JOURNAL OF COMPUTATIONAL PHYSICS 101, 94–103 (1992)

# An Adaptive Mesh Refinement Method for Nonlinear Dispersive Wave Equations

ERIC S. FRAGA

*Department of Computer Science, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1*

AND

JOHN LL. MORRIS

*Department of Mathematics and Computer Science, University of Dundee, Dundee, DD1 4HN, Scotland*

Received November 6, 1987; revised March 4, 1991

Adaptive mesh refinement techniques are often essential for solving nonlinear partial differential equations numerically. A new method for spatial grid refinement is developed and implemented. Several numerical experiments are performed to compare the method with results obtained using a uniform grid. The new method has the following properties: it is simple to implement and requires little modification of existing code to use; the solutions achieved as a result of using these methods prove to be accurate; and, the stability of the numerical methods is affected minimally. The effect of the grid refinement on essential properties of some of the equations, such as conservation, is minimized through the use of piecewise uniformity. 

## 1. NONLINEAR DISPERSIVE WAVE EQUATIONS

Recently, with the advent of techniques such as inverse scattering [1], there has been a shift in interest from linear partial differential equations to nonlinear differential equations. Of particular interest is a class of equations known as nonlinear dispersive wave equations. As the name suggests, these equations exhibit both dispersion and nonlinearity.

These nonlinear dispersive wave equations are of interest because of the many different problems they model. They are also of interest from a numerical point of view because, in general, solutions are not analytically available. With increases in the power and in the memory of computers, nonlinear problems have become tractable. It is now possible to attack these problems on computers that sit on a person's desk whereas, even a few years ago, these problems required huge mainframe computers. Nevertheless, special techniques to reduce the requirements needed to effectively solve these equations numerically are still needed, techniques such as adaptive mesh refinement.

A special property of some nonlinear dispersive wave equations is that the solutions may exhibit solitons. There are many different examples of these types of equations, each modelling several different physical problems. We shall concentrate on two equations, the Korteweg–de Vries and the nonlinear Schrödinger equations.

The Korteweg–de Vries (KdV) equation,

$$u_t + uu_x + \varepsilon u_{xxx} = 0, \tag{1}$$

where

$$u = u(x, t),$$
$$u(x, 0) = g(x),$$
$$-\infty < x < \infty,$$
$$t \geqslant 0,$$

and $\varepsilon$ is a real positive constant, is a simplified model of the full Euler equations for long waves. In particular, it is one of a set of eight equations which model long waves of small amplitude [10]. It arises in many areas, including plasma physics, shallow water wave problems, electric circuit theory with nonlinear capacitance, and in the theory of gravity waves in water of finite depth. The main properties of the KdV equation are that the waves propagate in one direction only and that solitons may appear given appropriate initial conditions.

The nonlinear Schrödinger (NLS) equation,

$$iu_t + u_{xx} + q\,|u|^2 u = 0, \qquad i^2 = -1, \tag{2}$$

where $u(x, t)$ is a complex valued function for $-\infty < x < \infty$,



0021-9991/92 $5.00

$t > 0$, and $q$ is a real valued parameter that describes the evolution of any weakly nonlinear, strongly dispersive almost monochromatic wave. The problems modelled by this equation may appear in laser optics, plasma physics, deep water wave models, etc. The NLS equation may also exhibit soliton solutions. An important feature of the NLS is that the waves may propagate in both directions.

### 1.1. Solitons

The two equations above, as well as others such as the sine–Gordon equation, may exhibit soliton solutions given the appropriate initial conditions [4]. Solitons appear as a result of the balance between the dispersion and the nonlinearity of the equations. Whereas dispersion will tend to spread out the wave, the nonlinear component of the equation tends to steepen the wave. Solitons were first reported by John Scott Russell in 1844 [11]. Since then, this solitary wave phenomenon has appeared in many diverse areas of mathematics and physics, including meteorology, elementary particle physics, plasma studies, and laser physics. The term, *soliton*, was not coined until 1967 by Zabusky and Kruskal [17].

It is important to note that the soliton is not a feature of all nonlinear partial dispersive wave equations which exhibit solitary waves. A soliton is more than just a solitary wave, having the property that its shape is not affected by collisions with other solitons.

The KdV and NLS equations allow waves of different speeds, so collisions of solitons are possible. For the KdV equation, solitons propagate in only one direction and the speeds of the solitons are directly proportional to their amplitudes. For the NLS equation, solitons may propagate in both directions and there is no direct relationship between the speed and the amplitude.

For the numerical analyst, solitons are interesting because many of the unsolved problems related to soliton-like behaviour are not solvable by the inverse scattering method. Few or no analytical tools exist for many of these problems.

## 2. MESH REFINEMENT

The numerical solution of nonlinear partial differential equations usually involves an implicit finite difference or Galerkin-type scheme. Due to the nonlinearity of the system, these methods require small spatial and temporal mesh spacing to achieve reasonable accuracy and to avoid nonlinear instability. A second property of the type of nonlinear partial differential equations characterized by the KdV and NLS equations is that they model long time problems. These requirements, in combination, imply that a great deal of computer time is necessary. Since only very specific areas of the solution require a fine mesh to achieve this high accuracy, mesh refinement is both essential and applicable. A suitable mesh refinement technique will generate a small mesh spacing where necessary and a coarser spacing, where the solution is not as critical. With mesh refinement, new problems may be solved using resources currently available, problems that are intractable using a uniform mesh due to time or space requirements.

There are two general methods for adaptive mesh refinement reported in the literature. The methods of Berger and Oliger [2] for hyperbolic partial differential equations and of Flaherty and Moore [5] for parabolic equations rely on error extrapolation to determine parts of the mesh to refine recursively. The other type of method uses an equidistribution principle to generate a nonuniform mesh. The usual case involves equidistributing some derivative of the solution or the arclength of the solution itself. Examples for this method include those described by White [16], Manoranjan [9], and Sanz-Serna and Christie [13]. Extensive reviews of this second type of method have been given by Thompson [14] and Russell and Christiansen [12].

### 2.1. A Novel Method

The method we wish to introduce is based on the geometric description of the solution and differs in approach from either type mentioned above: error estimation is not used and no particular quantity is equidistributed. The overall mesh created is nonuniform, although it will actually consist of a set of contiguous non-overlapping uniform sub-meshes.

The basic approach is as follows: locate each soliton in the solution, place a fine mesh so as to cover the support for each soliton, and fill in the gaps between each fine mesh with coarsely spaced points. Each sub-interval, be it the support for a soliton or a gap between two solitons, is discretized uniformly. However, any numerical method used will work on the whole mesh at once, considering it to be a non-uniform mesh overall.

#### 2.1.1. *Locating the Solitons*

The first step is best described in the form of an algorithm, shown in Fig. 1. $S_{\min}$ is used to calculate the cutoff value used to determine the location of solitons. Any value in the solution above this cutoff value is considered to be part of a soliton. In practice, we have found that $S_{\min} = 50$ (i.e., a cutoff value equal to 2% of the range of values in the solution) works well.

Applying this algorithm to the set of solution values will identify the location of every soliton (or group of solitons) in the solution, although this method may fail in extreme cases for a given value of $S_{\min}$. Since the cutoff value is set according to the range of function values, if the range of sizes of solitons is great, some of the smallest solitons may

```
cutoff ← v_min + (v_max − v_min)/S_min
j ← 0
for i ← 1 to n do
    if not in_soliton then
      if v_i > cutoff then
        in_soliton ← true
        j ← j + 1
        start_of_soliton_j ← i − 1
      endif
    else
      if v_i < cutoff then
        in_soliton ← false
        end_of_soliton_j ← i
      endif
    endif
endfor
number_of_solitons ← j
```

**FIG. 1.** Locating the solitons.

be missed. That is, if there are any solitons whose amplitudes are smaller than *cutoff* (defined in Fig. 1), these solitons will be skipped over. $S_{\min}$ may be increased for these cases. Increasing $S_{\min}$, however, may cause the method to find some spurious oscillations and treat them as solitons. Nevertheless, this will only increase the number of points in the mesh and will not affect the numerical method. The limiting case, as $S_{\min} \to \infty$, is a uniform mesh covering the complete interval defined for the problem.

The step described above locates the solitons but does not define the full domain of each soliton. The intervals, denoted by $I_j = [a_j, b_j]$, $j = 1, ..., ns$, where $a_j = start\_of\_soliton_j$, $b_j = end\_of\_soliton_j$, and $ns = number\_of\_solitons$, are extended on both sides by an amount proportional to the size of each interval: new intervals are defined as

$$I_j' \leftarrow [a_j - \beta \times |I_j|, b_j + \beta \times |I_j|], \qquad j = 1, ..., ns. \quad (3)$$

The value of $\beta$ is determined using the initial condition. It is defined as the minimum value which allows all solitons in the initial condition to be represented accurately on the non-uniform mesh in terms of the $L_2$ energy.

The intervals $I_j'$, although sufficient to cover the support of each soliton, need to be extended even further. Due to the evolutionary aspect of the problems, the solitons may move from one time step to the next. If the intervals are not extended, part of each soliton may be outside its respective interval after a time step. Therefore, we further extend the original intervals $I_i$:

$$I_j' \leftarrow [a_j - \text{buffer} \times \beta \times |I_j|, b_j + \text{buffer} \times \beta \times |I_j|], \quad j = 1, ..., ns, \quad (4)$$

where *buffer* depends on the problem being solved. In practice, we have used values ranging from buffer = 1 to buffer = 3.

The procedure of defining the intervals that cover the solitons, with all the necessary extensions, may cause two or more intervals to overlap. After each extension above, any overlapping intervals are combined into one bigger interval before proceeding with the next step.

### 2.1.2. *Defining the New Mesh*

A mesh may be defined with this final set of intervals. The mesh consists of several contiguous uniform sub-meshes. Each interval corresponds to one uniform sub-mesh with a spatial discretization $h_{\text{goal}}$. The gaps between each interval (and before the first interval and after the last interval) are also discretized uniformly, with $N_{\text{gap}}$ points placed in each gap. The total number of points $N$ is given by

$$N = N_{\text{gap}} + \sum_{j=1}^{ns} \left[ \frac{|I_j'|}{h_{\text{goal}}} + N_{\text{gap}} \right] + 1, \quad (5)$$

where $ns$ is the number of intervals after all extensions and overlaps are taken care of. $N$ is not constant over time as it depends on the numbers of solitons and the width of the intervals used to cover the solitons.

The value of $h_{\text{goal}}$ is also determined using the initial condition. It is the maximum value such that the $L_2$ energy of the initial condition, discretized using a uniform mesh, is resolved accurately.

### 2.1.3. *Summary*

This novel mesh refinement method is based on the geometric properties of the solution to a partial differential equation. The main advantages of this method are due to its simplicity: it is easy to implement, adds little overhead, and places points in the mesh only where they are needed. Since it refines only in the spatial dimension, any time integration scheme may be used. In particular, schemes which are adaptive in time should work well in conjunction with this method.

Although the method has been developed for soliton based solutions, any problem which exhibits soliton-like solutions may make use of the same techniques. Likewise, it is not necessary to use function values to determine the areas that need refinement. Other values may be used with only minor changes to the algorithm. For example, the first

derivative could be used which would allow the method to refine only where a solution was steep. This would be well suited to shock-type problems.

## 3. NUMERICAL EXPERIMENTS

The performance of the new method, labelled *soliton* in the discussion that follows, is evaluated by solving several test problems. A method based on a uniform mesh is used for comparison. Both the KdV and NLS equations with appropriate initial conditions are solved.

For notational purposes, vectors are denoted by bold lowercase letters and matrices by uppercase letters. In the following tables, no values for $n$, the number of points in the mesh, for the *soliton* method are given. The number of points changes as solitons move and interact. The value of $h$ given for the *soliton* method is that described, in the previous section, as $h_{\text{goal}}$.

All runs were carried out on a SUN 3/160 computer with the MC68881 floating point co-processor in the Department of Mathematics and Computer Science and the University of Dundee.

### 3.1. The Korteweg–de Vries Equation

#### 3.1.1. *The Numerical Method*

We now introduce the numerical method used to test the *soliton* method on the KdV equation (1). The first step in developing the method involves a semi-discrete approximation to (1):

$$\frac{d\mathbf{w}}{dt} = M(\mathbf{w}) \cdot \mathbf{w}, \tag{6}$$

where $\mathbf{w}(t)$ is the semi-discrete approximation to $u(x, t)$ with $w_i(t) \approx u(x_i, t)$, $i = 0, ..., n$, and $a = x_0 < x_1 < \cdots < x_n = b$.

One standard approximation [17] is

$$u_{xxx}|_{x=x_i} \approx \frac{1}{2h^3}(-w_{i-2} + 2w_{i-1} - 2w_{i+1} + w_{i+2}) = (L \cdot \mathbf{w})_i \tag{7}$$

and

$$uu_x|_{x=x_i} \approx \frac{(w_{i-1} + w_i + w_{i+1})}{3} \frac{(w_{i+1} - w_{i-1})}{h} = (N(\mathbf{w}) \cdot \mathbf{w})_i, \tag{8}$$

where $M(\mathbf{w}) \cdot \mathbf{w} = -N(\mathbf{w}) \cdot \mathbf{w} - \varepsilon L \cdot \mathbf{w}$ ($L$ and $N$ correspond to the linear and nonlinear parts of $M$, respectively) and $h = x_{i+1} - x_i$, $i = 0, ..., n-1$. This approximation, however, is only valid on a uniform mesh. In order to solve the KdV equation on a non-uniform mesh, we need to modify this approximation. Consider the generalization of each approximation:

$$u_{xxx}|_{x=x_i} \approx \sum_{\substack{j=i-2 \\ j \neq i}} \alpha_{i,j} w_j, \qquad i = 2, ..., n-2, \tag{9}$$

and

$$uu_x|_{x=x_i} \approx \frac{(w_{i-1} + w_i + w_{i+1})}{3}(\beta_{i,i-1} w_{i-1} + \beta_{i,i+1} w_{i+1}), \qquad i = 1, ..., n-1. \tag{10}$$

Expanding the right-hand sides using Taylor series and solving for the $\alpha_{i,j}$'s and $\beta_{i,j}$'s, we obtain

$$\alpha_{i,i-2} = \frac{-6}{h_{i-2}(h_{i-2} + h_{i-1} + h_i)(h_{i-2} + h_{i-1} + h_i + h_{i+1})}$$

$$\alpha_{i,i-1} = \frac{6}{h_{i-2}(h_{i-1} + h_i)(h_{i-1} + h_i + h_{i+1})}$$

$$\alpha_{i,i+1} = \frac{-6}{h_{i+1}(h_{i-1} + h_i)(h_{i-2} + h_{i-1} + h_i)}$$

$$\alpha_{i,i+2} = \frac{6}{h_{i+1}(h_{i-1} + h_i + h_{i+1})(h_{i-2} + h_{i-1} + h_i + h_{i+1})}, \qquad i = 2, ..., n-2,$$

and

$$\beta_{i,i-1} = \frac{-1}{h_{i-1} + h_i}$$

$$\beta_{i,i+1} = \frac{1}{h_{i-1} + h_i}, \qquad i = 1, ..., n-1,$$

where $h_i = x_{i+1} - x_i$. This general approximation for a non-uniform mesh reduces to that given in (7) and (8) for a uniform mesh.

The ordinary differential equation (6) may now be solved. Solving it directly, approximating the integral using the midpoint rule, and the exponential with a 1–1 Padé approximation, we obtain

$$\left[I - \frac{\tau}{4} M(\mathbf{v}^{m+1}) + M(v^m))\right] \mathbf{v}^{m+1} = \left[I + \frac{\tau}{4} M(\mathbf{v}^{m+1}) + M(\mathbf{v}^m))\right] \mathbf{v}^m, \tag{11}$$

where $\mathbf{v}$ is the fully discrete approximation to $u$ with $v_i^m \approx u(x_i, t_m)$, with $t_m = t_{m-1} + \tau$, $m = 1, ...$, and $I$ is the identity

**TABLE I**

Timings for Uniform Mesh, KdV Equation

| Problem | $h$ | $\tau$ | Time | $E_2$ | $E_\infty$ | Energy |
|---|---|---|---|---|---|---|
| Single soliton | 0.0049 | 0.01 | 5.51 | 0.393 | 0.072 | 0.0867482 |
| Single soliton | 0.0049 | 0.005 | 6.78 | 0.304 | 0.056 | 0.0867624 |
| Double soliton | 0.49 | 1.0 | 8.47 | 0.110 | 0.0222 | 0.839977 |
| Double soliton | 0.49 | 0.5 | 16.70 | .104 | 0.0210 | 0.840006 |

matrix. Equation (11) is a system of nonlinear equations which we solve for $\mathbf{v}^{m+1}$ using a Newton–Raphson method with $\mathbf{v}^m$ as a starting value.

This numerical method is used in all the subsequent tests based on the KdV equation, with or without mesh refinement. In fact, the same code is used as a basis for all of the tests.

3.1.2. *Single Soliton Problem*

This problem consists of the KdV equation with the initial condition

$$g(x) = 3c \operatorname{sech}^2(Ax + d),$$

at $t = 0$, whose theoretical solution is

$$u(x, t) = 3c \operatorname{sech}^2(Ax - Bt + d),$$

where $A = \frac{1}{2}\{bc/\varepsilon\}$ and $B = \frac{1}{2}c\sqrt{bc/\varepsilon}$.

The single soliton problem was solved on the interval $[a, b] = [0, 10]$ for $t = 0$ to $t = 25$ with $b = 1.0$, $c = 0.3$, $d = -12.0$, and $\varepsilon = 4.84 \times 10^{-4}$. The long time period is used so that any problems with any of the methods may show up if they exist. The large interval is needed because the soliton moves across the interval in that time period. The initial energy for this problem is 0.0867593. The parameters for the *soliton* method were set up as $N_{\text{gap}} = 3$ and buffer $= 2$.

As can be seen from Tables I and II, the *soliton* method compares favourably with the uniform mesh method. The accuracy of the *soliton* method is comparable to that of the uniform mesh yet the amount of time needed to compute the solution is greatly reduced. However, the uniform mesh method suffers less amount of change in the $L_2$ energy. For the smaller value of $\tau$, the difference is small in either case.

**TABLE II**

Timings for Soliton Method, KdV Equation

| Problem | $h$ | $\tau$ | Time | $E_2$ | $E_\infty$ | Energy |
|---|---|---|---|---|---|---|
| Single soliton | 0.0049 | 0.01 | 0.77 | 1.487 | 0.273 | 0.0860020 |
| Single soliton | 0.0049 | 0.005 | 0.96 | 0.208 | 0.038 | 0.0868131 |
| Double soliton | 0.49 | 1.0 | 1.84 | 0.010 | 0.0017 | 0.837283 |
| Double soliton | 0.49 | 0.5 | 3.71 | 0.037 | 0.0069 | 0.832947 |

3.1.3. *Double Soliton Problem*

A double soliton initial condition was used with the KdV equation (1) with $\varepsilon = 1$. The theoretical solution for this problem, given by Hirota [8], is

$$u = 12(\ln F)_{xx}$$

$$F = 1 + f_1 + f_2 + \left(\frac{\alpha_2 - \alpha_1}{\alpha_2 + \alpha_1}\right)^2 f_1 f_2,$$

where

$$f_i = e^{[-\alpha_i(x - x_i) + \alpha_i^3 t]}, \qquad i = 1, 2.$$

The parameters $x_1$, $x_2$, $\alpha_1$, and $\alpha_2$ represent the initial displacements and amplitudes of the two solitons. In the numerical experiments conducted, the following values were used:

$$x_1 = -1.0, \qquad \alpha_1 = 0.2$$

$$x_2 = -150.0, \qquad \alpha_2 = 0.3.$$

The soliton with the larger amplitude starts to the left of the smaller soliton. As the velocity of the solitons is directly related to their amplitudes, the larger soliton should eventually catch up and pass through the smaller one. Tables I and II again show the results of running both methods on this problem. All runs were done on the interval $[-250, 750]$ up to $t = 7000$ with $\varepsilon = 1.0$. The initial energy is 0.84.

Associated plots are shown in Figs. 2 and 3. Of special interest is Fig. 3 which shows the region of refinement for the *soliton* method for the double soliton problem. Initially, there are two regions of refinement which come together to form one region. This region eventually splits back into two regions when the faster soliton has overtaken the slower one.

3.1.4. *Non-soliton Initial Condition*

Although the *soliton* method was designed for soliton solutions, the KdV equation with a non-soliton initial condition was also tested. The initial function used was a linear hat,

$$g(x) = \begin{cases} 0, & x < x_{\text{start}} \\ \dfrac{\text{height}}{x_{\text{mid}} - x_{\text{start}}}(x - x_{\text{start}}), & x_{\text{start}} \leqslant x \leqslant x_{\text{mid}} \\ \dfrac{\text{height}}{x_{\text{mid}} - x_{\text{end}}}(x - x_{\text{end}}), & x_{\text{mid}} < x \leqslant x_{\text{end}} \\ 0, & x > x_{\text{end}} \end{cases}$$

with $x_{\text{start}} = -5$, $x_{\text{mid}} = 0$, $x_{\text{end}} = 5$, and height $= 0.9$. Equation (1) with $\varepsilon = 4.84 \times 10^4$ was solved.

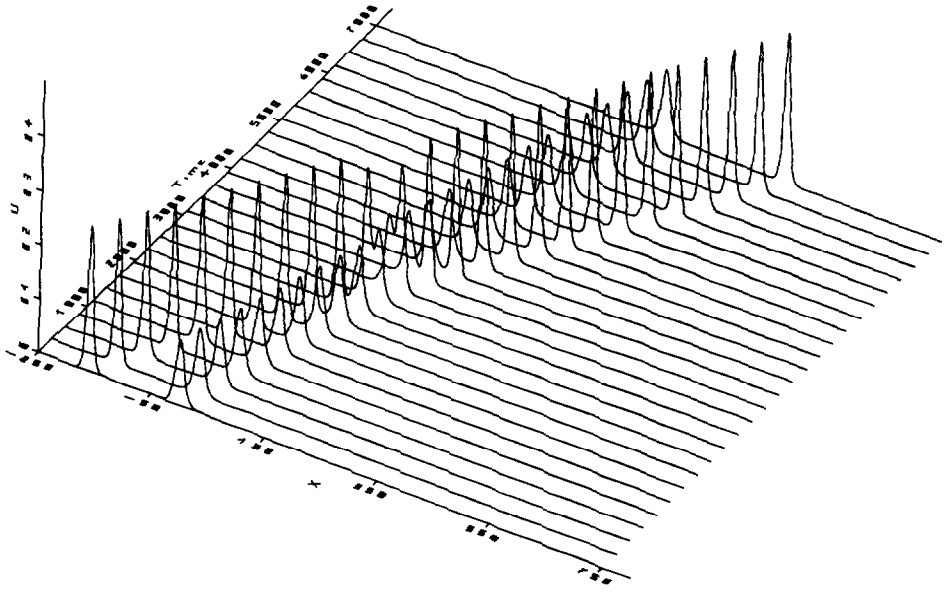

FIG. 2. Double soliton with soliton method: $h = 0.4883$, $\tau = 1.0$, buffer $= 2$, $S_{\min} = 50$, $N_{gap} = 3$, quadratic interpolation.

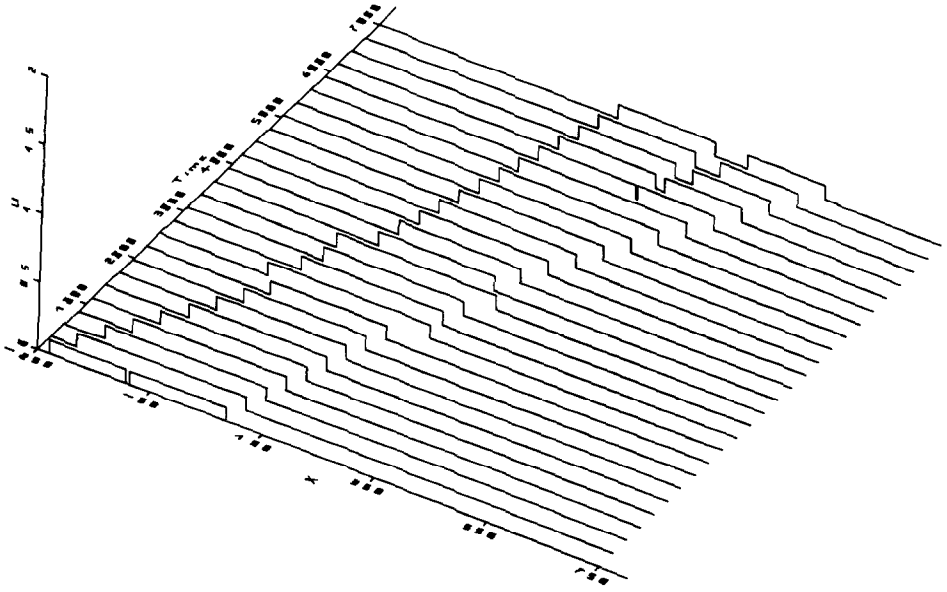

FIG. 3. Refinement areas for double soliton problem.

Figure 4 shows the result of solving this problem using the *soliton* method. As the amount of time needed to solve this problem was great (approximately 30 h of computer time), only the *soliton* method was tested. The purpose of this test was to demonstrate the fact that the *soliton* method can be used to solve problems with non-soliton initial conditions. At worst, the *soliton* method will reduce to a uniform mesh throughout the whole interval.

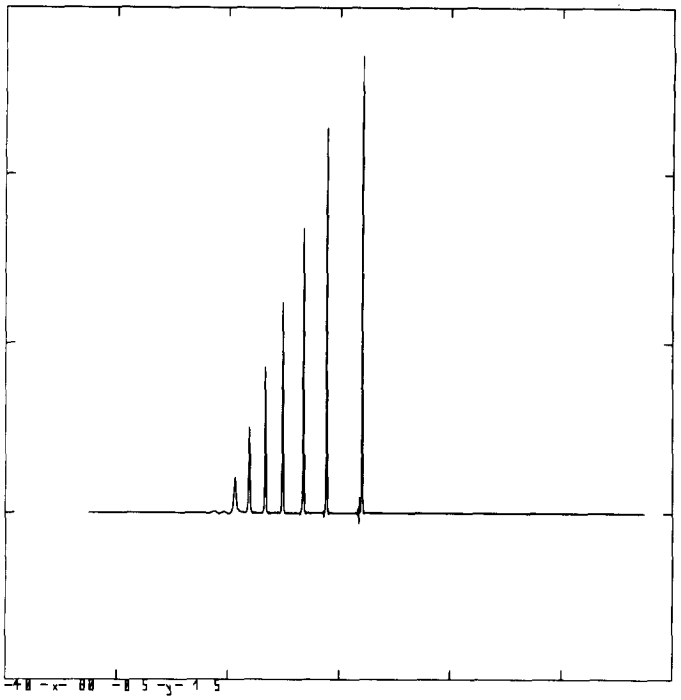

**FIG. 4.** Hat Linear initial condition with soliton method: $-40 \leqslant x \leqslant 40$, $h = 0.005$, $\tau = 0.005$, $\beta \times \text{buffer} = 1$, $S_{\min} = 50$, $N_{\text{gap}} = 300$, quadratic interpolation.

### 3.2. The Nonlinear Schrödinger Equation

#### 3.2.1. *Numerical Method*

The numerical method used to solve the NLS equation is based on a semi-discrete Galerkin method using piecewise linear basis functions and product approximation [3, 15] for the nonlinear term.

The final result of applying the above method to the NLS equation is

$$\begin{aligned} M\dot{\mathbf{u}} + S\mathbf{u} + qM\mathbf{f}(\mathbf{u}) &= 0, \\ \mathbf{u}(t) &= (\mathbf{u}_0, \ldots, \mathbf{u}_n)^{\mathrm{T}}, \\ \mathbf{u}_i &= (V_i, W_i)^{\mathrm{T}}, \end{aligned} \tag{12}$$

where $\mathbf{u}_i(t)$ is a semi-discrete approximation to $u(x_i, t)$ and $V$ and $W$ are the semi-discrete approximations to the real and complex parts, respectively, of $u$:

$$M = \frac{1}{6}\begin{bmatrix} 2h_0 I & h_0 I & & & & & \\ h_0 I & 2(h_0+h_1)I & h_1 I & & & & \\ & \ddots & \ddots & \ddots & & & \\ & & h_{i-1} I & 2(h_{i-1}+h_i)I & h_i I & & \\ & & & \ddots & \ddots & \ddots & \\ & & & h_{n-2} I & 2(h_{n-2}+h_{n-1})I & h_{n-1} I & \\ & & & & h_{n-1} I & 2h_{n-1} I \end{bmatrix},$$

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},$$

$$S = \begin{bmatrix} -\frac{1}{h_0}A & \frac{1}{h_0}A & & & & & \\ \frac{1}{h_0}A & -\left(\frac{1}{h_0}+\frac{1}{h_1}\right)A & \frac{1}{h_1}A & & & & \\ & \ddots & \ddots & \ddots & & & \\ & & \frac{1}{h_{i-1}}A & -\left(\frac{1}{h_{i-1}}+\frac{1}{h_i}\right)A & \frac{1}{h_i}A & & \\ & & & \ddots & \ddots & \ddots & \\ & & & \frac{1}{h_{n-2}}A & -\left(\frac{1}{h_{n-2}}+\frac{1}{h_{n-1}}\right)A & \frac{1}{h_{n-1}}A & \\ & & & & \frac{1}{h_{n-1}}A & -\frac{1}{h_{n-1}}A \end{bmatrix},$$

and

$$A = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}.$$

Finally,

$$\mathbf{f}(t) = (\mathbf{f}_0, ..., \mathbf{f}_n)^T,$$
$$\mathbf{f}_i = \mathbf{u}_i^T \mathbf{u}_j A \mathbf{u}_j.$$

This semi-discrete system is similar to that described by Griffiths *et al.* [6] and by Herbst *et al.* [7], but for a non-uniform mesh. In fact, this system reduces to that described by Griffiths *et al.* on a uniform mesh.

The system of ordinary differential equations (12) is solved by using a Crank–Nicholson method to yield the nonlinear system of equations

$$\left[M + \frac{\tau}{2}\left[S + qM\left(\frac{N(\alpha^{m+1}) + N(\alpha^m)}{2}\right)\right]\right]\alpha^{m+1}$$
$$= \left[M - \frac{\tau}{2}\left[S + qM\left(\frac{N(\alpha^{m+1}) + N(\alpha^m)}{2}\right)\right]\right]\alpha^m,$$

where $\alpha^m = (\alpha_0^m, ..., \alpha_n^m)^T$ with $\alpha_j^m$ as the discrete approximation to $u(x_j, m \times \tau)$ and $N(\alpha^m)$ is a diagonal matrix with entries $(\alpha_j^m)^T \alpha_j^m A$, $j = 0, ..., n$. This system is solved using a Newton–Raphson type of solver with Euler's method to generate the starting value.

### 3.2.2. *Problems*

The NLS equation (1) was solved numerically with several different initial conditions:

(a) $q = 1$, $u(x, 0) = \sqrt{2\alpha}\,\mathrm{sech}(\sqrt{\alpha}\,x)\,e^{0.5icx}$ with $\alpha = 0.5$ and $c = 1$. This initial condition yields a single soliton solution. The $L_2$ energy is 2.82843 initially.

(b) $q = 1$, $u(x, 0) = \sqrt{2\alpha}\,[\mathrm{sech}(\sqrt{\alpha}\,x_1)\,e^{0.5ic_1x_1} + \mathrm{sech}(e\alpha\,x_2)\,e^{0.5ic_2x_2}]$, $c_1 = 1.0$, $c_2 = 0.1$, $x_1 = x$, $x_2 = x - 25$. This results in a two-soliton solution. Both solitons have the same amplitude but the left one moves at a greater speed and eventually passes through the right one. Initially, the energy is 5.65685.

(c) Same initial condition as in (b) except with $c_2 = -1.0$ and $x_2 = x - 50$. In this case, the two solitons move in opposite directions with the same speed and amplitude. Initially, the energy is 5.65685.

(d) $q = 18$, $u(x, 0) = \mathrm{sech}(x)$. A value of $q = 2 * N^2$, $N = 1, ...$ generates bound states of $N$ solitons. This value of $q$ is a very stringent test of any numerical scheme (and hopefully also of any adaptive mesh scheme) due to the steep gradients that develop. Initially, the energy is 2.0.

**TABLE III**

Timings for Uniform Method, NLS Equation

| Problem | $h$ | $\tau$ | Time (m) | Energy |
|---|---|---|---|---|
| (a) | 0.195 | 0.125 | 24.573 | 2.82838 |
| (b) | 0.195 | 0.125 | 39.191 | 5.65663 |
| (c) | 0.195 | 0.125 | 37.549 | 5.65680 |
| (d), $T = 0.98$ | 0.03125 | 0.00125 | 2.385 | 2.00009 |
| (d), $T = 3.75$ | 0.0586 | 0.00125 | 7.122 | 2.00043 |
| (e) | 0.125 | 0.01 | 19.700 | 13.87910 |

(e) $q = 1$, $u(x, 0) = (1 - ix)\,e^{-0.1x^2}$. This is a non-soliton initial condition which should provide a good test for the soliton method especially. Initially, the energy is 13.87165.

### 3.2.3. *Results*

Again, both methods were run for each of the problems (a) through (e). The *soliton* method was run with buffer = 2 and $N_{\mathrm{gap}} = 3$, and with the cutoff value set to 2% ($S_{\min} = 50$) of the range of the solution values. Tables III and IV show the results for all problems.

*Single soliton.* The *soliton* method works well on this problem although a small amount of energy is lost.

*Double soliton.* The solution to problem (b) exhibits the collision of two solitons moving at different speeds. The soliton on the left at the initial time has a higher speed than the one on the right. As a result, the faster soliton eventually catches up and passes through the slower one.

Problem (c) also involves the interaction of two solitons except, in this case, the two are moving in opposite directions.

The results for these two problems are similar to those for problem (a). Again, a certain amount of energy is lost by the *soliton* method. The improvement in execution time is not as dramatic as it is for problem (a) because, in problems (b) and (c), the regions of refinement are a significant portion of the overall interval. If the problems were solved for longer time periods and, hence, larger intervals were required, the reduction in execution time for the *soliton* method would be more noticeable.

**TABLE IV**

Timings for Soliton Method, NLS Equation

| Problem | $h$ | $\tau$ | Time (m) | Energy |
|---|---|---|---|---|
| (a) | 0.195 | 0.125 | 9.897 | 2.81421 |
| (b) | 0.195 | 0.125 | 23.156 | 5.62240 |
| (c) | 0.195 | 0.125 | 24.923 | 5.62285 |
| (d), $T = 0.98$ | 0.03125 | 0.00125 | 1.117 | 1.99933 |
| (d), $T = 3.75$ | 0.0586 | 0.00125 | 2.495 | 1.95595 |
| (d), $T = 3.75$ | 0.0250 | 0.00125 | 5.184 | 1.99862 |
| (e) | 0.125 | 0.01 | 11.000 | 13.71370 |

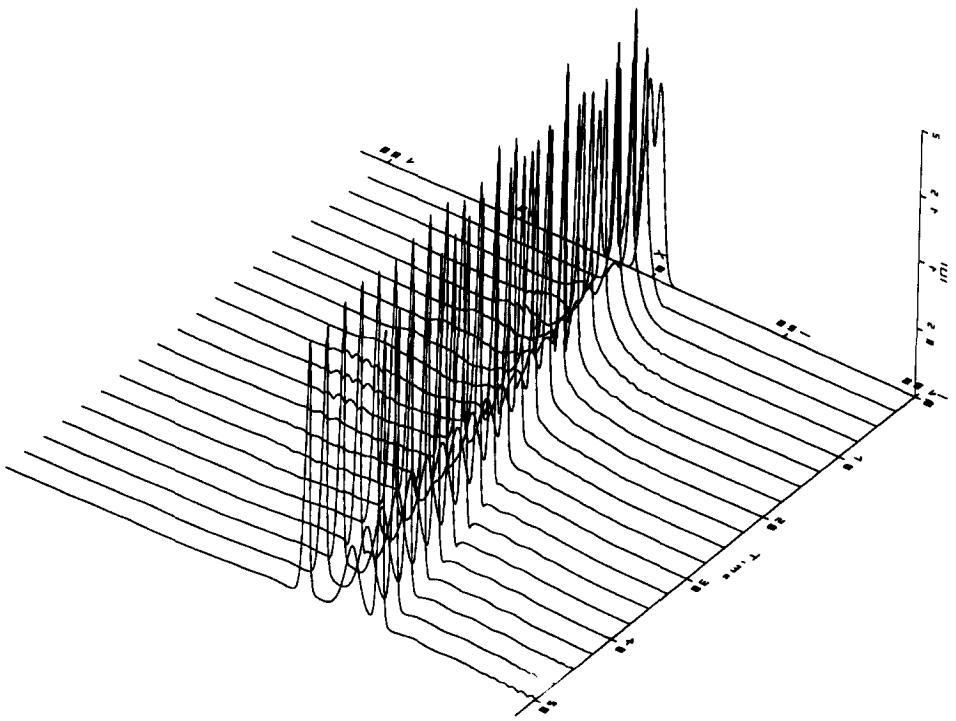

**FIG. 5.** Problem (e) with *Soliton* method: $h = 0.125$, $\tau = 0.01$, buffer $= 2$, $S_{\min} = 50$, $N_{\text{gap}} = 30$, cubic interpolation.

*Bound state of three solitons.* Problem (d) is a severe test for any numerical method and should therefore also be a good test for a spatial adaption scheme. The results show that the *soliton* method performs well. Note that this type of problem is more suited to an equi-distribution-type mesh refinement method because, although the soliton (and hence the region of refinement) does not move, the spatial derivatives change radically from time step to time step. The value of $h_{\text{goal}}$ must be small enough for the worst case. This restriction applies equally as well to the uniform mesh.

*Non-soliton initial condition.* The last experiment tried, (e), consists of a non-soliton initial condition. This experiment is carried out to again show that the *soliton* method is not restricted to soliton data. Figure 5 shows that the final result, viewed from the rear, at $T = 50$, consists of two large peaks with a smaller peak between them. The picture agrees with those shown by Griffiths *et al.* [6].

The *soliton* method is effective for this problem although it does not show such a dramatic reduction in computer time. Even though the *soliton* method was developed for soliton solutions, this experiment shows that it is not limited to such. At worst, the *soliton* method will generate a uniform mesh with only a slight overhead.

## 4. CONCLUSIONS

The soliton method, by construction, is well suited to solving problems whose solutions yield solitons. As has been shown above, the method is faster and as accurate as the uniform method. For the KdV equation, where lack of energy conservation would show up in phase errors, the *soliton* method achieves excellent results. The mesh generated by the soliton method, although nonuniform, consists of piecewise contiguous uniform sub-meshes. This amount of uniformity appears to be enough to enable the numerical method to, in practice, meet the conservation requirement to a sufficient tolerance.

The behaviour shown for the NLS equation is consistent with the behaviour shown for the KdV equation. Due to the independence of the speed and amplitude of the solitons for the NLS equation, a lack of energy conservation would not show up readily in actual plots of the solutions. Nevertheless, the results show that energy conservation is achieved to within a reasonable tolerance.

The soliton method generates accurate solutions including respecting the property of conservation of energy exhibited by the types of equations discussed. Even though the method has been developed for soliton producing equations, it is not excluded from being used on non-soliton problems, as shown by the numerical experiments.

## ACKNOWLEDGMENTS

Both authors were supported in part by the Natural Sciences and Engineering Research Council, Grant Number A3597; the second author was also supported in part by NATO Grant Number RG0358/82; and both authors also wish to acknowledge the support of the University of Dundee in providing research facilities.

## REFERENCES


1. M. J. Ablowitz and H. Segur, *Solitons and the Inverse Scattering Transform* (SIAM, Philadelphia, 1981).
2. M. J. Berger and J. Oliger, *J. Comput. Phys.* **53**, 484 (1984).
3. I. Christie, D. F. Griffiths, A. R. Mitchell, and J. M. Sanz-Serna, *IMA J. Numer. Anal.* **1**, 253 (1981).
4. J. C. Eilbeck, *Solitons and Condensed Matter Physics*, edited by A. R. Bishop and T. S. Schneider (Springer-Verlag, New York/Berlin, 1978), p. 28.
5. J. E. Flaherty and P. K. Moore, "An Adaptive Local Refinement Finite Element Method for Parabolic Partial Differential Equations," Proceedings Lisbon Conference on Numerical Analysis, 1984.
6. D. F. Griffiths, A. R. Mitchell, and J. L. Morris, *Comput. Methods Appl. Mech. Eng.* **45**, 177 (1984).
7. B. M. Herbst, J. L. Morris, and A. R. Mitchell, *J. Comput. Phys.* **60**, 282 (1985).
8. R. Hirota, *Phys. Rev. Lett.* **27**, 1192 (1971).
9. V. S. Manoranjan, *An Adaptive Scheme in One Space Dimension* (University of Dundee, Dept. of Math Report NA/76, 1984).
10. A. R. Mitchell and S. W. Schoombie, *Numerical Methods for Coupled Problems*, edited by E. Hinton, P. Betters and R. W. Lewis (Pineridge, Swansea, UK, 1981), p. 3.
11. J. S. Russell, in *14th Meeting of the British Assoc. for the Advancement of Science, London*, 1844.
12. R. D. Russell and J. Christiansen, *SIAM J. Numer. Anal.* **15**, 59 (1980).
13. J. M. Sanz-Serna and I. Christie, *J. Comput. Phys.* **67**, 348 (1986).
14. J. F. Thompson, *Appl. Numer. Math.* **1**, 3 (1985).
15. Y. Tourigny, *Product Approximation for Two Nonlinear Klein–Gordon Equations* (University of Dundee, Dept. of Math Report NA/99, 1987).
16. A. B. White, *SIAM J. Numer. Anal.* **16**, 472 (1979).
17. N. J. Zabusky and M. D. Kruskal, *Phys. Rev. Lett.* **15**, 240 (1965).