solution for the fourth-order method is not a monotone function of the mesh at least through $N = 201$. The second-order method becomes monotone for $N > 85$. Hence, it is not meaningful to determine which method is most efficient to achieve a given error tolerance. Instead we find the asymptotic order of accuracy. For a particular linear problem Majda *et al.* [26] have shown that for discontinuous solutions one cannot achieve better than second-order accuracy. Even that is possible only if the discontinuity is treated in a special way.

For the nonlinear problem with the shock the equations were solved with a series of meshes. For the second-order method this ranged between 51 and 151, while for the fourth-order method, it ranged between 101 and 201. In both cases only an odd number of points were used. The initial conditions were chosen as smooth functions. We then fit a straight line, using a least-squares estimate, for the error as a function of the mesh width on a log–log scale. We calculate both L_2 and L_1 errors over the following domains. D_1 is the entire region; the error in D_1 is dominated by the shock. D_2 contains the entire region excluding a fixed distance about the shock. It was found that if a fixed number of points were excluded rather than a physical distance, then no meaningful results were found. D_3 is the domain downstream of the shock. In this region two characteristic curves out of three enter the domain from the shock as t increases. Hence, we examine if errors propagate from the shock into D_3. The results are presented in Table V.

Near the shock the L_2 error behaves as $(\Delta x)^{1/2}$ independent of the scheme while the L_1 error is linear in Δx. Downstream of the shock, in D_3, both the L_2 and L_1 errors behave according to the formal accuracy of the scheme. Hence, errors do not propagate along the characteristics. When we consider D_2, the entire region excluding the region the L_1 error still behaves according to the formal accuracy. The L_2 error of the second-order scheme is quadratic in Δx. For the fourth-order method the order of convergence seems to deteriorate to third-order upstream. Thus, errors propagate upstream against the characteristic directions. Additional investigations indicate

TABLE V

Shocked Solution to System (3.10)[a]

Method	$L_2(D_1)$	$L_1(D_1)$	$L_2(D_2)$	$L_1(D_2)$	$L_2(D_3)$	$L_1(D_3)$
Second order	0.48	0.97	2.15	2.10	2.59	2.60
Fourth order	0.52	1.01	2.88	3.95	5.68	5.40

[a] All rates of convergence are least square estimates based on an odd number of mesh points. The definitions of the domains D_k are given in the text.

that the oscillations in the vicinity of the shock are much larger upstream than downstream. However, it is mainly errors at the sonic point that reduce the rate of convergence. If we use only the calculations using points between 151 and 201, then the estimate of the rate of convergence increases to 3.7.

In [12] are presented graphs for the Riemann problem that show that the overshoots and oscillations about a shock are not aggravated by fourth-order methods. Similar results have been obtained with a Chebyshev collocation method. This is even more true when small time steps are used and the amplitude of the oscillations grows with a second-order two-step method.

IV. Multidimensional Problems

In this section we consider the explicit methods for two-dimensional problems. The methods described in Section II are extended to multidimensions by use of the splitting methods [35, 36]. Writing the general equation as

$$w_t + f_x + g_y + h_z = F, \tag{4.1}$$

we denote the solution to

$$w_t + f_x = F_1 \tag{4.2}$$

by $u^{n+1} = L_x u^n$. Here $F = F_1 + F_2 + F_3$ with the division of F into three parts arbitrary from the mathematical side. Usually the physics of the situation will dictate the splitting of the inhomogeneous terms. We then solve (4.1) by

$$\begin{aligned} w^{n+1} &= L_x L_y L_z w^n, \\ w^{n+2} &= L_z L_y L_x w^{n+1}. \end{aligned} \tag{4.3}$$

This splitting preserves the second-order accuracy in time and does not disturb the spatial accuracy of the schemes. The stability condition is simply the intersection of the one-dimensional stability conditions.

We first consider two systems with constant coefficients and one periodic direction. A more difficult problem is then presented. We first consider the linear equations of elasticity for $0 \leqslant x \leqslant 1$, $0 \leqslant y \leqslant 1$, $0 < t < 1$. In the x direction the solution is periodic, while at $y = 0$ and $y = 1$ the shear and normal stresses are zero. In the interior we have

$$\begin{aligned} \rho u_t &= \tau_{11,x} + \tau_{12,y}\,, \\ \rho v_t &= \tau_{12,x} + \tau_{22,y}\,, \\ \tau_{11,t} &= (2\mu + \lambda)\, u_x + \lambda v_y\,, \\ \tau_{12,t} &= \mu(u_y + v_x), \\ \tau_{22,t} &= \lambda u_x + (2\mu + \lambda)\, v_y \end{aligned} \tag{4.4}$$

with ρ, λ, and μ as constant. The solution considered is presented in detail in [38]. This problem was solved with the second-order MacCormack method (2.6). With a mesh of $\Delta x = \Delta y = 1/32$ the phase error of the computational solution is 0.023. The difference between the analytic and numerical solutions in the L_2 norm is 0.0964. Using the fourth-order two-step method (2.8) with a mesh of $\Delta x = \Delta y = 1/16$ the phase error is 0.0079 while the L_2 error is 0.106. Thus, with half the mesh points in each direction the fourth-order method gives similar L_2 errors and considerably smaller phase errors. With smaller error tolerances the efficiency of the fourth-order code improves dramatically.

The next case we consider are the magnetohydrodynamic equations linearized about an equilibrium solution $u^0 = v^0 = w^0 = B_y{}^0 = 0$, ρ^0, p^0, $B_x{}^0$, $B_z{}^0$ constant. The resulting equations are

$$\begin{aligned} \rho^0 \frac{\partial u}{\partial t} + \frac{\partial p}{\partial x} + B_z{}^0 \frac{\partial B_z}{\partial x} &= 0, \\ \rho^0 \frac{\partial v}{\partial t} - B_x{}^0 \frac{\partial B_y}{\partial x} + \frac{\partial p}{\partial y} + B_x{}^0 \frac{\partial B_x}{\partial y} + B_z{}^0 \frac{\partial B_z}{\partial y} &= 0, \\ \rho^0 \frac{\partial w}{\partial t} - B_x{}^0 \frac{\partial B_t}{\partial x} &= 0, \\ \frac{\partial p}{\partial t} + \gamma p^0 \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} &= 0, \\ \frac{\partial B_x}{\partial t} + B_x{}^0 \frac{\partial v}{\partial y} &= 0, \\ \frac{\partial B_y}{\partial t} - B_x{}^0 \frac{\partial v}{\partial x} &= 0, \\ \frac{\partial B_t}{\partial t} + B_z{}^0 \frac{\partial u}{\partial x} - B_x{}^0 \frac{\partial w}{\partial x} + B_z{}^0 \frac{\partial v}{\partial y} &= 0, \\ 0 \leqslant x, y \leqslant 1, \qquad 0 \leqslant t \leqslant 5. \end{aligned} \tag{4.5}$$

TABLE VI

Results for the Linearized Two-Dimensional MHD Equations

Method		Mesh	L_2 error	Phase error	*CPU* time
x direction	y direction				
Second order	Second order	32×32	0.034	0.0025	160
Fourth order	Fourth order	16×16	0.031	−0.0022	50
Fourier	Fourth order	8×16	0.034	−0.0024	141[a] 62[b]

[a] Using FORTRAN FFT.
[b] Using assembly language FFT.

In the x direction the solution is periodic, while in the y direction we impose the boundary condition $B_y = 0$. We impose only one boundary condition since the boundaries $y = 0$ and $y = 1$ are characteristic boundaries. In Table VI we present the results for this case where we again compute the phase error and the difference between the numerical and analytical solutions measured in the L_2 norm. In the y direction both second-order and fourth-order methods were used. In the x direction these methods as well as the Fourier collocation method ((2.14) and (2.15)) were used. Details are given in Table VI. As before, the fourth-order method with half the number of points per direction gave errors similar to the second-order method on the finer mesh. So at error levels of about 3% the fourth-order method was more than three times more efficient. As before, the fourth-order method is even more efficient at lower error levels. The use of the stabilized Fourier method again allowed a halving of the variables without affecting the accuracy. The timing for the Fourier program is based on an assembly language fast Fourier transform and using the fact that we deal only with real data. Slightly faster times were achieved by using a FORTRAN FFT for the specific N involved. A highly optimized general FORTRAN FFT more than doubled the total computer time. In conclusion the Fourier method allowed a reduction in storage but did not give any increase in speed over the fourth-order method to achieve the same accuracy. This conclusion is strongly machine dependent. When the analytic solution contains small wavelengths the Fourier method may become more efficient. Also, in this case the total accuracy of the problem is determined by the boundaries. In problems which are periodic in all directions the Fourier method is probably the most efficient (see e.g., [9, 29]). Further discussion of this code for the three-dimensional nonlinear MHD equations is presented in [15].

As a final application we consider the Euler equations in axisymmetric coordinates linearized about an arbitrary mean flow $u_0(z, r)$, $v_0(z, r)$. These equations are used in acoustic studies.

The equations are

$$\begin{aligned} s_t + (su_0 + u)_z + (sv_0 + v)_r + v/r &= F(r, z, t), \\ u_t + (s + u_0 u)_z + (v_0 u)_r &= u(v_0)_r - v(u_0)_r , \\ v_t + (u_0 v)_z + (s + v_0 v)_r &= v(u_0)_z - u(v_0)_z . \end{aligned} \tag{4.6}$$

$F(r, z, t)$ is a forcing term which is meant to simulate the source of sound in a jet. The domain of integration is $r > 0$, $\infty < z < \infty$, $t > 0$ and the main interest is the solution in the far field. Hence, it is necessary to reduce the storage via high-order methods to keep the problem manageable. This problem was solved for various u_0, v_0, F using the two-step methods ((2.6) and (2.8)) and splitting in a series of one-dimensional problems as described earlier. To simulate the infinite domain of integration, radiation boundary conditions were imposed based on asymptotics of the wave equation. Since a simulation is point dispersive at the origin a stretched coordinate system is used. For realistic mean flows this stretching is also needed to resolve gradients in the mean flow. Because of this stretching, the time step is determined by points near the axis of symmetry. Thus, we are able to choose time steps near the stability limit without affecting the overall accuracy. More details are presented in [2].

For the first case we consider zero mean flow $u_0 = v_0 = 0$ and so the analytic solution (with F a delta function) is known. We compare the two methods for a time of 70 units, approximately 400 time steps, using the second-order method. The computer times are for the STAR-100 at NASA Langley. Again, the fourth-order method with half the grid points per dimension has greater accuracy than the second-order method with the fine mesh. The efficiency gain is between 3 and 4 at large error tolerances and increases at lower error tolerances.

TABLE VII

Results for Acoustic Equations (4.6) with Realistic Meanflow[a]

Angle to axis	Fourth order $N = 8800$	Fourth order $N = 12000$	Second order $N = 8800$	Second order $N = 16000$
0	107	119	38	91
6.7	370	409	129	298
13.	790	847	299	702
19.1	1180	1255	500	1017
25.7	1614	1730	821	1474
32.4	2160	2205	1706	2091
38.4	2538	2611	2674	2619

[a] The pressure is calculated for a range of angles from the axis at 60 jet diameters. N is the total number of mesh points in the $z - r$ plane. Fourth order, $N = 12000$ is closest to analytic solution.

In the second case we consider a realistic mean flow modeled after experiments, and a harmonic source at two diameters downstream of the nozzle. The Mach number at the jet exit is 0.8, while the source oscillates with an intermediate frequency. In Table VII we present the peak acoustic pressure at 60 jet diameters, for a range of angles above the axis of symmetry. Varying the mesh and boundaries shows that the fourth-order method with 12000 points yields essentially the analytic result. Using this as a standard we see that the fourth-order method with 8800 points gives much better accuracy than the second-order method with 16000 points. For pulse sources it was not possible to use the second-order method and achieve reasonable accuracy due to storage limitations.

V. Conclusion

We have compared second- and fourth-order methods using both explicit and implicit codes, in both one and two space dimensions. In general the fourth-order methods were about three to five times faster in order to give the same accuracy at about the 5% level. The savings in storage is a factor of 2 per space dimension. For three-dimensional problems or smaller error tolerances the fourth-order methods are even more efficient.

Given most second-order methods in space and time it is straightforward to increase the space accuracy to fourth order. In the author's experience it has taken less than one day of coding to change large scale programs. The boundaries do not usually create difficulties, especially if characteristic corrections are used as given by (3.5) and described in [14]. In some cases where the boundary treatment is extra sensitive the fourth-order method may be less stable than the corresponding second-order method. The fourth-order scheme is most efficient for problems involving several time scales where the fastest time scale is of less physical significance. When stretched meshes are used, the efficiency also increases. In this case the time steps are determined by a small portion of the domain of interest. So for most of the region, the computational time step is small compared to the local speed of propagation. Hence, time errors are small and the spatial accuracy is the dominant factor.

These methods were also tried on a one-dimensional nozzle problem which was marched to a steady state. The analytic solution at the steady state contains a shock. As a function of different meshes a least-squares estimate of the rate of convergence was computed. Over the full mesh the error decreased as $(\Delta x)^{1/2}$ and Δx using the L_2 and L_1 norms, respectively. Excluding the shock region the error decreased as $(\Delta x)^n$ where n is the order of the scheme. Furthermore, the overshoots at the shock were not accentuated by the fourth-order method. Hence, the fourth-order method is efficient even for problems containing shocks.

Several cases were also considered using spectral methods. Here, the timings depended crucially on the fast Fourier transform that was used. For a one-dimensional generalized wave equation the Chebyshev collocation method was used. The efficiency of this method was even better than the fourth-order two-step method. Applications

to other more complicated situations is being pursued. We also used a Fourier collocation method for the periodic direction in a linearized two-dimensional MHD system. The Fourier method allowed a reduction in the storage requirements but did not decrease the computer time to achieve the given accuracy.

The second order in time and fourth order in space extension of MacCormack's method is presently being used in several large scale programs. These include modeling of noise in realistic jets, three-dimensional nonlinear MHD properties, as well as the full Navier–Stokes equations. These codes are running on the CDC-6600, Cyber 175, STAR-100, and CRAY computers. Hence, the method has been tested for efficiency and reliability under a variety of problems. The domains have included periodic regions, bounded regions with both noncharacteristic and characteristic boundaries, as well as unbounded regions which require the use of radiation boundary conditions. Hence, the robustness of the method under boundary conditions is also demonstrated.

It is important to choose the time step for the 2–4 method (2.8) so that the temporal errors are not larger than the spatial errors. For the wave equation it was found that a Courant number of 0.30 was efficient at error tolerances of about 20%. At the 5% error level Courant numbers of 0.25 or less were required for efficiency. Choosing too large a time step severely degraded the solution. In general it was better to choose too small a time step than too large a time step. For the acoustic equations (4.6) a coordinate stretching was used, and so the time step was determined by only a small region. In this case we even exceeded the formal stability limit without affecting the stability or accuracy of the numerical solution. Hence, for many problems of practical significance (i.e., they are stiff or else the time step is determined by a small region of mesh points) the time step can be chosen near the stability limit without any deterioration in the accuracy of the solution.

Higher-order methods may not always be advantageous or feasible. Higher-order methods usually require more computer time per time step than lower-order methods. Hence, efficiency is increased only if a coarser mesh can be used. There are various circumstances where the mesh is constrained by considerations other than accuracy and hence little is achieved by higher-order methods. One case is when the geometry of the problem demands a large number of points. For example, if one wishes to describe the many perturbations on a real wing, then one needs many more points than are needed for reasonable accuracy with a second-order method. Another example occurs in meteorological flows over the globe. The accuracy of any algorithm is limited by uncertainties in the physics of the model and in observational data. However, one cannot choose too coarse a grid or the topography of the earth is distorted. Similar situations occur in other fields where the basic equations being integrated have only limited validity. However, for the majority of cases where the mesh is constructed mainly on accuracy considerations the use of higher-order methods can lead to large savings in time and storage. Furthermore, the implementation of these methods frequently does not require large modifications to existing codes.

APPENDIX

The use of rational approximations to give fourth-order accuracy can also be applied to explicit schemes. We consider two applications. With the leapfrog method the scheme is mildly implicit, while with the MacCormack method the resulting scheme is fully explicit.

We consider

$$u_t + f_x = 0 \tag{A1}$$

with the semidiscrete approximation

$$u_j^{n+1} = u_j^{n-1} - 2\Delta t f_x \,. \tag{A2}$$

To approximate the derivative we use the Padé approximation $f_x \simeq (D_0/(\frac{2}{3} + \frac{1}{3}\mu)) f_j$ where $D_0 f_j = (f_{j+1} - f_{j-1})/2\Delta x$ and $\mu f = (f_{j+1} + f_{j-1})/2$. Clearing fractions we have

$$\tfrac{2}{3}u_j^{n+1} + \frac{1}{6}(u_{j+1}^{n+1} + u_{j-1}^{n+1}) = \tfrac{2}{3}u_j^{n-1} + \frac{1}{6}(u_{j+1}^{n-1} + u_{j-1}^{n-1}) - \frac{\Delta t}{\Delta x}(f_{j+1}^n - f_{j-1}^n).$$

This is an implicit equation for u^{n+1}. However, the coefficients of the implicit portion are constant and so the L and U factors can be computed once and stored. Due to the sparseness of these matrices the operation count is not high and the accuracy is fourth order in space. The scheme is still only conditionally stable.

With MacCormack's method we have

$$\begin{aligned} \bar{u}_j &= u_j^n - \Delta t f_x^{(1)}, \\ u_j^{n+1} &= \tfrac{1}{2}(u_j^n + \bar{u}_j^n - \Delta t f_x^{(2)}). \end{aligned} \tag{A3}$$

We approximate $f_x^{(1)}$ by a forward Padé approximation and $f_x^{(2)}$ by a backward Padé approximation and clear the fractions. We then have

$$\alpha\,\Delta\bar{u}_{j+1} + (1-\alpha)\,\Delta\bar{u}_j = -\frac{\Delta t}{\Delta x}(f_{j+1} - f_j), \tag{A4a}$$

$$\alpha\,\Delta u_{j-1}^{n+1} + (1-\alpha)\,\Delta u_j^{n+1} = \frac{-\Delta t}{2\Delta x}(\bar{f}_j - \bar{f}_{j-1}), \tag{A4b}$$

where $\Delta\bar{u}_j = \bar{u}_j - u_j$ and $\Delta u_j^{n+1} = u^{n+1} - \frac{1}{2}(\bar{u}_j + u_j^n)$. This method is fourth order in space and second order in time if $\alpha = (3^{1/2} - 1)/(2 - 3^{1/2})$. The scheme only involves the points $j-1$, and $j+1$ and so no additional difficulties arise near the boundaries. We find $\bar{u}$ at the right boundary by some one-sided third-order differences and then solve (A4a) for $\Delta\bar{u}_j$. We then find Δu_j^{n+1} at the left boundary and solve (A4b) for Δu^{n+1}. A linear stability analysis shows that the scheme is stable if $(\Delta t/\Delta x)A \leqslant 0.57$, $A = \partial f/\partial u$.

This scheme was tried on some of the test problems with the following scheme for the predictor at the right boundary

$$\Delta\bar{u}_N = -\frac{\Delta t}{\Delta x}\left[\left(2\alpha - \frac{17}{6}\right)f_N + \left(\frac{11}{2} - \frac{\alpha}{2}\right)f_{N-1} + \left(4\alpha - \frac{7}{2}\right)f_{N-2} + \left(\frac{5}{6} - \alpha\right)f_{N-3}\right] \tag{A5}$$

and a similar formula for Δu_1^{n+1}. In spite of the simplicity of the boundary treatment this method did not seem to offer any advantages over (2.8) and (2.9). For more complicated and less stable boundary conditions this situation may change.

REFERENCES

1. S. ABARBANEL, D. GOTTLIEB, AND E. TURKEL, *SIAM J. Appl. Math.* **29** (1975), 329.
2. A. BAYLISS AND E. TURKEL, *Comm. Pure Appl. Math.*, in press.
3. R. W. BEAM AND R. F. WARMING, *J. Computational Phys.* **22** (1976), 87.
4. W. R. BRILEY AND H. MCDONALD, "Lecture Notes in Physics," Vol. 35, p. 105, Springer-Verlag, New York/Berlin, 1974.
5. S. Z. BURSTEIN AND A. A. MIRIN, *J. Computational Phys.* **5** (1970), 547.
6. M. CIMENT, S. LEVENTHAL, AND B. WEINBERG, *J. Computational Phys.* **28** (1978), 135–166.
7. L. COLLATZ, "The Numerical Treatment of Differential Equations," Springer-Verlag, Berlin/New York, 1960.
8. R. COURANT AND K. O. FRIEDRICHS, "Supersonic Flow and Shock Waves," Interscience, New York, 1948.
9. B. FORNBERG, *SIAM J. Numer. Anal.* **12** (1975), 509.
10. J. GARY, *J. Computational Phys.* **26** (1978), 339.
11. D. GOTTLIEB AND S. ORSZAG, "Numerical Analysis of Spectral Methods: Theory and Applications," SIAM, Philadelphia, 1977.
12. D. GOTTLIEB AND E. TURKEL, *Math. Comp.* **30** (1976), 703.
13. D. GOTTLIEB AND E. TURKEL, *Studies in Appl. Math.*, to appear.
14. D. GOTTLIEB, M. GUNZBURGER, AND E. TURKEL, *SIAM J. Numer. Anal.*, in press.
15. W. GROSSMAN AND E. TURKEL, to appear.
16. B. GUSTAFSSON, H. O. KREISS, AND A. SUNDSTRÖM, *Math. Comp.* **26** (1972), 649.
17. B. GUSTAFSSON, *Math. Comp.* **29** (1975), 396.
18. D. J. JONES, J. C. SOUTH, AND E. B. KLUNKER, *J. Comp. Phys.* **9** (1972), 496.
19. E. KALNAY-RIVAS, A. BAYLISS, AND J. STORCH, *Briträge Phys. Atmos*, **50** (1977), 299.
20. B. KEYFITZ, *Comm. Pure Appl. Math.* **24** (1971), 125.
21. E. KRAUSE, E. H. HIRSCHEL, AND W. KORDULLA, *in* "Proceedings of AIAA Comp. Fluid Dynamics Conference," American Institute of Aeronautics and Astronautics (1973), 99.
22. H. O. KREISS AND J. OLIGER, *Tellus* **24** (1972), 199.
23. H. O. KREISS AND J. OLIGER, "Methods for the Approximate Solution of Time Dependent Problems," GARP No. 10, World Meteorological Organization 1973.
24. I. LINDEMUTH AND J. KILLEEN, *J. Computational Phys.* **13** (1973), 181.
25. R. W. MACCORMACK, "Lecture Notes in Physics," Vol. 8, p. 151, Springer-Verlag, New York/Berlin, 1971.
26. A. MAJDA AND S. OSHER, *Comm. Pure Appl. Math.* **30** (1977), 671.
27. J. OLIGER, *Math. Comp.* **28** (1974), 15.
28. J. OLIGER, *Math. Comp.* **30** (1976), 124.
29. S. ORSZAG, *Studies in Appl. Math.* **50** (1971), 293.
30. S. ORSZAG AND M. ISRAEL, *Annual Rev. Fluid Mech.* **6** (1974), 281.

31. S. G. RUBIN AND P. K. KHOSLA, *AIAA J.* **14** (1976), 851.
32. V. V. RUSANOV, *J. Computational Phys.* **5** (1970), 507.
33. G. SKOLLERMO, Department of Computer Science Report 62, Uppsala, 1975.
34. J. STEPPELER, *J. Computational Phys.* **19** (1975), 390.
35. G. W. STRANG, *Arch. Rational Mech. Anal.* **12** (1963), 392.
36. G. W. STRANG, *SIAM J. Numer. Anal.* **5** (1968), 506.
37. B. SWARTZ AND B. WENDROFF, *SIAM J. Numer. Anal.* **11** (1974), 979.
38. E. TURKEL, *J. Computational Phys.* **15** (1974), 226.
39. E. TURKEL, S. ABARBANEL, AND D. GOTTLIEB, *J. Computational Phys.* **21** (1976), 85.
40. D. L. WILLIAMSON AND G. L. BROWNING, *J. Appl. Met.* **12** (1973), 264.
41. H. J. WIRZ, F. SCHUTTER, AND A. TURI, *Math. Comput. in Simulation* **19** (1977), 241
42. S. F. WORNOM, NASA Tech. Paper 1302, 1978.