An Easily Implementable Fourth-Order Method for the Time Integration of Wave Problems

J. DE FRUTOS AND J. M. SANZ-SERNA

Departamento de Matemática Aplicada y Computación, Facultad de Ciencias, Universidad de Valladolid, Valladolid, Spain

Received April 29, 1991; revised February 2, 1992

We are concerned with the time-integration of systems of ordinary differential equations arising from the space discretization of partial differential wave equations with smooth solutions. A method is suggested that, while being as easily implementable as the standard implicit midpoint rule, is fourth-order accurate. The new method is symplectic so that it is very well suited for long-time integrations of problems with a Hamiltonian structure. Numerical experiments are reported that refer to a fourth-order Galerkin space discretization of the Korteweg–de Vries equation and to a pseudospectral space discretization of the same equation. © 1992 Academic Press, Inc.

1. INTRODUCTION

Most numerical schemes for the solution of evolutionary partial differential equations (PDEs) can be derived in two stages, following the well-known method of lines methodology (see, e.g., [30–32, 35]). First, the spatial variables are discretized by a finite-difference, finite-element, or spectral technique. This yields a system of ordinary differential equations (ODEs)

$$d\mathbf{U}/dt = \mathbf{F}(\mathbf{U}), \tag{1.1}$$

where t is the time and $\mathbf{U}(t)$ is an unknown d-dimensional vector whose entries are, typically, values of the PDE solution at the points of the spatial grid. In a second stage, the system (1.1) is integrated in time by means of an ODE numerical method. In this paper we are concerned with PDEs describing wave phenomena and having smooth solutions (hyperbolic systems of conservation laws with shock solutions are thus excluded). In the sort of problems we have in mind, the linear part of the right-hand side function $\mathbf{F}$ usually has purely imaginary eigenvalues, and, accordingly, the stability region [2] of the ODE solver used must have a nontrivial intersection with the imaginary axis. (For a survey of the relation between ODE and PDE numerical stability see [30].) Typical examples of ODE integrators with suitable stability regions include the explicit midpoint rule (leapfrog)

$$\mathbf{U}^{n+1} - \mathbf{U}^{n-1} = 2\tau \mathbf{F}(\mathbf{U}^n) \tag{1.2}$$

(throughout, superscripts denote time-levels and τ is the time-step), the trapezoidal rule

$$\mathbf{U}^{n+1} - \mathbf{U}^n = \frac{\tau}{2}[\mathbf{F}(\mathbf{U}^{n+1}) + \mathbf{F}(\mathbf{U}^n)] \tag{1.3}$$

and the (closely related) implicit midpoint rule

$$\mathbf{U}^{n+1} - \mathbf{U}^n = \tau \mathbf{F}(\tfrac{1}{2}[\mathbf{U}^{n+1} + \mathbf{U}^n]). \tag{1.4}$$

These three second-order methods are in fact in wide use. The explicit method (1.2) is extremely easy to implement and has a very low cost per step. On the other hand, it can operate only if τ is limited by a stability restriction. The implicit methods (1.3)–(1.4) can usually work with longer time-steps and this may or may not offset the disadvantage of their higher cost per step [13].

The backward-Euler scheme

$$\mathbf{U}^{n+1} - \mathbf{U}^n = \tau \mathbf{F}(\mathbf{U}^{n+1})$$

is, like (1.3) and (1.4), stable on the whole imaginary axis, but, for the problems we have in mind, is not competitive with the methods (1.2)–(1.4). First of all, this is only a first-order accurate method. Furthermore, on the imaginary axis, the amplification factor has modulus strictly less than 1, and the implied dissipation makes the method unsuitable for long-time integrations.

Very often there is no need to resort to something more sophisticated than (1.2)–(1.4); the errors coming from the space-discretization may be so significant that there is no purpose in using a high-order scheme for the time integration of (1.1). However, if for the spatial part a high-order finite-difference or finite-element method is employed, it is sensible to try and look for a higher order time-integrator. Such a need for high-order time-integrators is even more marked [34] when the spatial part is dealt with via a spectrally accurate technique [5, 10, 11, 15]. Fourth-order explicit Runge–Kutta methods are sometimes used with success. However, they suffer from stability step-size restrictions and for finite-difference and finite-element space discretizations such a restriction may represent a serious drawback (see Section 4.2 below). Furthermore, those methods are dissipative for small time steps and such a dissipation, even if small as it corresponds to a high-order method, may be unwelcome in long-time integrations (see Section 2.1). If we seek a suitable implicit high-order integrator, we realize that not very many are around. In the class of linear multistep methods we find the popular backward differentiation formulae of orders 3 to 6 which are not suited for the problems we are interested in, due to their poor stability properties along the imaginary axis. Implicit Runge–Kutta methods of high order [2, 16] are often believed not to be of practical value, because of the difficulties implied by their implementation.

The purpose of this paper is to present a nondissipative implicit fourth-order time-integrator which can be implemented as easily as the implicit midpoint rule (1.4). In fact a step with the new method is just a succession of three steps with (1.4). The new method is described in Section 2. Sections 3 and 4 contain two instances of the application of the suggested method. One of them refers to a modified Galerkin spatial discretization and the other to a pseudospectral spatial discretization. The final Section 5 is devoted to conclusions.

2. THE SUGGESTED METHOD

When the approximation $\mathbf{U}^n$ to the solution $\mathbf{U}$ of (1.1) at time t_n has been found, we determine the approximation $\mathbf{U}^{n+1}$ corresponding to time $t_{n+1} = t_n + \tau$ as follows. We first compute the auxiliary vector $\mathbf{Y}_1$ that solves

$$\begin{gathered}\mathbf{Y}_1 - \mathbf{U}^n = (\beta_1 \tau)\, \mathbf{F}(\tfrac{1}{2}[\mathbf{Y}_1 + \mathbf{U}^n]),\\ \beta_1 = (2 + 2^{1/3} + 2^{-1/3})/3 \simeq 1.3512.\end{gathered} \tag{2.1}$$

The vector $\mathbf{Y}_1$ is meant to approximate $\mathbf{U}(t_n + \beta_1 \tau)$. Once $\mathbf{Y}_1$ has been found we compute an auxiliary approximation $\mathbf{Y}_2$ to $\mathbf{U}(t_n + (\beta_1 + \beta_2)\tau)$ by solving

$$\begin{gathered}\mathbf{Y}_2 - \mathbf{Y}_1 = (\beta_2 \tau)\, \mathbf{F}(\tfrac{1}{2}[\mathbf{Y}_2 + \mathbf{Y}_1]),\\ \beta_2 = 1 - 2\beta_1 \simeq -1.7024.\end{gathered} \tag{2.2}$$

Finally we compute $\mathbf{Y}_3 \simeq \mathbf{U}(t_n + (\beta_1 + \beta_2 + \beta_3)\tau) = \mathbf{U}(t_{n+1})$ by solving

$$\mathbf{Y}_3 - \mathbf{Y}_2 = (\beta_3 \tau)\, \mathbf{F}(\tfrac{1}{2}[\mathbf{Y}_3 + \mathbf{Y}_2]), \qquad \beta_3 = \beta_1, \tag{2.3}$$

and then set $\mathbf{U}^{n+1} = \mathbf{Y}_3$. We follow standard Runge–Kutta terminology [2, 16] and say that each of the transitions $\mathbf{U}^n \mapsto \mathbf{Y}_1$, $\mathbf{Y}_1 \mapsto \mathbf{Y}_2$, and $\mathbf{Y}_2 \mapsto \mathbf{Y}_3$ is the computation of a *stage* of the method, whereas the overall transition $\mathbf{U}^n \mapsto \mathbf{U}^{n+1}$ is a *step* of the method. The main feature of (2.1)–(2.3) is that the computation of the ith stage, $i = 1, 2, 3$, is just the computation of a step of the midpoint rule (1.4) with steplength $\beta_i \tau$. Thus, to implement the suggested method it is in fact sufficient to implement the standard implicit midpoint rule. Furthermore, note that, once $\mathbf{Y}_1$ has been computed, there is no need to keep $\mathbf{U}^n$ in memory; once $\mathbf{Y}_2$ has been computed, there is no need to keep $\mathbf{Y}_1$, etc.; so that the overall step of the new method can be implemented with the same storage, one implements a step of the standard implicit midpoint rule.

2.1. *Theoretical Properties of the New Method*

As mentioned before, the method (2.1)–(2.3) is of order 4, i.e., $\mathbf{U}^n - \mathbf{U}(t_n) = O(\tau^4)$, as $\tau \to 0$ while t_n is kept constant. A proof of this fact has been given in [25], where the method was first mentioned. It is shown in [25] that if a concatenation of three implicit midpoint steps like (2.1)–(2.3) is to have order 3, then the coefficients β_i, $i = 1, 2, 3$, must satisfy $\beta_1 + \beta_2 + \beta_3 = 1$ and $\beta_1^3 + \beta_2^3 + \beta_3^3 = 0$. If, furthermore, one chooses $\beta_1 = \beta_3$ then the method is time-reversible and as a consequence its order of accuracy must be even, i.e., 4. This leads to the values of β_i, quoted in (2.1)–(2.3). Note that the equation $\beta_1^3 + \beta_2^3 + \beta_3^3 = 0$ implies that not all β_i can be positive. It is the high accuracy of (2.1)–(2.3) that makes the method potentially interesting in the first place.

An additional useful feature is that the method is *symplectic* or *canonical*; see the survey [24] and also [1, 3, 4, 6, 8, 9, 14, 17, 21–23, 25, 27, 28, 33]. This means that when (2.1)–(2.3) is applied to a problem (1.1) with a Hamiltonian structure, it will automatically inherit important qualitative features of the ODE system being integrated. Furthermore, for Hamiltonian problems, symplectic integrators have better long-time linear and nonlinear stability properties than their nonsymplectic counterparts. For instance, it may be shown rigorously [3] that, for many oscillation problems, canonical methods yield errors that grow linearly with time, where dissipative methods like explicit Runge–Kutta methods yield errors that grow quadratically. In general, there is a growing body of evidence [4, 6, 24] for the fact that problems (such as many wave propagation problems) having a Hamiltonian structure should be integrated by means of symplectic integrators, especially if one is interested in long-time simulations. Indeed, the methods (1.2), (1.4), which have always been in wide use for this sort of problems, are symplectic. The trapezoidal rule (1.3) is not symplectic. However, this method is closely related to (1.4). If a sequence $\{\mathbf{U}^n\}$ satisfies the recurrence (1.4), then [7] the averages $\{\tfrac{1}{2}(\mathbf{U}^n + \mathbf{U}^{n+1})\}$ satisfy (1.3). Hence (1.3) is, in a sense, equivalent to a symplectic method. (This equivalence only holds if τ is kept constant during the time integration. With variable time-steps it is well known from the ODE literature [7] that stability of the trapezoidal rule is inferior to that of the midpoint rule.) A fuller discussion of the advantages of symplectic integrators is not within the scope of the present paper and the interested reader is referred to the literature quoted above.

Turning now to the familiar linear stability analysis, let us recall that for the standard midpoint rule, the application to the model scalar problem $dU/dt = \lambda U$, λ complex, results in

$$U^n = R(\tau\lambda)^n\, U^0, \qquad R(z) = \frac{1 + z/2}{1 - z/2}.$$

The rational function R satisfies $|R(z)| \leqslant 1$ if and only if $\Re z \leqslant 0$ and hence the modulus of the numerical solution U^n grows if and only if the modulus of the theoretical solution $U(t)$ grows. In particular, the method is A-stable. In view of the structure of the new method, it is obvious, that, for the model problem, (2.1)–(2.3) yield

$$U^n = S(\tau\lambda)^n\, U^0,$$
$$S(z) = R(\beta_1 z)\, R(\beta_2 z)\, R(\beta_3 z).$$

Now β_2 is negative, so that $R(\beta_2 z)$ has a pole in the left half-plane $\Re z \leqslant 0$ and, as a result, the modulus of $S(z)$ cannot be bounded by unity in this half-plane. In other words, the new method is not A-stable. A detailed analysis of S reveals that if $\Re z \leqslant 0$ then $|S(z)| > 1$ for z inside a small island of almost circular shape, whose intersections with the real axis are given by $-2\beta_1^{-1}\, |\beta_2|^{-1/2} = -1.1344\cdots$ and $-2\, |2\beta_1\beta_2 + \beta_1^2|^{-1/2} = -1.2006\cdots$. The integration of a dissipative PDE by means of (2.1)–(2.3) would therefore require a stability step-size restriction. On the other hand, for z purely imaginary, $S(z)$ has unit modulus, so that the suggested method is stable on the whole imaginary axis (I-stable) and hence can integrate linear wave problems with arbitrarily long values of the time-step τ. Due to the symplectic character, the method also has good stability properties when applied to nonlinear Hamiltonian problems.

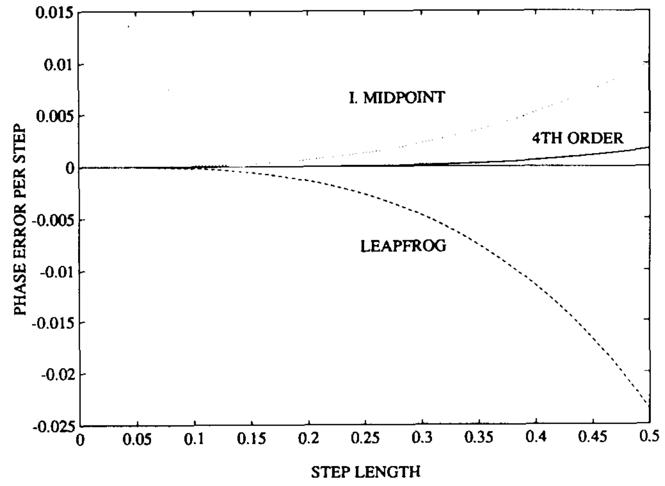

FIG. 1. Phase error $\theta - \arg R(i\theta)$ as a function of the nondimensional steplength θ.

Since S has unit modulus on the imaginary axis, the amplitude errors for (2.1)–(2.3) (or for (1.2) or (1.4)) are zero for purely imaginary eigenvalues. In Fig. 1 we have depicted, for the methods (1.2), (1.4), (2.1)–(2.3), the phase error per step $\theta - \arg R(i\theta)$ as a function of the nondimensional steplength θ ($i\theta$ equals the product of τ and the eigenvalue λ).

2.2. *Implementation*

Clearly how best to implement (2.1)–(2.3) is a problem-dependent issue that cannot be discussed once and for all (see Sections 3 and 4 below). Nevertheless the following remarks are in order. Let us start by recalling that to implement the standard implicit midpoint rule (1.4), it is computationally advisable [29] to introduce as an unknown the average $\mathbf{Z} = \frac{1}{2}[\mathbf{U}^{n+1} + \mathbf{U}^n]$ that satisfies

$$\mathbf{Z} - \mathbf{U}^n = \frac{\tau}{2}\, \mathbf{F}(\mathbf{Z}). \qquad (2.4)$$

Once (2.4) is solved for $\mathbf{Z}$, the next approximation is obtained via a simple extrapolation,

$$\mathbf{U}^{n+1} = 2\mathbf{Z} - \mathbf{U}^n.$$

Of course, except for the case where $\mathbf{F}$ is linear, (2.4) must be solved by means of some iterative procedure. The best choice of such a procedure is very much problem-dependent. As far as the initial guess $\mathbf{Z}^{[0]}$ to start the iteration for $\mathbf{Z}$, the following possibilities come easily to mind:

(i) Use the approximation $\mathbf{U}^n$ corresponding to the previous time level. This cheap initial guess is $O(\tau)$ away from the solution $\mathbf{Z}$ of (2.4).

(ii) Compute $\mathbf{Z}^{[0]}$ by means of Euler's rule $\mathbf{Z}^{[0]} = \mathbf{U}^n + (\tau/2)\, \mathbf{F}(\mathbf{U}^n)$. Now the initial guess is $O(\tau^2)$ away from the solution of (2.4). In PDE applications and for large values of τ, Euler's method, being explicit, will lead to a vector $\mathbf{Z}^{[0]}$, where the high Fourier modes are excessively represented (see, e.g., the discussion in [19]).

(iii) An alternative $O(\tau^2)$ initial guess can be obtained by linear extrapolation from $\mathbf{U}^n$, $\mathbf{U}^{n-1}$, i.e., $\mathbf{Z}^{[0]} = \frac{3}{2}\mathbf{U}^n - \frac{1}{2}\mathbf{U}^{n-1}$.

(iv) Quadratic extrapolation from $\mathbf{U}^n$, $\mathbf{U}^{n-1}$, $\mathbf{U}^{n-2}$ is also possible. However, care should be exercised. According to (2.4), $\mathbf{Z}$ is a backward-Euler solution and hence its local error is $O(\tau^2)$. Therefore, if we choose $\mathbf{Z}^{[0]}$ to be the value $\mathbf{Q}(t_n + \tau/2)$ at time $t_n + \tau/2$ of the quadratic interpolant $\mathbf{Q}(t)$

that fits the data $(t_n, \mathbf{U}^n)$, $(t_{n-1}, \mathbf{U}^{n-1})$, $(t_{n-2}, \mathbf{U}^{n-2})$, then we come up with an initial guess which is only $O(\tau^2)$ accurate (just as with the linear extrapolation above). On the other hand, if we recall that $\mathbf{Z} = \frac{1}{2}[\mathbf{U}^{n+1} + \mathbf{U}^n]$ and take into account that the local error in $\mathbf{U}^{n+1}$ is $O(\tau^3)$ (as corresponds to a second-order method), we conclude that an $O(\tau^3)$ initialization for $\mathbf{Z}$ is given by the average between $\mathbf{U}^n$ and the value $\mathbf{Q}(t_{n+1})$ of the interpolant $\mathbf{Q}(t)$ at time $t = t_{n+1}$. This yields

$$\mathbf{Z}^{[0]} = 2\mathbf{U}^n - \tfrac{3}{2}\mathbf{U}^{n-1} + \tfrac{1}{2}\mathbf{U}^{n-2}.$$

After this discussion of the implementation of the implicit midpoint rule, let us return to the fourth-order method. The idea behind (2.4) can be easily applied to rewrite (2.1)–(2.3) as follows:

$$\begin{aligned} \mathbf{Z}_1 - \mathbf{U}^n &= \frac{\beta_1 \tau}{2} \mathbf{F}(\mathbf{Z}_1), \\ \mathbf{Y}_1 &= 2\mathbf{Z}_1 - \mathbf{U}^n, \\ \mathbf{Z}_2 - \mathbf{Y}_1 &= \frac{\beta_2 \tau}{2} \mathbf{F}(\mathbf{Z}_2), \\ \mathbf{Y}_2 &= 2\mathbf{Z}_2 - \mathbf{Y}_1, \\ \mathbf{Z}_3 - \mathbf{Y}_2 &= \frac{\beta_3 \tau}{2} \mathbf{F}(\mathbf{Z}_3), \\ \mathbf{U}_{n+1} &= \mathbf{Y}_3 = 2\mathbf{Z}_3 - \mathbf{Y}_2. \end{aligned} \tag{2.5}$$

If $\mathbf{F}$ is nonlinear, each of the systems for $\mathbf{Z}_1$, $\mathbf{Z}_2$, $\mathbf{Z}_3$ in (2.5) must be solved by a suitable iteration. The initial guesses for $\mathbf{Z}_1$, $\mathbf{Z}_2$, $\mathbf{Z}_3$ may be obtained by procedures that directly generalize the technique in the points (i)–(iv) above. For $O(\tau^3)$ accurate initialization, we again resort to the quadratic interpolant $\mathbf{Q}(t)$ through the information at times t_n, t_{n-1}, t_{n-2} and set

$$\mathbf{Z}_1^{[0]} = \tfrac{1}{2}[\mathbf{U}^n + \mathbf{Q}(t_n + \beta_1 \tau)].$$

Once $\mathbf{Y}_1$ has been computed we initialize $\mathbf{Z}_2$ with

$$\mathbf{Z}_2^{[0]} = \tfrac{1}{2}[\mathbf{Y}_1 + \mathbf{Q}(t_n + (\beta_1 + \beta_2)\tau)].$$

Finally, when $\mathbf{Z}_2$ has been found, we set

$$\mathbf{Z}_3^{[0]} = \tfrac{1}{2}[\mathbf{Y}_2 + \mathbf{Q}(t_n + \tau)].$$

2.3. *Computational Cost*

The new method, while being as easily implementable as the implicit midpoint rule, has the advantage of its higher order. However, we should keep in mind that a step $n \to n+1$ of the new method is, in principle, as expensive as *three* steps of (1.4). Assumed first that (1.1) is a linear problem with time-independent coefficients, i.e., $\mathbf{F}(\mathbf{U}) = A\mathbf{U}$ for a suitable constant matrix A, and that the linear algebra is performed with a direct solver, such as Gaussian elimination. In these circumstances, (1.4) demands, once and for all, the factorization of the matrix $I - (\tau/2)A$ corresponding to (2.4), and then, at each time step, the solution of a linear system. On the other hand, (2.5) requires, once and for all, the factorization of two matrices $I - (\tau\beta_1/2)A$, $I - (\tau\beta_2/2)A$ and the solution of three linear systems per step. Thus in the present scenario and for a given steplength τ, the new method is, per step, exactly three times as expensive as the implicit midpoint rule. In other words, to equalize costs, the implicit midpoint rule should be run with a third of the steplength being used for the fourth-order method. It is of course an open question to see whether the accuracy advantage that (1.4) would obtain by operating with a smaller value of τ compensates for the higher order of the new method.

For nonlinear problems (or for linear problems with iterative linear algebra) it is not as easy to compare costs without actually performing experiments. In fact, the number of iterations of the procedure used to solve the algebraic equation depends on τ, since at lower values of τ better initial guesses are available. At any rate, it is clear that to equalize computational costs, the method (1.4) has to be run with smaller steplengths than the fourth-order method and it remains to be seen which is, in practice, the more efficient of the two. The remainder of the paper is aimed at gaining experience in this direction.

3. NUMERICAL ILLUSTRATION: FINITE ELEMENTS/FINITE DIFFERENCES

In this section and in the section below we present numerical tests of the suggested fourth-order method. We first study finite-element/finite-difference space discretizations and then move to spectral space discretizations. To keep the paper within reasonable length, we only present examples corresponding to the well-known Korteweg–de Vries (KdV) equation. This is a typical representative of the class of PDEs the new method is meant to deal with. Experiments (nor reported here) have also been performed which involved other equations, such as the nonlinear Schrödinger equation. The conclusions that can be derived from tests with alternative PDEs do not differ essentially from the conclusions for the KdV equation to be presented now.

3.1. *The KdV Equation*

We write the KdV equation in the form

$$u_t + 6uu_x + u_{xxx} = 0. \tag{3.1}$$

This form has been used by Nouri and Sloan in a recent comparison of pseudospectral methods for the KdV equation [18]. These authors considered both one-soliton and two-soliton solutions of (3.1). For reasons of brevity, we limit ourselves here to the most difficult one-soliton solution considered in [18]. This is given by

$$u(x, t) = 2\,\mathrm{sech}^2(x-4t), \tag{3.2}$$

so that the soliton amplitude is 2 and the soliton speed is 4. As in [18], we consider (3.2) between the initial time $t=0$ and the final time $t=2$ and limit the range of the spatial variable to $x_L = -20 \leqslant x \leqslant 20 = x_R$. Outside this spatial range (3.2) is 0 for all practical purposes.

We should emphasize at this stage that the aim of the experiments below is not to identify good numerical techniques for the KdV equation. On the contrary, we just want to ascertain whether, in the time-integration of the solutions of a *given* spatial semidiscretization, the new method is more or less efficient than the standard implicit midpoint rule.

3.2. *Space Discretization*

We use the fourth-order modified Galerkin space discretization suggested by Sanz-Serna and Christie [26]. We introduce a uniformly spaced grid in $[x_L, x_R]$ given by the points $x_j = x_L + hj$, $j = 0, 1, ..., J$, $h = (x_R - x_L)/J$, J a positive integer. If $U_j(t)$ denotes the approximation to $u(x_j, t)$, the functions $U_j(t)$ solve the system of differential equations

$$\begin{aligned}&\frac{1}{120}\dot{U}_{j-2} + \frac{26}{120}\dot{U}_{j-1} + \frac{66}{120}\dot{U}_j + \frac{26}{120}\dot{U}_{j+1}\\ &\quad + \frac{1}{120}\dot{U}_{j+2} - \frac{1}{8h}U_{j-2}^2 - \frac{10}{8h}U_{j-1}^2 + \frac{10}{8h}U_{j+1}^2\\ &\quad + \frac{1}{8h}U_{j+2}^2 - \frac{1}{2h^3}U_{j-2} + \frac{2}{2h^3}U_{j-1} - \frac{2}{2h^3}U_{j+1}\\ &\quad + \frac{1}{2h^3}U_{j+2} = 0, \qquad 0 \leqslant j \leqslant J.\end{aligned}$$

Here a dot represents differentiation with respect to time and it is assumed that U_{-2}, U_{-1}, U_{J+1}, U_{J+2} vanish for all values of t. Note that this system of ODEs is of the form

$$M\frac{d\mathbf{U}}{dt} = \mathbf{G}(\mathbf{U}), \tag{3.3}$$

where $\mathbf{U}$ denotes the $(J+1)$-dimensional vector with entries U_j, M is a (positive definite) mass-matrix, and $\mathbf{G}$ is a nonlinear function, whose Jacobian matrix possesses a pentadiagonal structure. The system (3.3) is not of the form (1.1) considered so far (i.e., is not solved for the time-derivatives). Nevertheless it is obvious how the methods in Sections 1 and 2 may be extended to cater for (3.3); for instance, the midpoint rule (1.4) becomes

$$M(\mathbf{U}^{n+1} - \mathbf{U}^n) = \tau\mathbf{G}(\tfrac{1}{2}[\mathbf{U}^{n+1} + \mathbf{U}^n]). \tag{3.4}$$

Alternatively, we can think that (3.3) has been rewritten in the format (1.1) by setting $\mathbf{F}(\mathbf{U}) = M^{-1}\mathbf{G}(\mathbf{U})$ and then the time-integrators are applied to the rewritten system. This alternative approach is of theoretical interest because it shows that properties such as the order of accuracy do not depend on whether the time-integrator is applied to systems of the form (3.3) or to systems of the explicit form (1.1). On the other hand, the alternative approach has nothing to offer from a practical point of view: to evaluate $M^{-1}\mathbf{G}(\mathbf{U})$ one, of course, solves a system with matrix M and one is back in formulations like (3.4).

3.3. *Time Discretization*

The system (3.3), with initial condition taken from (3.2) was integrated by both the implicit midpoint rule and the fourth-order method (2.5). It should be stressed that the midpoint rule (or the closely related trapezoidal rule) is a "natural" method for the time-integration at hand, in fact this was the technique employed in the original paper [26]. Our aim is to see whether the suggested method is competitive with such a "natural" method. The nonlinear system to be solved at each step of the midpoint rule and at each stage of the fourth-order method is treated by Newton iteration. A fresh Jacobian is computed and factorized at each step of (2.4), but the factors are not updated in the successive Newton iterations corresponding to a given time-step. Similarly, a fresh Jacobian is computed and factorized at each *stage* of (2.5), but the factors are not updated in the successive Newton iterations corresponding to a given stage. The initial guess at each step/stage is found by quadratic extrapolation as described in Section 2.2 above. This only works once three past time-levels are available, so that in the computation of $\mathbf{U}^1$ and $\mathbf{U}^2$ the initial guesses were taken to be $\mathbf{U}^0$ and $\mathbf{U}^1$, respectively.

3.4. *Numerical Results*

The algorithms considered in this section and in Section 4 below were implemented, using MATLAB, both on a Macintosh II personal computer and on a Sun SPARC-station. In (3.3) we set $h = 0.1$, a typical value. For this meshsize, the maximum norm error at the final time $t=2$ in the solution of the semidiscretization (3.3) is of about 3E−5. It may be thought that this is too small an error for a PDE solution, so that perhaps the value of h is unnecessarily small, However, it should be kept in mind that, following [18], errors are measured at time $t=2$, which is a small value; in practical simulations one is interested in

TABLE I

		Implicit midpoint			Fourth order		
τ	No. of steps	Error	Matrix factorizations	Linear systems	Error	Matrix factorizations	Linear systems
5.0E−2	40	—	—	—	1.6E−2	120	408
2.5E−2	80	3.1E−2	80	164	1.1E−3	240	631
1.25E−2	160	7.7E−3	160	322	3.4E−5	480	967
6.25E−3	320	1.9E−3	320	642	—	—	—
3.125E−3	640	4.4E−4	640	1282	—	—	—
1.5625E−3	1280	7.4E−5	1280	2051	—	—	—

monitoring the behaviour of solutions over much larger time intervals and one would accordingly find larger errors. Furthermore, it is only if small errors are aimed at that one should use the current fourth-order space discretization; for crude simulations the straightforward second-order method of Zabusky and Kruskal [36] would be a more suitable choice. In view of the spatial error to be reckoned with, we interrupted the Newton iteration when two consecutive approximations differed in less than 1E−6 in the maximum norm. The results are presented in Table I. For each method, the first column provides the maximum norm error, at $t=2$, of the fully discrete solution as an approximation to the true PDE solution (3.2). The second and third columns give the number of matrix factorizations and the number of linear systems solved. We see that to bring the error below the 1E−4 threshold (a most reasonable requirement when the spatial discretization introduces errors of about 3E−5), the fourth-order method can operate with a steplength $\tau = 0.0125$ and requires 480 matrix factorizations and the solution of 967 linear systems. The implicit midpoint rule requires $\tau = 1.5625\text{E}-3$, 1280 factorizations, and 2051 forward/backward solves.

In Fig. 2 we have depicted, in a loglog scale, error against

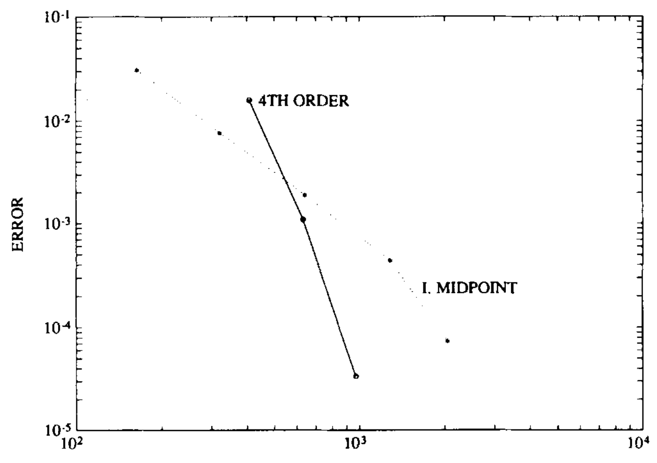

FIG. 2. Finite-difference results, $h=0.1$, $t=2$.

computational effort, as measured by the number of linear systems solved. It should be stressed that this, machine-independent measure of computational effort is biased and tends to favour the implicit midpoint rule. In fact, we see from Table I, that 642 systems for the implicit midpoint rule imply 320 matrix factorizations, whilst 631 systems for the fourth-order time-integrator go in hand with only 240 matrix factorizations. The figure shows that the second-order method would be more efficient than the fourth-order method if errors larger than 2E−3 were acceptable, but it is less efficient if smaller errors are required.

The point of view could be taken that, once the space discretization and spatial mesh have been suitably fixed, one should integrate in time in such a way that the time error is roughly as small as the error in the continuous-time, discrete-space solution $\mathbf{U}(t)$. The reason for this is that, for efficiency, if one is prepared to accept "large" errors from the time-integration, one should simultaneously move to a coarser spatial mesh and/or to a lower order spatial method. On the other hand, it is clearly wasteful to go on reducing the value of τ when the space discretization error provides a bottleneck for the size of the error in the full discrete solution. From such a point of view, Fig. 2 shows that, when the time-integrations are performed so as to have time errors as small as the spatial errors, the fourth-order method is more than twice as efficient as the implicit midpoint rule.

Experiments on the coarser spatial grid $h=0.2$ were also performed. Again one finds that, for small time errors, the fourth-order time-integrator should be preferred and that, for time errors of the size of the spatial error, the new method is more than twice as efficient as the second-order conventional method.

It is possible that the current implementation of the methods is not the best conceivable and that adjustments leading to gains in efficiency could be made. However, any improvement in the implementation of the implicit midpoint rule would automatically imply an improvement in the implementation of each stage of the new method. Therefore we feel strongly that the conclusions we have reached as to the relative merit of the methods would not have to be modified substantially if implementation refinements were introduced.

4. NUMERICAL ILLUSTRATION: PSEUDOSPECTRAL METHODS

We now move to pseudospectral space discretizations of (3.1). The time integration of these discretizations presents a number of specific features that did not manifest themselves in the case studied in the previous section.

4.1. *Space Discretization*

We again work with the spatial grid $\{x_j\}$ employed in the previous section, but we now suppose that all grid functions considered are periodic and take at x_R the same value they take at x_L. Hence the vector $\mathbf{U}$ of grid approximations is now J-dimensional, rather than $(J+1)$-dimensional. The semidiscretization is given by

$$\frac{d\mathbf{U}}{dt} = -6D\frac{1}{2}\mathbf{U}^2 - D^3\mathbf{U}, \tag{4.1}$$

with D the standard pseudospectral discretization of the $\partial/\partial x$ operator. In other words, if $\mathbf{V}$ is a grid-function, the vector $D\mathbf{V}$ is obtained by first fitting a trigonometric interpolant to the entries of $\mathbf{V}$ and then differentiating the interpolant with respect to x and evaluating the derivative at the grid points [5, 10, 11, 13, 15]. It is clear that D can be factorized as $\mathscr{F}^{-1}\Lambda\mathscr{F}$, where $\mathscr{F}$ denotes the matrix that transforms nodal values into Fourier coefficients and Λ is a diagonal matrix (Λ represents differentiation in Fourier-transformed space). The key points to be observed is that the errors in (4.1) decay faster than any power of h and that the matrix D is full.

4.2. *Time Discretization*

The system (4.1), with initial condition taken from (3.2), was integrated by both the implicit midpoint rule and the fourth-order time integrator. However, in the present application, the implicit midpoint rule is not a "natural" method for the time integration; it implies the solution of systems of coupled nonlinear algebraic equations that do not possess a banded structure (more on this later). In fact, we feel that most researchers would choose the explicit leapfrog method (1.2) for the nondissipative time-integration of (4.1) [11, 12]. This has the advantage of being easily implementable. Furthermore, the stability restriction $\tau = O(h^3)$ is not as bad as it is might be feared at first glance. In fact, with spectral techniques in space, all the Fourier modes that can be represented on a given grid are dealt with successfully by the space discretization. If a time-integrator is operating with the maximum value of τ allowed by the stability restriction, then the least stable Fourier mode, say mode number m, is being treated by the time integrator in a very crude way. But then, on efficiency grounds, one would be better off by using a coarser spatial grid in which mode number m were not represented. Therefore with spectral techniques the stability restriction is not very serious; on efficiency grounds, one is likely to apply the methods with lower values of τ than the largest allowed by the stability restriction (see the discussion in [34]). The situation is very different with finite-difference or finite-element methods. There the higher frequencies are grossly falsified by the process of spatial discretization, so that when it comes to the time-integration we want to integrate the high frequencies in a stable way, but not accurately. Hence τ should be chosen so that all modes are integrated in a stable way and the low modes are integrated accurately (see [20, Chap. 1]). For explicit time-integrators, the best value of τ is often close to the maximum allowed by the stability restriction (see, e.g., [13]).

Since, as explained above, the leapfrog method (1.2) is a good candidate for a time-integrator, we have implemented it as a reference method. It remains to be seen whether in the present circumstances an implicit method like (1.4) is competitive with (1.2) and whether the suggested fourth-order method (with three nonlinear systems to be solved per step) is competitive with the simpler methods (1.2) or (1.4).

To implement (1.2), it is best to work in Fourier-transformed space, where we have the recursion for the Fourier transforms $\mathscr{F}\mathbf{U}^n$,

$$\begin{aligned}&\mathscr{F}\mathbf{U}^{n+1} - \mathscr{F}\mathbf{U}^{n-1}\\ &\quad = 2\tau[-6\Lambda\,\tfrac{1}{2}\mathscr{F}[\mathbf{U}^n]^2 - \Lambda^3\mathscr{F}\mathbf{U}^n].\end{aligned} \tag{4.2}$$

Thus, per step, we perform an inverse Fourier transform to recover $\mathbf{U}^n$ from its Fourier coefficients and a direct Fourier transform to find $\mathscr{F}[\mathbf{U}^n]^2$ once $[\mathbf{U}^n]^2$ has been found by pointwise multiplication. On top of these Fourier/inverse Fourier transforms, which imply an $O(J\log J)$ computational cost if performed by FFT techniques, the implementation of (4.2) requires an $O(J)$ additional arithmetic operation. The missing level $\mathbf{U}^1$ can be found by a step of the explicit Euler rule in a standard way.

To implement (2.4) we also work in Fourier-transformed space to bring the matrix of the stiffest term u_{xxx} into diagonal form. Thus we have

$$\mathscr{F}\mathbf{Z} - \mathscr{F}\mathbf{U}^n = \frac{\tau}{2}\left[-6\Lambda\frac{1}{2}\mathscr{F}[\mathbf{Z}]^2 - \Lambda^3\mathscr{F}\mathbf{Z}\right]. \tag{4.3}$$

It is not sensible to solve (4.3) via Newton's method, since the Jacobian matrix of the quadratic term is full. We then resort to the iteration

$$\begin{aligned}&\mathscr{F}\mathbf{Z}^{[\nu+1]} - \mathscr{F}\mathbf{U}^n\\ &\quad = \frac{\tau}{2}\left[-6\Lambda\frac{1}{2}\mathscr{F}[\mathbf{Z}^{[\nu]}]^2 - \Lambda^3\mathscr{F}\mathbf{Z}^{[\nu+1]}\right],\\ &\qquad \nu = 0, 1, 2, \ldots.\end{aligned} \tag{4.4}$$

When $\mathscr{F}\mathbf{Z}^{[\nu]}$ is available, we perform an inverse Fourier transform to obtain $\mathbf{Z}^{[\nu]}$, multiply pointwise to obtain $[\mathbf{Z}^{[\nu]}]^2$, Fourier-transform $[\mathbf{Z}^{[\nu]}]^2$, and apply (4.4) to find $\mathscr{F}\mathbf{Z}^{[\nu+1]}$ (recall that Λ is a diagonal matrix). Thus an inner iteration for the midpoint rule (4.3) (i.e., a transition $\nu \to \nu+1$ in (4.4)) is as expensive as a step $n \to n+1$ of the leapfrog method (4.2). The starting guess $\mathbf{Z}^{[0]}$ is found by quadratic extrapolation.

The fourth-order method is implemented by setting up an iteration analogous to (4.4) at each stage. The initial guesses are found by quadratic extrapolation.

4.3. *Numerical Results*

Since the Fourier transforms are most efficient if J is a power of 2, we first took $J = 128$, which yields $h = 0.3125$ (64 modes were found not to resolve the problem). For $J = 128$ the maximum error in the time-continuous space-discrete solution is of about 3E−6. In the experiments to be reported, we interrupted the inner iterations of the implicit methods when the difference between two consecutive inner iterants was less than 5E−8. Since (4.4) is not a Newton iteration, its convergence is expected to be slow and one should be rather demanding in the stopping tolerance. In fact, inner tolerances above 5E−8 were found to damage the accuracy of the results.

The results are given in Table II, where, for the implicit methods we give, along with the error, the number of inner iterations. The same information is displayed in Fig. 3, where the (machine independent) unit of cost is taken to be one step for the leapfrog scheme and one inner iteration for the implicit methods or, equivalently, a pair fast Fourier transform/inverse fast Fourier transform. We see that, in agreement with a previous discussion, in the explicit scheme τ must be reduced well below the maximum value compatible for stability if one wants to reap the benefits of the accurate spatial discretization being used. For coarser errors the implicit midpoint rule is more expensive than the leapfrog method. However, for small values of τ that lead to accurate integrations, the implicit midpoint rule requires only one inner iteration per step. In this regime, one step of the implicit midpoint is as expensive as a leapfrog step and yields errors half as large. Therefore if one aims at time errors of a size comparable to that of the space errors, the implicit midpoint rule is a more efficient choice than the leapfrog method. This outcome was perhaps unexpected. Nevertheless, the figure clearly reveals that for accurate integrations the fourth-order method is the best choice, in spite of the computational cost stemming from the full matrices.

Let us finally consider the choice $J = 256$, with a tolerance of 5E−10 for the inner iteration. The results are given in Table III and Fig. 4 and show that the conjunction of a spectral technique in space with a high order time-integrator can efficiently provide extremely accurate simulations of partial differential equations. Admittedly the test equation has the advantage of being only one-dimensional, but, on other hand, it should be recalled that the simulations presented were carried out on a modest personal computer.

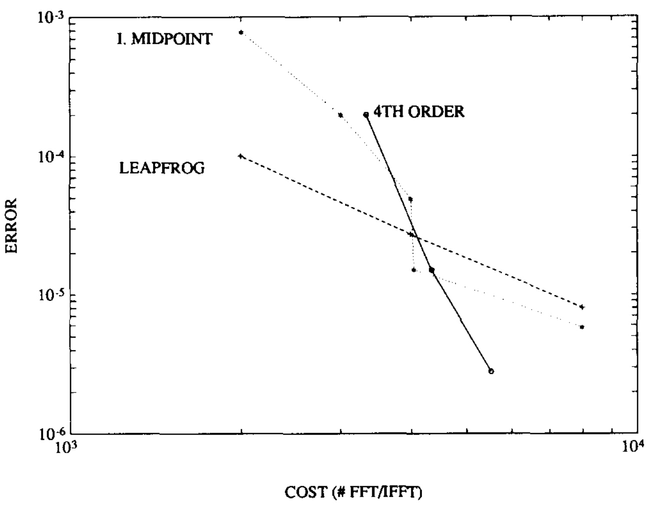

FIG. 3. Pseudospectral results, 128 Fourier modes, $t = 2$.

TABLE II

			Implicit midpoint		Fourth order	
τ	No. of steps	Leap frog Error	Error	Inner itera-tions	Error	Inner itera-tions
1.6E−2	125	Unstable	—	—	2.0E−4	3330
8.0E−3	250	Unstable	—	—	1.5E−5	4341
4.0E−3	500	Unstable	7.8E−4	2004	2.8E−6	5523
2.0E−3	1000	Unstable	2.0E−4	3004	—	—
1.0E−3	2000	1.0E−4	4.9E−5	4004	—	—
5.0E−4	4000	2.7E−5	1.5E−5	4047	—	—
2.5E−4	8000	8.0E−6	5.8E−6	8004	—	—

TABLE III

			Implicit midpoint		Fourth order	
τ	No. of steps	Leap frog Error	Error	Inner itera-tions	Error	Inner itera-tions
1.6E−2	125	Unstable	—	—	2.1E−4	4642
8.0E−3	250	Unstable	—	—	1.4E−5	6519
4.0E−3	500	Unstable	—	—	9.4E−7	9516
2.0E−3	1000	Unstable	2.0E−4	4006	7.0E−8	14015
1.0E−3	2000	Unstable	5.0E−5	6006	1.1E−8	18018
5.0E−4	4000	Unstable	1.2E−5	8006	—	—
2.5E−4	8000	Unstable	3.1E−6	16004	—	—
1.25E−4	16000	1.6E−6	7.7E−7	32004	—	—
6.125E−5	32000	4.0E−7	2.0E−7	32004	—	—
3.0625E−5	64000	1.0E−7	5.0E−8	64004	—	—

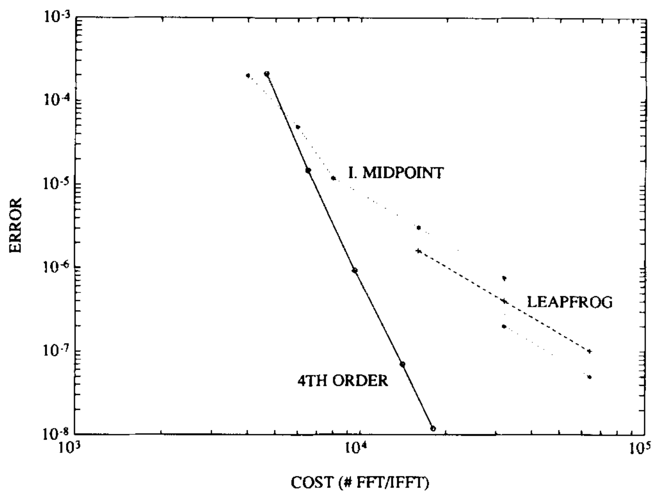

FIG. 4. Pseudospectral results, 256 Fourier modes, $t = 2$.

5. CONCLUSIONS

A fourth-order accurate time-integrator has been presented for the advancement in time of simulations of partial differential equations describing the motion of smooth waves. The suggested method is of symplectic type and can be implemented as easily as the standard midpoint rule. As a case study, we have considered the time integration of a Galerkin spatial discretization of the KdV equation and of a pseudospectral spatial discretization of the same equation. The experiments clearly show that the new method should be preferred to the implicit midpoint rule whenever high accuracy is required. We have also found that, contrary to a widely held opinion, explicit integrators may not always be the best choice in the time-integration of spectral semidiscretizations of the problems we are concerned with here.

ACKNOWLEDGMENTS

This research has been supported by "Junta de Castilla y León" under Project 1031-89 and by "Dirección General de Investigación Científica y Técnica" under Project PB89-0351.

REFERENCES

1. L. Abia and J. M. Sanz-Serna, Applied Mathematics and Computation Report 1990/8, Universidad de Valladolid, December 1990 (unpublished).
2. J. C. Butcher, *The Numerical Analysis of Differential Equations* (Wiley, Chichester, 1987).
3. M. P. Calvo and J. M. Sanz-Serna, *BIT* **32**, 131 (1992).
4. J. Candy and W. Rozmus, *J. Comput. Phys.* **92**, 230 (1991).
5. C. Canuto, M. Y. Hussani, A. Quarteroni, and T. Zang, *Spectral Methods in Fluid Dynamics* (Springer-Verlag, New York, 1988).
6. P. J. Channel and C. Scovel, *Nonlinearity* **3**, 231 (1990).
7. G. Dahlquist, in *Numerical Analysis*, edited by G. A. Watson (Springer-Verlag, Berlin, 1976), p. 60.
8. T. Eirola and J. M. Sanz-Serna, *Numer. Math.* **61**, 281 (1992).
9. K. Feng, *J. Comput. Math.* **4**, 279 (1986).
10. B. Fornberg, *SIAM J. Numer. Anal.* **12**, 509 (1975).
11. B. Fornberg, *Geophysics* **52**, 483 (1987).
12. B. Fornberg and G. B. Whitham, *Philos. Trans. R. Soc. London* **289**, 373 (1978).
13. J. de Frutos and J. M. Sanz-Serna, *J. Comput. Phys.* **83**, 407 (1989).
14. J. de Frutos, T. Ortega, and J. M. Sanz-Serna, *Comput. Methods Appl. Mech. Eng.* **80**, 417 (1990).
15. D. Gottlieb and S. A. Orszag, *Numerical Analysis of Spectral Methods: Theory and Applications* (SIAM, Philadelphia, 1977).
16. E. Hairer, S. P. Nørsett, and G. Wanner, *Solving Ordinary Differential Equations I, Nonstiff Problems* (Springer-Verlag, Berlin, 1987).
17. F. M. Lasagni, *Z. Angew. Math. Phys.* **39**, 952 (1988).
18. F. Z. Nouri and D. M. Sloan, *J. Comput. Phys.* **83**, 324 (1989).
19. M. A. Revilla, *Int. J. Numer. Methods Eng.* **23**, 2263 (1986).
20. R. D. Richtmeyer and K. W. Morton, *Difference Methods for Initial Value Problems* (Interscience, New York, 1967).
21. R. Ruth, *IEEE Trans. Nucl. Sci.* **30**, 2669 (1984).
22. J. M. Sanz-Serna, *BIT* **28**, 877 (1988).
23. J. M. Sanz-Serna, in *Advances in Numerical Analysis*, Vol. 1, edited by W. Light (Clarendon Press, Oxford, 1991), p. 147.
24. J. M. Sanz-Serna, *Acta Numerica* **1992**, 243 (1992).
25. J. M. Sanz-Serna and L. Abia, *SIAM J. Numer. Anal.* **28**, 1081 (1991).
26. J. M. Sanz-Serna and I. Christie, *J. Comput. Phys.* **39**, 94 (1981).
27. J. M. Sanz-Serna and F. Vadillo, in *Numerical Analysis*, edited by D. F. Griffiths and G. A. Watson (Longman, London, 1986), p. 187.
28. J. M. Sanz-Serna and F. Vadillo, *SIAM J. Appl. Math.* **47**, 92 (1987).
29. J. M. Sanz-Serna and J. G. Verwer, *IMA J. Numer. Anal.* **6**, 25 (1986).
30. J. M. Sanz-Serna and J. G. Verwer, *Appl. Numer. Methods* **5**, 117 (1989).
31. J. M. Sanz-Serna and J. G. Verwer, *Appl. Math. Comput.* **31**, 183 (1989).
32. J. M. Sanz-Serna, J. G. Verwer, and W. H. Hundsdorfer, *Numer. Math.* **42**, 405 (1987).
33. Y. B. Suris, *Zh. Vychisl. Mat. Mat. Fiz.* **29**, 202 (1987).
34. H. Tal-Ezer, *SIAM J. Numer. Anal.* **23**, 11 (1986).
35. J. G. Verwer and J. M. Sanz-Serna, *Computing* **33**, 297 (1984).
36. N. J. Zabusky and M. D. Kruskal, *Phys. Rev. Lett.* **15**, 240 (1965).