JOURNAL OF COMPUTATIONAL PHYSICS 67, 348–360 (1986)

# A Simple Adaptive Technique for Nonlinear Wave Problems

J. M. SANZ-SERNA

*Departamento Ecuaciones Funcionales, Facultad de Ciencias, Universidad de Valladolid, Valladolid, Spain*

AND

I. CHRISTIE

*Department of Mathematics, West Virginia University, Morgantown, West Virginia 26506*

Received May 20, 1985; Revised March 28, 1986

A method for the numerical integration of the nonlinear Schrödinger equation is derived which uses variable time steps and a moving spatial grid. The benefits of adaptation are clearly demonstrated in the numerical experiments reported. The simple technique employed to move the nodes can be applied with little coding effort to general one-dimensional systems of PDEs. © 1986 Academic Press, Inc.

## I. INTRODUCTION

The numerical treatment of the initial value problem for the nonlinear Schrödinger equation

$$iu_t + u_{xx} + q\,|u|^2\,u = 0, \qquad -\infty < x < \infty,\ 0 \leqslant t \leqslant T, \tag{1.1}$$

$$u(x, 0) = u_0(x), \qquad -\infty < x < \infty \tag{1.2}$$

($u$ complex, $i^2 = -1$, $q$ a given positive constant) has attracted much attention in the past few years (see Delfour et al. [4], Griffiths et al. [8], Herbst and Mitchell [9], Herbst et al. [10], Sanz-Serna and Manoranjan [17], Sanz-Serna [16], and Verwer and Sanz-Serna [18, 21]).

The *linear* Schrödinger equation

$$iu_t + u_{xx} = 0$$

models *dispersive* situations and its solutions have an amplitude which decays like $t^{-1/2}$ for $t$, $x \to \infty$, $x/t$ fixed (Whitham [24]). The cubic term in (1.1) opposes dispersion and thus makes it possible for the nonlinear Schrödinger equation to possess solutions where the competing forces of nonlinearity and dispersion balance each other exactly. The typical example of such a balanced solution is provided by the soliton.

The integration of bound states of solitons provides a stringent test for numerical schemes due to the large spatial and temporal gradients developing in the solution. In fact, Herbst et al. [10] and Sanz-Serna and Verwer [18] found that when working on a uniform grid in space and employing constant time steps, it was impossible to compute an accurate solution unless extremely small mesh sizes were chosen.

In this paper we attempt to overcome these difficulties by devising a scheme with *time and space* adaptive capabilities. The technique used for the time step control is the one commonly employed in ODE codes [19]. The adaptation in space (grid movement) is based on the equidistribution of the arclength of the solution and therefore can be regarded as a modification of the procedure suggested by White [22, 23]. In our code the time step $t_n \to t_{n+1}$ involves two stages. The first of these consists of advancing the solution on a *fixed* non-uniform grid by a suitable finite-element or finite-difference discretization. Substantial parts of existing fixed-grid programs can be employed to code this stage. The second stage is *problem independent* (i.e., would apply to any *one-dimensional* time dependent system of PDEs), only involves a moderate computational cost and can be coded once and for all as a subroutine.

The paper is divided into five sections. Section 2 reviews White's technique. Sections 3 and 4 describe the present procedure as applied to a *general* one-dimensional system of PDEs and to (1.1), respectively. Numerical experiments are reported in Section 5. The last section is devoted to a discussion of the scope of the present technique, its possible extensions and its relations to other available methods for node movement.

The literature on moving grids has grown substantially over the past few years; see in particular Miller and co-workers [13, 14, 7], Davis and Flaherty [2], Dwyer et al. [6] and the recent survey by Thompson [20], where further references can be found. A comparison of the numerical performance of the method in this paper with those of other available techniques is outside the scope of the present article.

## II. White's Technique

White [23] has suggested a technique for integrating numerically systems of $d$ time dependent, first order partial differential equations in one space variable $x$. His technique replaces the variables $x$, $t$ by the new variables $s$, $t$, where $s$ is an arclength-like coordinate. The advantage of this procedure is that in the new variables the solution $u(s, t)$ cannot develop large spatial gradients $\partial u/\partial s$. With White's technique $x$ becomes a dependent variable and therefore the transformed system involves $d+1$ scalar unknowns. Furthermore the transformation introduces a coupling between values of $u$ and $x$ which makes it impossible for the (sparse) discrete equations to possess banded form (see White [23, Fig. 6] for an illustration of the systems which appear).

## III. The Basic Procedure

We consider now a modification of White's approach whereby it is not necessary to carry out explicitly the change to $(s, t)$-variables. In the new procedure the discretization takes place in the original $(x, t)$-variables and there is no coupling between the computation of the values $U_j^n$ and the computation of the nodes $x_j^n$, so that the systems of equations to be solved in order to advance the solution in time are banded and only involve $d$ scalar unknowns per grid point.

Consider a *one-dimensional*, time-dependent system of the form

$$u_t = F(u, u_x, u_{xx}, x, t), \qquad x_L \leqslant x \leqslant x_R, 0 \leqslant t \leqslant T, \tag{3.1}$$

where $u$ is a $d$-dimensional vector. (The material in this section may be extended to more general one-dimensional situations in a straightforward manner.) The system (3.1) is supplemented by the initial condition

$$u(x, 0) = u_0(x), \qquad x_L \leqslant x \leqslant x_R \tag{3.2}$$

and by suitable boundary conditions.

Assume that at the time level $t_n$, we have computed a (non-uniform) grid $x_j^n$, $j = 0, 1, ..., J$, together with approximations $U_j^n$ to $u(x_j^n, t_n)$. In order to advance the solution and the grid up to time $t_{n+1} = t_n + \tau_{n+1}$, $\tau_{n+1} > 0$, we proceed in two stages:

(i) Use a suitable numerical scheme on the grid $\{x_j^n\}$ to obtain approximations $\bar{U}_j^{n+1}$ to $u(x_j^n, t_{n+1})$.

(ii) Join the points $(x_j^n, \bar{U}_j^{n+1})$ by straight lines, compute the length $\theta^{n+1}$ of the resulting polygon. Find the points $P_j^{n+1}$, $j = 0, 1, ..., J$, on the polygon which divide its total length into $J$ equal parts. Define the new nodes $x_j^{n+1}$, $j = 0, 1, ..., J$, as the projection of $P_j^{n+1}$ onto the $x$-axis. Compute $U_j^{n+1}$, $j = 0, 1, ..., J$, as an approximation to $u(x_j^{n+1}, t_{n+1})$ by means of a suitable interpolation of the values $x_j^n$, $\bar{U}_j^{n+1}$, $j = 0, 1, ..., J$.

Stage (i) above involves the same coding and computational effort as the advancement from $t_n$ to $t_{n+1}$ of the solution of (3.1) on a non-uniform *time-independent* grid. Therefore, a substantial part of existing non-adaptive codes could be used to construct adaptive programs. On the other hand, Stage (ii) is computationally inexpensive and is *problem independent*. It is therefore possible to code this stage once and for all. An efficient means of accomplishing this is presented in the following

*Algorithm*

```
(1) Given $J$, $x_j^n$, $\bar{U}_j^{n+1}$, $j=0, 1,\dots, J$, and $h_j^n$, $j=1, 2,\dots, J$, satisfying $h_j^n = x_j^n - x_{j-1}^n$.
(2) $S_0 = 0$.
(3) For $j=1, 2,\dots, J$. $S_j = S_{j-1} + ((h_j^n)^2 + \| \bar{U}_j^{n+1} - \bar{U}_{j-1}^{n+1} \|_2^2)^{1/2}$. Next $j$.
(4) $\delta = S_J/J$. $k=1$. $x_0^{n+1} = x_0^n$. $x_J^{n+1} = x_J^n$. $U_0^{n+1} = \bar{U}_0^{n+1}$. $U_J^{n+1} = \bar{U}_J^{n+1}$.
(5) For $j=1, 2,\dots, J-1$. $B = j\delta$.
(6) If $B \leqslant S_k$ go to (8).
(7) $k = k+1$. Go to (6).
(8) $x_j^{n+1} = x_{k-1}^n + (B - S_{k-1}) h_k^n / (S_k - S_{k-1})$.
(9) If $k \neq 1$ and $k \neq J$ compute $U_j^{n+1}$ as the value at $x_j^{n+1}$ of the cubic polynomial through $(x_{k+i}^n, \bar{U}_{k+i}^{n+1})$, $i = -2, -1, 0, 1$. If $k=1$, compute $U_j^{n+1}$ as the value at $x_j^{n+1}$ of the quadratic polynomial through $(x_{k+i}^n, \bar{U}_{k+i}^{n+1})$, $k = -1, 0, 1$. If $k=J$, compute $U_j^{n+1}$ as the value at $x_j^{n+1}$ of the quadratic polynomial through $(x_{k+i}^n, \bar{U}_{k+i}^{n+1})$, $k = -2, -1, 0$.
(10) $h_j^{n+1} = x_j^{n+1} - x_{j-1}^{n+1}$.
(11) Next $j$.
(12) $h_J^{n+1} = x_J^{n+1} - x_{J-1}^{n+1}$.
(13) Return $x_j^{n+1}$, $U_j^{n+1}$, $j=0, 1,\dots, J$; $h_j^{n+1}$, $j=1, 2,\dots, J$.
```

Some comments are in order. Step (3) computes the length $S_j$ of the polygon between $(x_0^n, \bar{U}_0^{n+1})$ and $(x_j^n, \bar{U}_j^{n+1})$. Steps (6) and (7) find the interval $x_{k-1}^n \leqslant x \leqslant x_k^n$ which contains the point $P_j^{n+1}$. Once $k$ is available $x_j^{n+1}$, $U_j^{n+1}$ can be readily computed. We have suggested cubic interpolation for $U_j^{n+1}$ as this is the form of interpolation we employed in the experiments reported later, but it is clear that other choices are possible.

In order to compute the initial grid $\{x_j^0\}$, the following procedure is adopted: The nodes are first placed provisionally according to a uniform distribution in $x_L \leqslant x \leqslant x_R$. Then the initial datum $u_0$ is evaluated at the nodes and a polygon fitted through the resulting values as described in stage (ii) above. Division of the total length of the polygon into equal parts leads to an improved set of nodes. The procedure is iterated until two consecutive grids differ by less than a small tolerance.

It is important to keep in mind that the arclength of a curve in the $x$, $u$ space depends on the choice of units for $x$ and $u$. In the subroutine presented above it has been tacitly assumed that both $x$ and $u$ represent suitable non-dimensional values, i.e., that (3.1) is written in non-dimensional form. In practice, it is not necessary to rewrite the system being solved to render it non-dimensional. It rather suffices to replace step (3) above by

(3′) For $j=1, 2,\dots, J$. $S_j = S_{j-1} + (\alpha (h_j^n)^2 + \| \bar{U}_j^{n+1} - \bar{U}_{j-1}^{n+1} \|_2^2)^{1/2}$. Next $j$.

Here, $\alpha$ is a positive parameter whose square root represents the ration between a typical (dimensional) $u$ value and a typical (dimensional) $x$ value.

Finally it has been found in practice that it is not advisable to have, at a given time level, elements of widely different sizes $h_j^n = x_j^n - x_{j-1}^n$, i.e., sizes which differ by several orders of magnitude. A simple means of ensuring that for each $n$

$$\max_j h_j^{n+1} \leqslant \sqrt{1+\beta} \min_j h_j^{n+1}, \tag{3.3}$$

where $\beta$ is a prescribed constant, is afforded by the replacement of step (3′) by

(3″) For $j = 1, 2, ..., J$. Temp $= \| \bar{U}_j^{n+1} - \bar{U}_{j-1}^{n+1} \|_2^2$. If temp $> \beta\alpha(h_j^n)^2$ then temp $= \beta\alpha(h_j^n)^2$. $S_j = S_{j-1} + (\alpha(h_j^n)^2 + \text{temp})^{1/2}$. Next $j$.

Now we have that

$$\sqrt{\alpha} h_j^n \leqslant S_j - S_{j-1} \leqslant \sqrt{1+\beta} \sqrt{\alpha} h_j^n$$

and (3.3) follows readily.

In our experiments we used $\beta = 100$ so that $\max h_j \lesssim 10 \min h_j$.

## IV. The Cubic Shroedinger Equation

In the time interval $0 \leqslant t \leqslant T$ under consideration the solutions of (1.1)–(1.2) in which we are interested are negligibly small outside an interval $x_L \leqslant x \leqslant x_R$. Therefore, in our numerical study, the pure initial value problem (1.1)–(1.2) is replaced by an initial-boundary value problem in $x_L \leqslant x \leqslant x_R$, $0 \leqslant t \leqslant T$, with homogeneous Neumann boundary conditions at $x = x_L$, $x_R$ (see [8, 9, 18]). For numerical work $u$ is decomposed into its real and imaginary parts $v$ and $w$, respectively. This leads to a system of $d = 2$ real partial differential equations. Assume that at $t = t_n$ a non-uniform grid $x_j^n$, $j = 0, 1, ..., J$, $x_0^n = x_L$, $x_J^n = x_R$ and corresponding approximations $V_j^n$, $W_j^n$ have been computed. The procedure for stepping to $t_{n+1} = t_n + \tau_{n+1}$ is as follows. First the system of partial differential equations is discretized in space on the grid $x_j^n$ by means of the usual three point replacement for $\partial^2/\partial x^2$ with the standard treatment of the boundary conditions. This results in a system of ordinary differential equations of the form

$$\frac{dU}{dt} = F_n(U) = S_n U + B(U) U, \tag{4.1}$$

where $U = U(t) = (U_0^T, ..., U_J^T)$ with $U_j = (V_j, W_j)^T$ and $S_n U$ and $B(U) U$ denote the contributions from the linear and nonlinear terms $u_{xx}$, $|u|^2 u$, respectively. The matrix $S_n$ is block-tridiagonal with $2 \times 2$ blocks depending only on the lengths $h_j^n$ and $B(U)$ is block diagonal.

We are interested in computing approximately the value at $t = t_{n+1}$ of the solution of (4.1) which at $t=t_n$ takes the value $U^n=(U_0^{nT},..., U_J^{nT})^T$, $U_j^n=(V_j^n, W_j^n)^T$, $j=0, 1,..., J$. In order to do so, and benefiting from the experience gained in [18], we take a single step of the implicit midpoint rule. More precisely $\bar{U}^{n+1}$ is computed from

$$\bar{U}^{n+1}=U^n+\tau_{n+1}F_n((U^n+\bar{U}^{n+1})/2), \tag{4.2}$$

where $\bar{U}^n=(\bar{U}_0^{nT},..., \bar{U}_J^{nT})^T$, $\bar{U}_j^n=(\bar{V}_j^n, \bar{W}_j^n)^T$, $j=0, 1,..., J$. Once $\bar{U}^{n+1}$ is available a new grid $\{x_j^{n+1}\}$ and a new solution $U^{n+1}$ are computed by means of the subroutine described in the previous section, thus completing the step $t_n \to t_{n+1}$.

The solution of the nonlinear system of algebraic equations (4.2) deserves some comments. An initial approximation $U^*$ to $\bar{U}^{n+1}$ is first computed by means of the standard Euler rule

$$U^*=U^n+\tau_{n+1}F_n(U^n).$$

This is followed by the corrector stages ($U_{[0]}=U^*$), $r=0, 1, 2,...,$

$$\left(I-\frac{1}{2}\tau_{n+1}S_n\right)U_{[r+1]}=\left(I+\frac{1}{2}\tau_{n+1}S_n\right)U^n + \frac{1}{2}\tau_{n+1}B\left(\frac{U^n+U_{[r]}}{2}\right)(U^n+U_{[r]})$$

until two consecutive iterates $U_{[r]}$, $U_{[r+1]}$ are found which differ (in the maximum norm) by less than a prescribed tolerance. (This was chosen to be $5\times 10^{-5}$ in our experiments.) Then, $\bar{U}^{n+1}$ is taken to be $U_{[r+1]}$. The corrector stages can be regarded as a modified Newton iteration for (4.2). See [18] for a further discussion of this point and for an efficient implementation of the corrector stages. We emphasize that our computations in the evolution $U^n \to \bar{U}^{n+1}$ are those of a time step of Method 0 in [18] except for the fact that now the grid $\{x_j^n\}$ is not uniform. The conservation of the $L^2$ norm in [18] can be shown easily to apply here, i.e., $\| U^n \| = \| \bar{U}^{n+1} \|$. On the non-uniform grid $\{x_j^n\}$ the $L^2$ norm of $U^n$ is given by

$$\| U^n \|^2=\frac{1}{2}h_1^n((V_0^n)^2+(W_0^n)^2)+\sum_{j=1}^{J-1}\left(\frac{h_j^n+h_{j+1}^n}{2}\right)((V_j^n)^2+(W_j^n)^2) + \frac{1}{2}h_J^n((V_J^n)^2+(W_J^n)^2).$$

Note that both a predicted value $U^*$ which is of first order accuracy in time and a corrected value $\bar{U}^{n+1}$ of second order are available and therefore it is possible to implement a standard control of the time steps based on local extrapolation and absolute error per step (Shampine and Gordon [19]). Namely, when $U^*$ and $\bar{U}^{n+1}$ have been obtained we compute the maximum norm

$$\text{EST}=\| U^*-\bar{U}^{n+1} \|_\infty$$

as an estimate of the error due to time stepping in (4.1). Note [19] that in doing so, one really estimates the error in the *lower order* approximation $U^*$. Here $\|\cdot\|_\infty$ denotes the usual $L^\infty$ norm for $2(J+1)$ real vectors. If EST does not exceed a prescribed tolerance TOL (which was 0.01 in our experiments) the vector $\bar{U}^{n+1}$ is accepted and (after the new grid $\{x_j^{n+1}\}$ is available) we are ready to take a new step up to $t_{n+2} = t_{n+1} + \tau_{n+2}$, $\tau_{n+2} = \tau_{\text{new}}$. If, on the contrary, EST > TOL the computed value $\bar{U}^{n+1}$ is discarded and the time step $t_n \rightarrow t_{n+1}$ is attempted again with a length $\tau_{\text{new}}$. In both instances $\tau_{\text{new}}$ is obtained from

$$\tau_{\text{new}} = 0.5\tau_{n+1}(\text{TOL}/\text{EST})^{1/2}.$$

The "safety" factor 0.5 is introduced to minimize the risk of rejections. In fact it was found that our algorithm *never* rejected steps and this was so even in problems with solutions whose character changed significantly during the time evolution. It is of course possible to increase the value 0.5 to, say, 0.8 or 0.9 but experiments in that direction were not undertaken.

## V. Numerical Experiments

In this section we apply the space/time adaptive algorithm outlined above to some of the test problems considered in [18]. For comparison we also use an algorithm based on a uniform $x$-grid with time adaption.

(A) *Single Soliton Solution.* Here, $q = 1$ and $u_0$ is given by

$$u_0(x) = \sqrt{2}\exp(0.5ix)\operatorname{sech} x$$

leading to the soliton solution

$$u(x, t) = \sqrt{2}\exp(i(0.5x + 0.75t))\operatorname{sech}(x - t).$$

In the $x$, $|u|$ plane this represents a wave of height $\sqrt{2}$ initially located at $x = 0$ and travelling to the right at speed 1 without changing its shape. Note, however, that the real and imaginary parts of $u$ are oscillatory. As in [18] we place the (artificial) boundaries at $x_L = -30$, $x_R = 70$ and take $T = 30$ to bear out the effects of long time integrations. In these, dissipative schemes are likely to reduce significantly the amplitude of the computed wave, while non-dissipative schemes are threatened by dispersion leading to spurious oscillations.

The problem was first discretized by means of central differences on a *fixed uniform* $x$-grid of width $h$, along with the variable time stepping procedure outlined in the previous section. Starting with $h = 1$, the interval length had to be halved twice ($h = 0.25$) before the theoretical solution was reproduced satisfactorily. Note that this corresponds to $J = 400$. The result is shown in Fig. 1, where virtually negligible upstream and downstream oscillations are still present. (The solid line

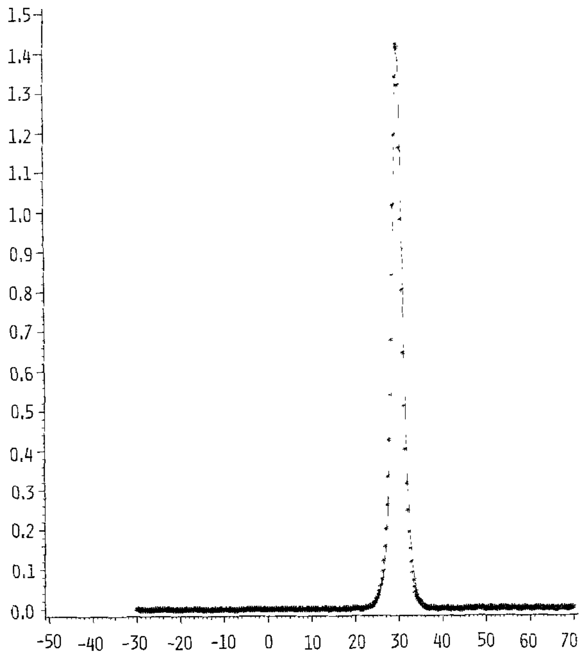

FIG. 1. $q=1$ solution at $T=30$. Uniform $x$-grid (401 nodes) with variable time stepping. $*$ denotes the numerical solution; ——denotes the exact.

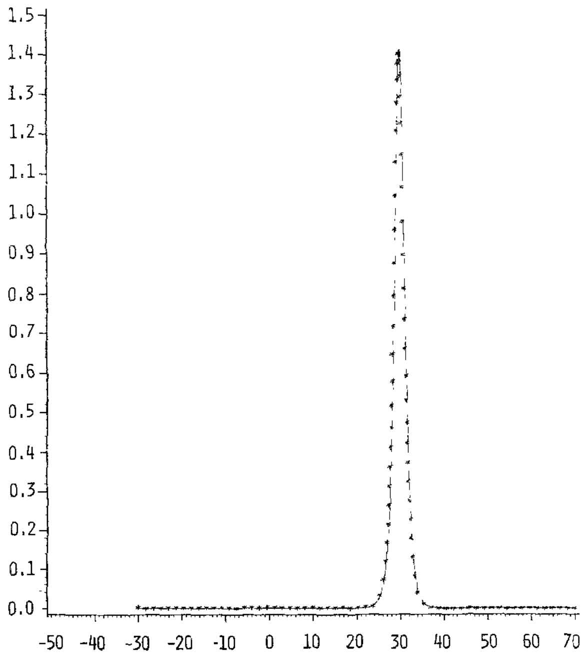

FIG. 2. As in Fig. 1 except that space adaptation (101 nodes) has been used.

TABLE I

| | *x*-Adaptive | | Non-adaptive | |
|---|---|---|---|---|
| *J* | $L^\infty$ | $L^2$ | $L^\infty$ | $L^2$ |
| 25 | 0.1030 | 0.3702 | not tested | |
| 50 | 0.0276 | 0.0676 | 0.7915 | 1.6604 |
| 100 | 0.0063 | 0.0161 | 0.5664 | 0.7688 |
| 200 | 0.0016 | 0.0039 | 0.1123 | 0.1904 |
| 400 | not tested | | 0.0256 | 0.0407 |

represents the theoretical solution which has moved from $x = 0$ to $x = 30$.) Approximately 600 time steps were taken with two or three iterations of the corrector per step.

Next, we employed the space/time-adaptive algorithm described in the previous section. In all the experiments the factor $\alpha$ governing the "aspect ratio" was set equal to $10^{-3}$. We did not attempt to "trim" the parameters $\alpha$ or $\beta$ and we emphasize that all the experiments in the paper correspond to one choice of $\alpha$ and

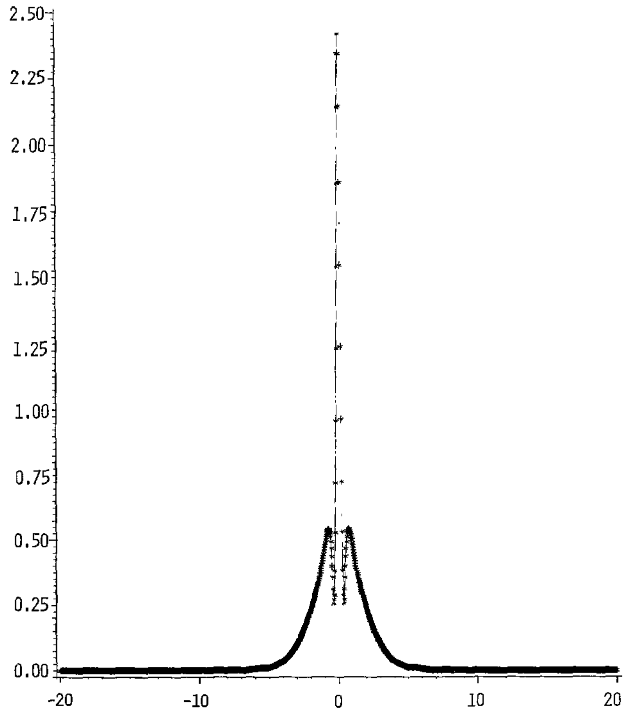

FIG. 3. $q = 18$ solution at $T = 0.98$. Uniform $x$-grid (1281 nodes) with variable time stepping. * denotes the numerical solution. Theoretical solution unavailable.

$\beta$. Starting with $J = 25$ we doubled $J$ twice before reaching, with $J = 100$, a solution of comparable accuracy to that of the scheme with no space adaption and $h = 0.25$. The result is depicted in Fig. 2, where no oscillations whatsoever are present. The number of time steps was similar to that in the fixed grid experiment.

Experiments were conducted in order to ascertain the convergence properties of the new technique. The results in Table I correspond to $t = 1$ and show an $O(J^{-2})$ behaviour in the error. Due to our choice of tolerance TOL, the contribution of the time stepping to the error is negligible. For comparison we have also included the error in the scheme without space adaption. The norms $L^{\infty}$, $L^2$ are as defined previously.

(B) *Bound State of Three Solitons* [12]. Now $q = 18$ and

$$u_0(x) = \operatorname{sech} x.$$

The solution of this problem is periodic in time and develops extremely large space and time gradients thus providing a stringent test to any numerical scheme [10, 18]. As in [18] we set $x_L = -20$, $x_R = 20$, $T = 0.98$. For the fixed-space, time-adaptive algorithm $h$ had to be successively halved until a value of 1/32 was found which provided an accurate solution. (The theoretical solution is not available and

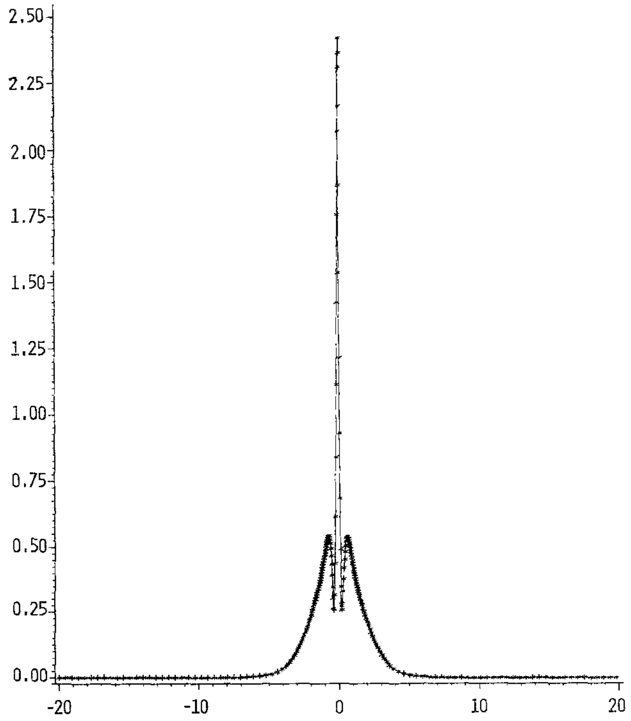

FIG. 4. As in Fig. 3 except that space adaptation (201 nodes) has been used.

accuracy was checked against a numerical solution calculated on a very fine mesh.) This value $L = 1/32$ corresponds to 1281 grid points. The solution at $T = 0.98$ is depicted in Fig. 3., where the computed points have been joined by straight lines. An approximate number of 450 time steps were taken with three correction iterations at each step regardless of the step length. This oscillates between 0.0051 at locations where the solution is smooth and 0.00086 at places with large gradients. In [18] a fixed time step of $\tau = 0.00625$ was used. One of the drawbacks encountered there stemmed from the fact that in the steps corresponding to large changes in the solution it was extremely costly or even impossible to get convergence in the iteration of the corrector.

Figure 4 corresponds to the space-time adaptive algorithm with $J = 200$. The solution with $J = 100$ was also found to be good except for a small phase error. The behaviour of the time stepping mechanism was similar to that of the fixed $x$-grid algorithm reported above. The number of steps required in Fig. 4 was 464.

## VI. Concluding Remarks

In White's original technique, the nodes $x_j$ move according to differential equations which are coupled to the evolution equations for the unknowns $U_j$. This coupling is also present in the approaches of Miller and co-workers [13,14, 7], Dukowicz [5] and Mosher [15]. In this paper the procedures for determining the nodes and computing the solution have been decoupled. Our rationale for doing so is that the node positions and solutions values do not play a symmetric role. When a set of nodes $\{x_j^n\}$ is given, the interest lies in computing a solution $U_j^n$ which comes as close as possible to the theoretical $u(x_j^n, t_n)$. However, we are *not* interested in obtaining *very accurately* any prescribed set of nodes $\{x_j^n\}$. It is sufficient to have a set of nodes which in some sense allows a satisfactory discretization of the solution at hand. An alternative technique for uncoupling the computation of the grid and that of the solution has been suggested by Manoranjan [11].

It should also be pointed out that our technique for determining the nodes is equivalent to the following procedure: First, compute approximately the integrals

$$S(x_j^n) = \int_{x_0^n}^{x_j^n} \left(1 + \left\| \frac{\partial u}{\partial p}(p, t_{n+1}) \right\|_2^2 \right)^{1/2} dp \tag{6.1}$$

by replacing the derivative by the piecewise constant function given by $(\bar{U}_j^{n+1} - \bar{U}_{j-1}^{n+1})/h_j^n$ in $x_{j-1}^n \leqslant p \leqslant x_j^n$. (This corresponds to Step (3) of our algorithm.) Then find, by inverse linear interpolation, the values $x_j^{n+1}$ so that the arclength between $x_L = x_0^{n+1}$ and $x_j^{n+1}$ is $j/J$ times the total $S(x_j^n)$ (Step (8)), so that as $j$ varies

$$\int_{x_{j-1}^{n+1}}^{x_j^{n+1}} \left(1 + \left\| \frac{\partial u(p, t_{n+1})}{\partial p} \right\|_2^2 \right)^{1/2} dp \sim \text{constant}. \tag{6.2}$$

When seen in this light the procedure appears to be closely related to that suggested by de Boor [3] in time independent situations. It is possible by merely changing Step (3) in the algorithm to replace the function $(1+\|u_p\|_2^2)^{1/2}$ being equidistributed in (6.1)–(6.2) by other functions. In particular, the choice $(1+\|u_p\|_2^2)^{1/2}$ and its modification have received much attention in the literature going back at least to Ablow and Schechter [1]. In fact, the reading of the first draft of the present paper has led M. A. Revilla to experiment with the choice $(1+\|u_{pp}\|_2^2)^{1/2}$. He has found [25] that this new equidistribution principle can lead to a further halving of the number of grid points required.

In our approach there is much freedom in the choice of method used to advance the solution $U^n \rightarrow \bar{U}^{n+1}$ on the fixed grid $\{x_j^n\}$. We also experimented with a discretization employing piecewise linear finite elements and found that it did not improve on the simple finite difference scheme described in Section 4. This is in contrast to the situation for uniform grids where finite elements perform significantly better than finite differences, see the experiment in Griffiths et al. [8]. The present paper thus supports Thompson's claim [20] that "when the grid adapts to the solution most algorithms work well."

Finally we would like to point out that there is no need to keep the number $J$ of subintervals constant in time. Our algorithm monitors the total length of the solution curve at each time step and it would be a simple matter to make $J$ proportional to that total length.

## REFERENCES

1. C. M. ABLOW AND S. SCHECHTER, *J. Comput. Phys.* **27** (1978), 351.
2. S. F. DAVIS AND J. E. FLAHERTY, *SIAM J. Sci. Statist. Comput.* **3** (1982), 6.
3. C. DE BOOR, *in* "Conference on the Numerical Solution of Differential Equations," edited by G. A. Watson (Springer-Verlag, New York, 1974), p. 12.
4. M. DELFOUR, M. FORTIN, AND G. PAYNE, *J. Comput. Phys.* **44** (1981), 277.
5. J. K. DUKOWICZ, *J. Comput. Phys.* **56** (1984), 324.
6. H. A. DWYER, R. J. KEE, AND B. R. SANDERS, *AIAA J.* **18** (1980), 1205.
7. R. J. GELINAS, S. K. DOSS, AND K. MILLER, *J. Comput. Phys.* **40** (1981), 202.
8. D. F. GRIFFITHS, A. R. MITCHELL, AND J. LL. MORRIS, *Compt. Meth. Appl. Mech. Eng.* **45** (1984), 177.
9. B. M. HERBST AND A. R. MITCHELL, Report NA/66, University of Dundee, 1983, unpublished.
10. B. M. HERBST, J. LL. MORRIS, AND A. R. MITCHELL, *J. Comput. Phys.* **60** (1985), 282.
11. V. S. MANORANJAN, Report NA/76, University of Dundee, 1984 (unpublished).
12. J. W. MILES, *SIAM J. Appl. Math.* **41** (1981), 227.
13. K. MILLER AND R. N. MILLER, *SIAM J. Numer. Anal.* **18** (1981), 1019.
14. K. MILLER, *SIAM J. Numer. Anal.* **18** (1981), 1033.
15. M. C. MOSHER, *J. Comput. Phys.* **57** (1985), 157.
16. J. M. SANZ-SERNA, *Math. Comp.* **43** (1984), 21.
17. J. M. SANZ-SERNA AND V. S. MANORANJAN, *J. Comput. Phys.* **52** (1983), 273.
18. J. L. SANZ-SERNA AND J. G. VERWER, *IMA J. Numer. Anal.* **6** (1986), 25.

19. L. F. SHAMPINE AND M. K. GORDON, *Computer Solution of Ordinary Differential Equations* (Freeman, San Francisco, 1975).
20. J. F. THOMPSON, *Appl. Numer. Math.* **1** (1985), 3.
21. J. G. VERWER AND J. M. SANZ-SERNA, Computing **33**, 297 (1984).
22. A. B. WHITE, JR., *SIAM J. Numer. Anal.* **16** (1979), 472.
23. A. B. WHITE, JR., *SIAM J. Numer. Anal.* **19** (1982), 683.
24. G. B. WHITHAM, "Linear and Nonlinear Waves" (Wiley–Interscience, New York, 1974).
25. M. A. REVILLA, *Int. J. Numer. Anal.*, in press.