JOURNAL OF COMPUTATIONAL PHYSICS **128,** 274–288 (1996)
ARTICLE NO. 0210

Some Observations on a Static Spatial Remeshing Method Based on Equidistribution Principles

P. SAUCEZ

Laboratoire de Mathématique et Recherche Opérationnelle, Faculté Polytechnique de Mons, 7000 Mons, Belgium

A. VANDE WOUWER[1]

Laboratoire d'Automatique, Faculté Polytechnique de Mons, 7000 Mons, Belgium

AND

W. E. SCHIESSER

Department of Computer Science and Engineering, Lehigh University, Bethlehem, Pennsylvania 18015

Received April 21, 1995; revised May 14, 1996

This paper proposes a method of lines solution procedure with time and space adaptation for one-dimensional systems of partial differential equations whose solutions display steep moving fronts. The spatial remeshing algorithm, which is a variation of the method published by Sanz-Serna and Christie and an extension suggested by Revilla, is a static remeshing method based on equidistribution principles. The selection of the several algorithm components, i.e., grid placement criterion, spatial discretization scheme, time integrator, adaptation frequency, and parameter tuning, are investigated and illustrated with several test examples, i.e., the cubic Schrödinger equation, a model of a single-step reaction with diffusion, and a model of flame propagation. © 1996 Academic Press, Inc.

1. INTRODUCTION

The numerical study of one-dimensional systems of evolutionary partial differential equations (PDEs) whose solutions display steep moving fronts has demonstrated the need for numerical solution procedures with time and space adaptation. In recent years, the interest in spatial remeshing techniques has increased enormously (see, e.g., [2, 5, 8]). These techniques use nonuniform grids and, as time evolves, automatically concentrate the grid points in the spatial regions where the solution is rapidly changing. Among the large variety of spatial remeshing techniques, two major approaches can be distinguished depending on the way—continuously in the time domain or only at discrete time levels—the grid moves.

In this study, we consider the discrete time level (or static) approach, and we investigate a class of methods based on equidistribution principles as originally introduced by White [23]. Previous contributions in this area include Sanz-Serna and Christie [20] and Revilla [16] who study the numerical solution of the cubic Schrödinger equation (CSE) in one spatial dimension. Their experiments demonstrate clearly the benefits of the adaptation in space, resulting in a reduction of the number of nodes by a factor 4 to 12 (depending on the example considered and on the choice of the function to be equidistributed) in comparison to that required by similar fixed-grid methods.

Starting with the spatial remeshing technique proposed by those authors, we explore several important issues including the selection of a grid placement criterion, the choice of a spatial discretization scheme, the time integration, the grid adaptation frequency, and the tuning of the parameters used in the algorithm. With regard to these points, the results of a large number of numerical experiments are discussed, and several departures from the conclusions of Sanz-Serna and Christie [20] and Revilla [16] are highlighted. In particular, our experiments show that the selection of a spatial discretization scheme has a significant influence on the quality of the numerical solution, and a discretization employing higher-order finite differences allows the number of nodes to be further reduced by a factor up to two and reduces the computational costs significantly.

As a result of these investigations, a numerical scheme is proposed which makes use of the implicit RK solver RADAU5 [7] for the integration in time and of a newly developed spatial remeshing routine called AGE (*a*daptive *g*ridding through *e*quidistribution). The complete Fortran programs are available on request from the authors.

This paper is organized as follows. Section 2 introduces

[1] Author to whom correspondence should be addressed. E-mail: vdw@autom.fpms.ac.be.

0021-9991/96 $18.00

the MOL solution procedure along with the grid adaptation mechanism. Section 3 deals with the spatial remeshing technique and describes the structure of the algorithm. In Section 4, several spatial discretization schemes, including cubic spline differentiators and finite differences, are considered and their relative merits are highlighted. Section 5 discusses several aspects related to the time integration, e.g., initialization procedure and grid adaptation frequency, and compares two standard ODE solvers, e.g., LSODI [10] and RADAU5 [7]. In Section 6, the solution procedure is applied to a variety of soliton solutions of the CSE, i.e., the propagation of a single soliton, the interaction between two solitons, and the bound state of several solitons. Performance of the solution procedure is also illustrated with examples of another nature, i.e., a model of a single-step reaction with diffusion and a model of flame propagation.

2. THE BASIC SOLUTION PROCEDURE

The MOL solution of an initial-boundary value problem using spatial remeshing is split into four separate tasks:

(1) the spatial discretization of the PDEs using finite difference or finite element approximations on a fixed nonuniform grid,

(2) the time integration of the resulting system of stiff ODEs using an established solver,

(3) the adaptation of the spatial grid,

(4) the interpolation of the solution to produce new initial conditions.

In the following sections, we focus attention on steps (1)–(4) and, based on the grid adaptation algorithm developed by Sanz-Serna and Christie [20] and Revilla [16], we propose several further extensions and improvements. An essential feature of this procedure is that the grid adapts only at discrete time levels and no coupling exists between the computation of the PDE solution and the grid adaptation. Thus, the grid adaptation algorithm is problem independent and can be coded once and for all. The discrete time level (or static) approach contrasts with continuously moving grid (or dynamic) techniques, a well known example of which is the moving finite element method introduced by Miller and Miller [13, 14]. In this second approach, the nodes move according to ODEs which are coupled to the problem equations. While these techniques have proven very successful in the solution of problems displaying very steep gradients, algorithms which move the grid continuously in the space-time domain are more complicated to derive and result in larger, usually ill-conditioned, sets of nonlinear ODEs to be integrated in time. Uncoupling the computations of the PDE solution and the grid is the major advantage of the discrete time level approach.

3. THE SPATIAL REMESHING ALGORITHM

Sanz-Serna and Christie [20] and Revilla [16] have developed an efficient method for moving the nodes which is based on the equidistribution of some appropriate functional, e.g., the arc-length of the solution. The computation of the nonuniform grid x_i^k, $i = 1, ..., N$, occurs at discrete time levels t_k and between two time levels t_k and $t_{k+1} = t_k + \Delta t$; the solution is advanced on this fixed grid by a suitable space-time discretization of the PDEs (steps 1, 2 defined in Section 2).

The algorithm proposed by Sanz-Serna and Christie [20] is based on the equidistribution of the arc-length of the solution, i.e., at the time level t_{k+1}, the nodes x_i^{k+1}, $i = 1, ..., N$, are located such that

$$\int_{x_{i-1}^{k+1}}^{x_i^{k+1}} \left(1 + \left\| \frac{\partial u(x, t_{k+1})}{\partial x} \right\|_2^2 \right)^{1/2} dx \approx \text{constant}, \qquad (3.1)$$

where $u(x, t_{k+1})$ is the PDE solution which has been advanced to the time t_{k+1} using the fixed grid x_i^k, $i = 1, ..., N$. In fact, the criterion is not used in this form, but with a positive parameter α which can be used to modify the relative importance of values of x and values of u, i.e.,

$$\int_{x_{i-1}^{k+1}}^{x_i^{k+1}} \left(\alpha + \left\| \frac{\partial u(x, t_{k+1})}{\partial x} \right\|_2^2 \right)^{1/2} dx \approx \text{constant}. \qquad (3.2)$$

In his paper [16], Revilla points out that this choice of the equidistributed function leads to the location of an excessive number of nodes in regions where the solution is linear and proposes another equidistribution principle based on the second derivative of the solution instead of the first derivative, i.e.,

$$\int_{x_{i-1}^{k+1}}^{x_i^{k+1}} \left(\alpha + \left\| \frac{\partial^2 u(x, t_{k+1})}{\partial x^2} \right\|_2 \right)^{1/2} dx \approx \text{constant} \qquad (3.3)$$

which results in a further improvement of the scheme.

An additional parameter β is introduced to avoid the excessive clustering of nodes in regions where $\|\partial^2 u/\partial x^2\|_2$ is large, i.e., the values of the second derivatives which exceed β are reduced to the value β.

In this study, the grid adaptation algorithm has been implemented and tested using both forms of the equidistribution principle. As pointed out by Revilla, we have also observed that the use of the criterion (3.3) leads to a better grid distribution and allows the number of nodes to be further reduced. Following this observation and the idea of Dwyer *et al.* in an earlier study [3], who suggested using both first- and second-order derivatives in order to define

the grid placement criterion, we have selected two criteria in the forms

$$\int_{x_{i-1}^{k+1}}^{x_i^{k+1}} \left(\alpha + \left\| \frac{\partial^2 u(x, t_{k+1})}{\partial x^2} \right\|_\infty \right)^{1/2} dx \approx \text{constant} \tag{3.4a}$$

$$\int_{x_{i-1}^{k+1}}^{x_i^{k+1}} \left(\alpha + \left\| \frac{\partial u(x, t_{k+1})}{\partial x} \right\|_\infty^2 + \left\| \frac{\partial^2 u(x, t_{k+1})}{\partial x^2} \right\|_\infty \right)^{1/2} dx \approx \text{constant}. \tag{3.4b}$$

The criterion (3.4a) is the same as (3.3) except that, by using the ∞ norm instead of the 2 norm, the nodes now move according to the value of the largest component of the derivative vectors, i.e., they follow the solution component which is more rapidly changing. The idea behind the criterion (3.4b) is to weight the flat part and the slope in a solution differently, so that some nodes are drawn from the curvature to the slope. In this case, a parameter β, which now limits the effect of both first and second derivatives, is used to avoid an excessive grid distortion. In the majority of test problems considered, the numerical scheme based on (3.4b) seems to be slightly less expensive but also slightly less accurate than the one based on (3.4a). Anyway, the differences observed when using (3.4b) rather than (3.4a) are marginal when compared to the benefits obtained when using (3.4a) rather than (3.2), which shows that the second order derivative is the most important information to be used in a grid placement criterion, and that, in fact, the slope in a solution does not require more nodes than the flat part. The influence of the grid placement criterion and the tuning parameters (i.e., the number of nodes N, the scaling factor α, and the limiting factor β) on the performance of the algorithm will be illustrated with several test examples in Section 6. The spatial remeshing algorithm has been coded in portable Fortran in subroutine AGE and is available on request from the authors.

4. THE SPATIAL DISCRETIZATION SCHEMES

Numerical approximations of spatial derivatives are necessary for transforming the PDEs into a set of ODEs (step 1) and for computing the equidistribution principle which involves the first- and/or second-order derivatives of the solution (step 3). In their work, Sanz-Serna and Christie [20] and Revilla [16] make use of a three-point finite difference scheme for spatial discretization. They also experiment with a piecewise linear Galerkin method and obtain similar performance. Hence, they conclude that finite differences and finite elements on the adaptive grid provide the same accuracy and that, when the grid adapts to the solution, most algorithms work well. However, this conclusion seems a bit premature, and our numerical experiments show that the selection of a discretization scheme does have a significant influence on the performance of the solution procedure.

In this work, several spatial approximations have been considered:

(a) cubic spline differentiators for first- and second-order derivatives on a nonuniform grid as implemented in subroutine NCSPLE of the DSS/2 MOL library developed by Schiesser [21],

(b) finite difference approximations for derivatives up to any order and up to any level of accuracy on a nonuniform grid as implemented in the subroutine Weights by Fornberg [4].

Cubic spline differentiators for second-order derivatives have the same level of accuracy as three-point finite differences and lead to similar performance. On the other hand, the use of an approximation scheme with a higher level of accuracy, i.e., method (b), allows the number of nodes required for the spatial approximation of the CSE to be further reduced by a factor up to two as compared to the numerical results reported by Revilla. Actually, the selection of a spatial differentiator seems more important at step 1 of the solution procedure, i.e., the spatial approximation of the PDEs, than for computing the derivatives involved in the grid adaptation criterion (step 3). This observation is logical since, as mentioned by Sanz-Serna and Christie, the grid point positions and the solution values do not play a symmetric role. It is sufficient to move the nodes in a way which allows for a satisfactory spatial discretization of the PDEs, and not necessary to obtain very accurately any prescribed set of nodes. Accordingly, cubic spline differentiators, as implemented in NCSPLE, have been used in the grid adaptation subroutine AGE. On the other hand, a fourth-order accurate scheme for approximating the spatial derivatives in the CSE seems appropriate and yields better algorithm performance. The use of the several spatial approximations is illustrated in Section 6 wherein several test examples are considered.

5. THE TIME INTEGRATION

The spatial discretization of an initial-boundary value problem results in a set of stiff ODEs which must be integrated in time (step 2). In the space/time adaptive algorithm of Sanz-Serna and Christie, an implicit midpoint rule is used for the integration in time and a new spatial grid is computed at each variable time step taken by the time integrator. In our approach, the time integration is halted and the spatial grid is updated periodically, either at specified discrete time levels $t_k = k\ \Delta t$ or after a fixed number *n*steps of integration steps. This procedure has been adopted because it is not required to adapt the spatial grid at each time step in order to obtain good accuracy and, in

fact, larger grid adaptation periods can be considered. Thus, in the implementation described in this paper, the grid adaptation frequency is a tuning parameter of the algorithm which can be chosen so that the grid adapts well without an excessive amount of computation time spent on computing the spatial grid and interpolating the solution. Both adaptation mechanisms, i.e., grid adaptation at specified time levels t_k or after a fixed number of integration steps, have been used and the second method has finally been selected. As discussed by Adjerid and Flaherty [1], updating the grid after a fixed number of integration steps rather than at fixed times reduces the chance of having a rapid transient suddenly appear and disappear between two grid adaptation times, which could result in inaccurate solutions being accepted as correct. The time steps taken by the ODE solver will become smaller when the solution is changing more rapidly, and the grid will adapt more frequently. Moreover, we have observed that when the grid is updated at specified time levels $t_k = k\,\Delta t$, there is an interplay between the number of nodes and the adaptation period Δt, which makes the tuning of the adaptive algorithm more difficult. Actually, as Δt does not vary with the speed of the solution, the number of nodes must be larger than strictly needed to represent the solution. Otherwise the adaptation period Δt should be chosen smaller and the method overall becomes less efficient.

In this study, the initial value problem resulting from spatial discretization is handled as a system of differential algebraic equations (DAEs), i.e., ODEs come from the spatial approximation of the PDEs, and AEs may result from the approximation of the boundary conditions. Two standard solvers have been used in combination with our spatial remeshing technique, and their performances have been compared:

(a) the variable-step, variable-order, backward differentiation formulas (BDF) solver LSODI (Hindmarsh [10]), and

(b) the variable-step, fifth-order, implicit Runge–Kutta (RK) solver RADAU5 (Hairer and Wanner [7]).

The space/time adaptive algorithm proceeds as follows:

Initially, the nodes x_i^0, $i = 1, ..., N$, are distributed uniformly in the spatial domain. Then a call to subroutine AGE is made in order to adapt the grid according to the initial condition $u_0(x)$. Actually, computation of this initial nonuniform grid is required to successfully start the time integration, i.e., LSODI or RADAU5 usually fail to converge when the integration is started using the uniform grid which contains a purposely small number of nodes. This initialization procedure, which is also suggested by Sanz-Serna and Christie [20], does not need to be iterated as opposed to the comments of those authors, i.e., in all the examples considered, a satisfactory distribution of the nodes is obtained after a single call to subroutine AGE so it is not necessary to iterate the procedure.

After every *n*steps integration steps, a nonuniform grid is computed (step 3), and the solution is interpolated (step 4) using cubic splines as implemented in subroutine NCSPLE [21] in order to generate new initial conditions for the next time interval.

When using a variable-order BDF method as implemented in LSODI, the integration must be restarted with a first-order method, i.e., the parameter ISTATE of LSODI is set equal to 1 to reset the BDF formulas to first order, so that no past history of the dependent variables is required. However, the implementation of the space/time adaptive algorithm in this manner decreases the efficiency of LSODI because there are many restarts and the solver uses low-order BDF most of the time. In fact, the performance of LSODI will greatly depend on the frequency of grid updating, which in turn depends on the problem to be solved and the accuracy requirements. When the problem is not very demanding, the number of integration steps after which a new grid is computed can be relatively large so that LSODI can increase the order and perform satisfactorily. In general, however, LSODI is not an optimal choice of solver for this type of application and an implicit RK method as implemented in RADAU5 is more appropriate.

A pseudo-Fortran description of the MOL solution procedure is given in Fig. 1 and statistics for both solvers are illustrated in the next section.

6. NUMERICAL EXPERIMENTS

The numerical techniques described in the previous sections are now applied to a variety of soliton solutions of the CSE, i.e., the propagation of a single soliton, the interaction between two solitons, and the bound state of several solitons. Performance of the solution procedure is also illustrated with examples of another nature, i.e., a model of a single-step reaction with diffusion and a model of flame propagation.

6.1. *Cubic Schrödinger Equation*

During recent years, a great deal of interest has developed in the numerical treatment of PDEs giving rise to solitary waves (or solitons). In this paper, we consider an important particular case, namely, the CSE in one spatial dimension given by

$$\begin{aligned} i\frac{\partial u}{\partial t} + \frac{\partial^2 u}{\partial x^2} + q|u^2|u = 0, \qquad -\infty < x < \infty \\ u(x,0) = u_0(x), \end{aligned} \tag{6.1}$$

where $u(x, t)$ is a complex-valued function of space and

```
Program

      compute the initial conditions and an initial uniform grid;
      compute a nonuniform grid according to criterion (3.4);
      k=0;
      tout=tinitial+tprint;

1     continue

      if (t.lt.tout) then

            approximate the spatial derivatives in the PDE problem;
            integrate the resulting DAE system for one time step;
            k=k+1;

            if (k.eq.nsteps) then

                  update the grid according to criterion (3.4);
                  interpolate the solution;
                  reinitialize the time integrator;
                  k=0;

            end if

      else

            print the solution;

      end if

      if (t.ge.tfinal) stop
      tout=tout+tprint;
      go to 1

end
```

FIG. 1. Pseudo-Fortran description of the MOL solution procedure.

time and q is a real parameter. This equation has been used extensively to model nonlinear dispersive waves (see Strauss [22] and Whithman [24]).

For the numerical treatment of the CSE (see, e.g., [6, 9, 16, 17, 18, 19, 20]), we assume that, for the time interval $0 \le t \le T$ under consideration, the solution vanishes outside some interval (x_L, x_R), and we introduce the artificial Dirichlet boundary conditions

$$u(x_L, t) = u(x_R, t) = 0 \tag{6.2}$$

which is a reasonable approximation for the solutions we are interested in. Alternatively, Neuman boundary conditions ($u_x = 0$, $x = x_L$ and $x = x_R$) could be considered.

Furthermore, the complex-valued function $u(x, t)$ is decomposed into its real and imaginary parts

$$u(x, t) = v(x, t) + iw(x, t) \tag{6.3}$$

so that (6.1), (6.2) lead to a set of two PDEs with initial and boundary conditions for the real functions $v(x, t)$ and $w(x, t)$,

$$\frac{\partial v}{\partial t} + \frac{\partial^2 w}{\partial x^2} + q(v^2 + w^2)w = 0 \tag{6.4}$$

$$\frac{\partial w}{\partial t} - \frac{\partial^2 v}{\partial x^2} - q(v^2 + w^2)v = 0 \tag{6.5}$$

$$v(x, 0) = u_{0R}(x) \qquad w(x, 0) = u_{0I}(x) \tag{6.6}$$

$$v(x_L, t) = v(x_R, t) = 0 \qquad w(x_L, t) = w(x_R, t) = 0 \tag{6.7}$$

where $u_0(x) = u_{0R}(x) + iu_{0I}(x)$.

6.1.1. *Propagation of a single soliton.* The initial condition u_0 is given by

$$u_0(x) = \sqrt{2a/q}\, \exp[i0.5s(x - x_0)]\, \mathrm{sech}[\sqrt{a}(x - x_0)] \tag{6.8}$$

and the corresponding soliton solution is

$$\begin{aligned} u(x, t) = \sqrt{2a/q}\, \exp[i(&0.5s(x - x_0) \\ &- (0.25s^2 - a)t)]\, \mathrm{sech}[\sqrt{a}((x - x_0) - st)] \end{aligned} \tag{6.9}$$

The modulus $|u(x, t)|$ represents a wave initially centered at $x = x_0$, whose amplitude $\sqrt{2a/q}$ is determined by the real parameter a, and which propagates with speed s in the positive direction of x. Note that the soliton travels without change of shape.

As in [16, 19, 20] the problem is solved for $q = 1$, $a = 1$, $s = 1$, and $x_0 = 0$. The artificial boundaries are located at $x_L = -30$, $x_R = 70$, and the time interval of interest is $(0, 30)$.

In order to assess the influence of the grid placement criterion, of the choice of a spatial discretization scheme, and of the parameter tuning of the algorithm on the performance of the MOL solution procedure, the problem is solved using several variations of the adaptive algorithm. Version 1 uses the grid placement criterion (3.4a) based on the second-order derivative. The spatial derivatives in the PDEs (6.4), (6.5) are approximated by three-point central finite differences. Version 2 uses the same grid placement criterion but approximates the spatial derivatives using five-point central finite differences. Version 3 uses the grid placement criterion (3.4b), which combines first- and second-order derivatives, and approximates the spatial derivatives by five-point finite differences. In all three cases, the solution is computed with $\alpha = 10^{-4}$ and $\beta = 100$ (in fact, for this example, numerical experiments show that the limitor β has no influence on the results as long as $\beta >$ 10, i.e., as long as β is larger than the maximum computed value of $\|\partial^2 u/\partial x^2\|$). The time integration is performed using RADAU5 with relative and absolute error tolerances set to 10^{-7} and a frequency of grid adaptation $n$steps = 10. Version 4 is the same as version 2 except the time integration is performed using LSODI with error tolerances set to 10^{-7} and a frequency of adaptation $n$steps = 10. Tight

TABLE I

Propagation of a Single Soliton -2 Norm of the Error $\times 10^2$

Version	N	$t = 5$	$t = 10$	$t = 15$	$t = 20$	$t = 25$	$t = 30$
1	51	0.681	1.19	0.568	2.52	3.27	4.07
1	151	0.0513	0.106	0.165	0.227	0.295	0.369
2	31	1.13	1.18	1.69	2.23	2.99	4.29
2	41	0.356	0.497	0.515	0.517	0.672	0.740
2	51	0.227	0.195	0.134	0.352	0.151	0.346
3	31	1.82	2.18	2.13	2.22	3.06	3.87
3	41	0.612	0.765	0.794	0.898	1.24	1.16
3	51	0.282	0.359	0.486	0.465	0.534	0.535
4	51	0.132	0.202	0.291	0.318	0.385	0.392

tolerances are used for the time integration so that the spatial errors dominate. The results of the computation using the four different versions are presented in Tables I and II. Results in Table I illustrate the evolution of the 2 norm of the error in the numerical solution, as compared to the theoretical solution (6.9), given by

$$\|\text{error}\|_2 = \left(\left(\frac{1}{(x_R - x_L)} \sum_{i=1}^{N-1} \frac{(x_{i+1} - x_i)}{2} \right.\right. \\ \left.\left. (\text{error}(x_i)^2 + \text{error}(x_{i+1})^2)\right)^{1/2}\right. \tag{6.10}$$

and computed at several output times $t = 5, 10, 15, 20, 25, 30$.

Table II presents some computational statistics: N is the number of nodes, STEPS is the number of time steps needed to complete the solution (RADAU5, IWORK(16); LSODI, IWORK(11)), FNS is the number of function evaluations (RADAU5, IWORK(14); LSODI, IWORK(12)), JACS is the number of Jacobian evaluations (RADAU5, IWORK(15); LSODI, IWORK(13)), and CPU is the computation time including some I/O costs, which is given for information only. All computations were done on a Pentium 133.

TABLE II

Propagation of a Single Soliton—Computational Statistics

Version	N	STEPS	FNS	JACS	CPU (s)
1	51	3056	16759	332	171
1	151	3499	18451	349	1871
2	31	1893	10867	441	62
2	41	2030	11655	402	100
2	51	1733	10231	367	125
3	31	1807	10554	424	56
3	41	1741	10254	380	85
3	51	1436	8761	357	117
4	51	9400	366825	3435	357

It is apparent from Table I that the solution obtained with version 1 is less accurate than the one obtained with version 2, i.e., the selection of a spatial discretization scheme significantly influences the accuracy of the numerical solution. To obtain similar accuracy with version 1, it is necessary to use at least 151 nodes, which significantly increases the computational load. Version 2 and version 3 have been used with different number of nodes, e.g., $N = 31, 41, 51$, in order to illustrate the convergence properties of the method. These results illustrate that the grid placement criteria (3.4a), (3.4b) give similar performances. Depending on the number of nodes, one or the other of the solutions can be slightly more accurate or less expensive. Finally, the use of version 4 shows that LSODI does not perform as well as RADAU5 on this problem, a predictable conclusion in view of the large number of restarts which reduce the efficiency of the multistep method.

The known analytical solution (6.9) and the numerical results obtained with version 2 and $N = 51$ are compared in Fig. 2, where the solution is graphed every 5 units in t. The location of the adaptive nodes is indicated at the bottom of the figure. The grid adapts well, and the computed solution follows almost exactly the theoretical solution.

Finally, Table III presents some computational statistics corresponding to the case of the soliton moving through the spatial domain at different speeds, e.g., $s = 1, 2, 3, 4$. The time interval of interest (0, TF) is inversely proportional to the speed of the soliton, so that the final location of the soliton is $x = 30$ in all cases. Table III gives the evolution of the 2 norm of the error in the numerical solution in this final location. These results, which are computed with versions 2 and 3 ($N = 51$), show that the grid adaptation frequency increases with the speed of the soliton, and so the number of grid updates only slightly decreases with version 2 and remains more or less the same with version 3. However, we observe in both cases that the accuracy level of the numerical solution slightly decreases as the speed of the soliton is increased.

This example illustrates the advantage of a procedure based on a grid adaptation after a fixed number of integration steps as compared to an adaptation at specified time levels $t_k = k\,\Delta t$. In this latter case, the tuning parameter Δt must be reduced proportionally to the speed of the soliton, which requires a priori knowledge about the solution.

6.1.2. *Interaction between two solitons.* The parameters a and s are independent and so two initial wave profiles can propagate with different amplitudes and speeds. Thus, we consider an initial condition which is the superposition of two solitons with amplitudes a_1 and a_2, respectively, centered at x_{01} and x_{02}, and traveling with different speeds s_1 and s_2:

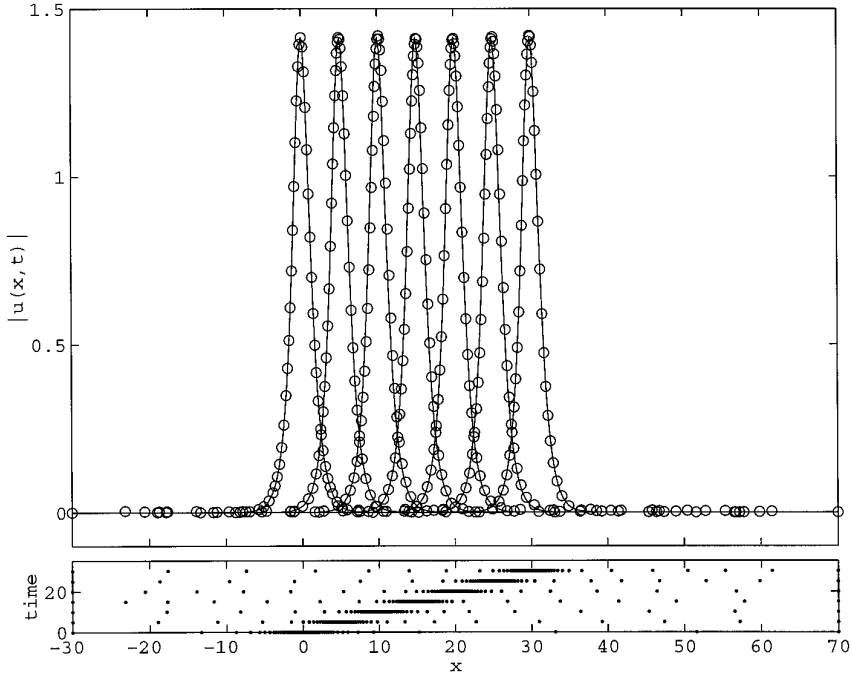

FIG. 2. Propagation of a single soliton—Analytical solution (solid line) and numerical solution (circled points) at $t = 0, 5, ..., 30$; five-point finite differences; equidistribution principle (3.4a) with $N = 51$, $\alpha = 10^{-4}$, $\beta = 100$, $n$steps $= 10$.

$$u_0(x) = \sqrt{2a_1/q}\,\exp[0.5is_1(x - x_{01})]\,\mathrm{sech}[\sqrt{a_1}(x - x_{01})] + \sqrt{2a_2/q}\,\exp[0.5is_2(x - x_{02})]\,\mathrm{sech}[\sqrt{a_2}(x - x_{02})]. \tag{6.11}$$

As time evolves, the faster soliton catches the slower one and passes through it. A remarkable property is that the shape and velocity of both solitons are unaltered after the interaction; the only effect is a phase shift.

As a further illustration, we consider this problem with $q = 1$, $a_1 = 0.2$, $a_2 = 0.5$, $s_1 = 1$, $s_2 = 0.1$, $x_{01} = 0$, and $x_{02} = 25$. The artificial boundaries are located at $x_L = -20$, $x_R = 80$, and the time span of interest is (0,45). The CSE is solved using versions 1–3 of the algorithm described in the previous section with $N = 51$, $\alpha = 10^{-4}$, $\beta = 100$, and $n$steps $= 10$. Table IV presents some computational statistics. Figures 3 through 5 show the results of the evolution of the two interacting solitons at $t = 10$, 25, and 45, respectively, which have been obtained with version 2. The results at $t = 25$ obtained with version 1 are graphed in Fig. 6, which clearly shows that the selection of a spatial discretization scheme significantly influences the accuracy of the numerical solution. In his paper [16], Revilla discusses also the problem of the interaction between two solitons and found that 101 nodes are necessary. Note that, on a fixed uniform grid, 401 nodes are required [19].

In addition, Fig. 7 illustrates the evolution of the stepsize taken by RADAU5. It is apparent that the stepsize, and thus the adaptation period, is automatically reduced when the two solitons interact.

TABLE III

Propagation of a Single Soliton—Computational Statistics and 2 Norm of the Error $\times 10^2$

Version	s	STEPS	FNS	JACS	error at $x = 30$
2	1	1733	10231	367	0.346
2	2	1362	7976	253	1.45
2	3	1298	7518	227	2.73
2	4	1241	7189	190	4.51
3	1	1436	8761	357	0.535
3	2	1445	8549	266	1.53
3	3	1497	8603	211	4.27
3	4	1544	8850	166	5.46

6.3.3. *Bound state of several solitons.* Miles [12] has

TABLE IV

Interaction of Two Solitons—Computational Statistics

Version	STEPS	FNS	JACS	CPU (s)
1	2303	13447	322	164
2	1549	8766	324	137
3	1082	6325	296	105

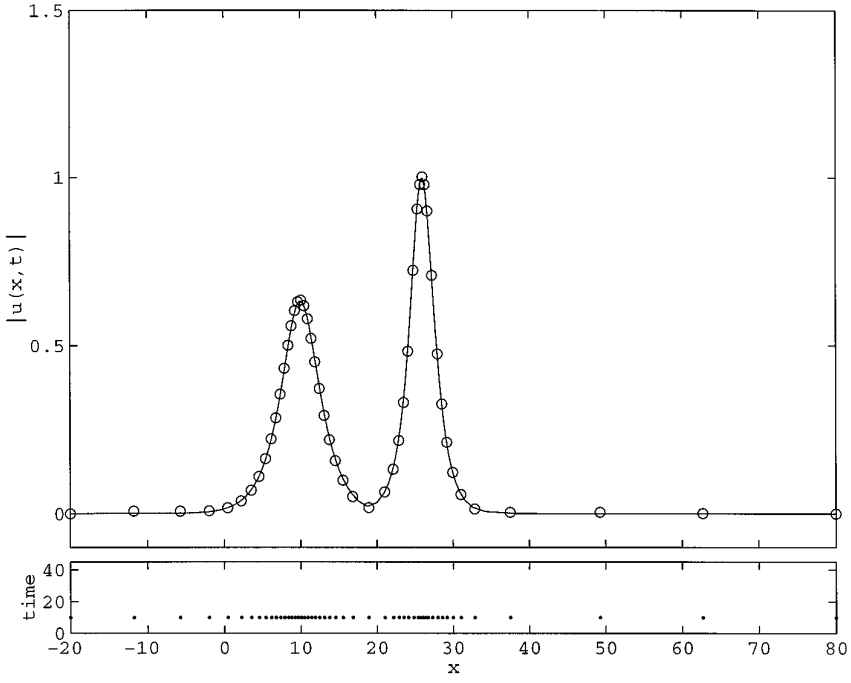

FIG. 3. Interaction between two solitons at $t = 10$—five-point finite differences; equidistribution principle (3.4a) with $N = 51$, $\alpha = 10^{-4}$, $\beta = 100$, *n*steps $= 10$.

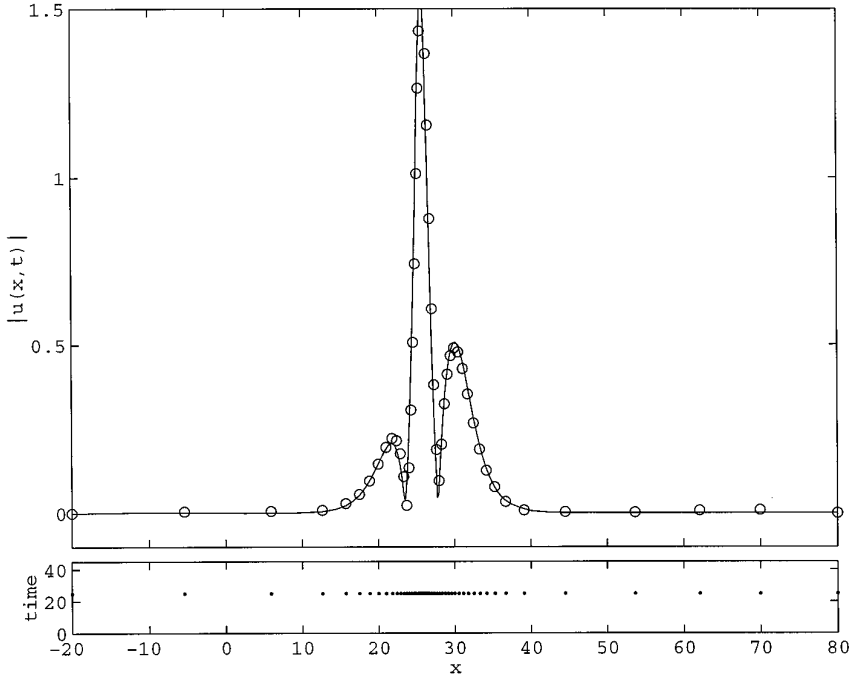

FIG. 4. Interaction between two solitons at $t = 25$—five-point finite differences; equidistribution principle (3.4a) with $N = 51$, $\alpha = 10^{-4}$, $\beta = 100$, *n*steps $= 10$.

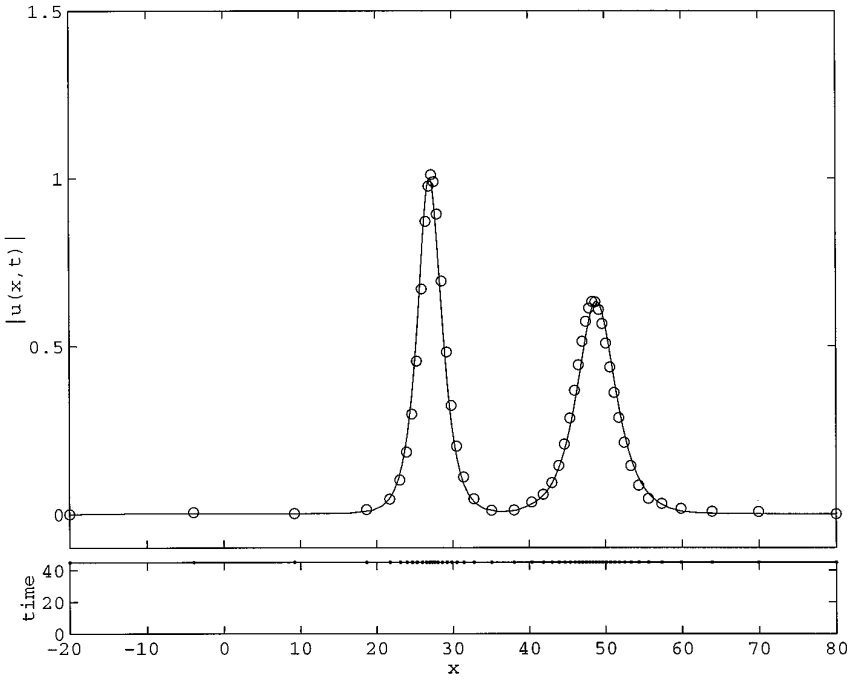

FIG. 5. Interaction between two solitons at $t = 45$—five-point finite differences; equidistribution principle (3.4a) with $N = 51$, $\alpha = 10^{-4}$, $\beta = 100$, *n*steps $= 10$.

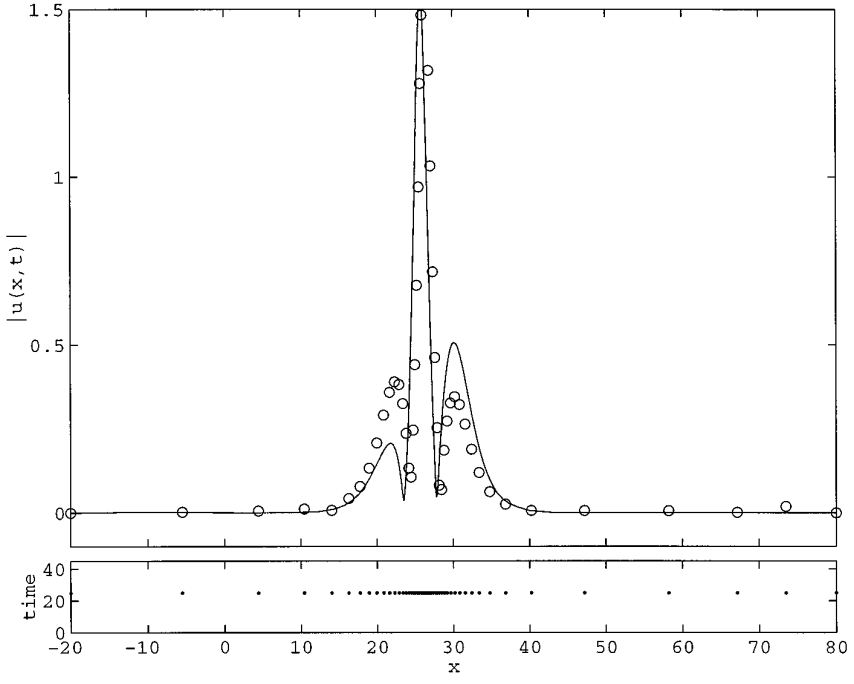

FIG. 6. Interaction between two solitons at $t = 25$—three-point finite differences; equidistribution principle (3.4a) with $N = 51$, $\alpha = 10^{-4}$, $\beta = 100$, *n*steps $= 10$.

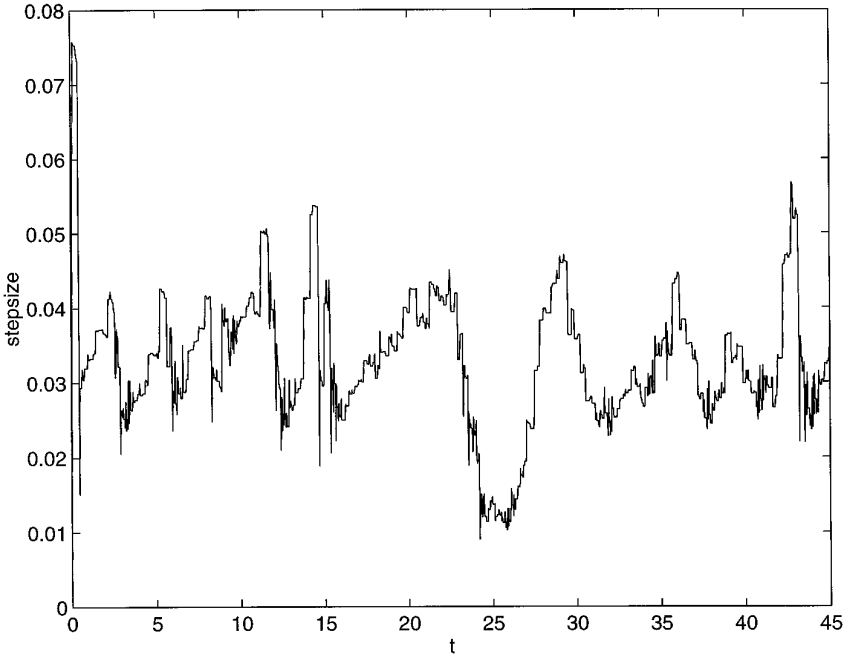

FIG. 7. Interaction between two solitons—evolution of the stepsize taken by RADAU5.

shown that for $q = 2N^2$ (where N is a positive integer) and an initial condition given by

$$u_0(x) = \operatorname{sech}(x), \tag{6.12}$$

$u(x, t)$ is a bound state of N solitons.

As in [16, 19, 20], $q = 18$, the time span of interest is (0,1), and the artificial boundaries are located at $x_L = -20$, $x_R = 20$. Versions 1–3 are used to solve this problem with the parameter values $N = 41$, $\alpha = 0.005$, $\beta = 100$, and $n$steps $= 10$. The numerical results obtained at $t = 0.98$ using version 3 are graphed in Fig. 8 and Table V presents some computational statistics. In his paper [16], Revilla proposes an adaptive solution of this problem which requires 101 nodes. Note that, on a fixed uniform grid, 1281 nodes are required [19].

Although these results seem very satisfactory, this difficult test problem, the solution of which develops very steep spatial and temporal gradients, highlights the limitations of our numerical scheme. As a further test, we solve the problem on a longer time interval (0, 4) using the same parameter values and we observe that the accuracy of the numerical solution gradually deteriorates. This is illustrated in Fig. 9, where the numerical results are graphed every 0.2 unit in t. We were not able to improve the efficiency of the method by tuning the algorithm parameters, unless the number of nodes is increased. Figure 10 illustrates the numerical results obtained with an adaptive grid of 101 points. In this latter case, the numerical results are better on the time interval considered, but the accuracy still decreases gradually, indicating that the instability is only postponed and will eventually show up.

In conclusion, we observe that our space/time adaptive algorithm, when applied to this test problem on a longer time interval, suffers from numerical instabilities. The source of instability is not clear to us and this issue, which was not addressed in [16, 20], would require more investigation.

We mention here that we have observed similar problems when solving this test example with the moving collocation code MOVCOL [11] by Huang and Russel. This method is a dynamic remeshing strategy in which the movement of the nodes is governed by a (moving mesh) PDE which is derived from equidistribution principles. To solve the bound state of three solitons, we have selected a monitor function based on the second-order derivative, an adaptive grid of 41 points, and the default values of the method parameters (no attempt has been made to adjust these parameters). In this case, the numerical results are very satisfactory for short times, but the accuracy decreases gradually until $t = 2$, after which the results very rapidly deteriorate.

At this stage, more work should be done in order to analyze these observations.

6.2. *A Model of a Single-Step Reaction with Diffusion*

This example has been treated numerically by Adjerid and Flaherty in [1] and by Petzold in [15]. The problem is described by

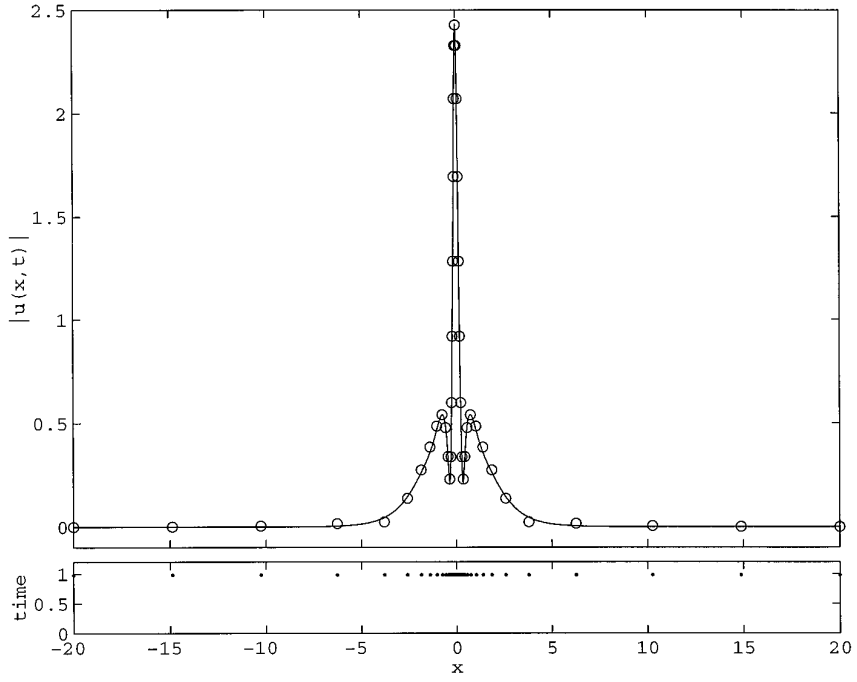

FIG. 8. Bound state of three solitons at $t = 0.98$—five-point finite differences; equidistribution principle (3.4b) with $N = 41$, $\alpha = 0.005$, $\beta = 100$, $n$steps $= 10$.

$$\frac{\partial T}{\partial t} = \frac{\partial^2 T}{\partial x^2} + D(1 + \kappa - T)e^{-\delta/T}, \qquad 0 < x < 1, t > 0 \tag{6.13}$$
$$T_x(0,t) = 0, T(1,t) = 1, T(x,0) = 1, \qquad 0 \le x \le 1$$

where $D = Re^{\delta}/\kappa\delta$.

The problem is solved on a time interval (0, 0.29) and for the parameter values $\kappa = 1$, $\delta = 20$, $R = 5$. For short times, the temperature gradually increases from unity with a maximum at $x = 0$. Suddenly an ignition occurs and the temperature at $x = 0$ rapidly increases to $1 + \kappa$. A steep front then forms and propagates towards $x = 1$ with a speed proportional to $e^{\kappa\delta}/2(1 + \kappa)$. The problem reaches a steady state once the front propagates to $x = 1$.

This problem is solved using versions 1–3 with the parameter values $N = 21$, $\alpha = 0.05$, $\beta = 1000$, and $n$steps $= 10$. Table VI presents some computational statistics. In Fig. 11, the solution obtained with version 2 is graphed from $t = 0.2$ to 0.29 at time intervals of 0.01 (before $t = 0.2$, the temperature increases gradually and the curves have been omitted in order to obtain a clearer picture) and is compared to a reference solution obtained with 101 adaptive grid points. For comparison purposes, we mention that this problem is solved in [1] using a moving finite element technique with local refinement and a maximum of 34 elements are required. So, this example demonstrates that our algorithm is competitive with other adaptive techniques.

TABLE V

Bound State of Three Solitons—Computational Statistics

Version	STEPS	FNS	JACS	CPU (s)
1	2265	12956	362	118
2	2079	11574	344	117
3	1098	6305	297	75

6.3. *A Model of Flame Propagation*

We consider a model of flame propagation proposed by Dwyer and Sanders and treated numerically in several papers, e.g., [15]. This model consists of two coupled equations for mass density and temperature which are given by

$$\frac{\partial \rho}{\partial t} = \frac{\partial^2 \rho}{\partial x^2} - N_{\mathrm{DA}}\rho,$$
$$\frac{\partial T}{\partial t} = \frac{\partial^2 T}{\partial x^2} + N_{\mathrm{DA}}\rho, \qquad 0 < x < 1, t > 0 \tag{6.14}$$

where $N_{\mathrm{DA}} = 3.52 \times 10^6 e^{-4/T}$.

The initial conditions are

$$\rho(x,0) = 1, T(x,0) = 0.2, \qquad 0 \le x \le 1 \tag{6.15}$$

The boundary conditions are

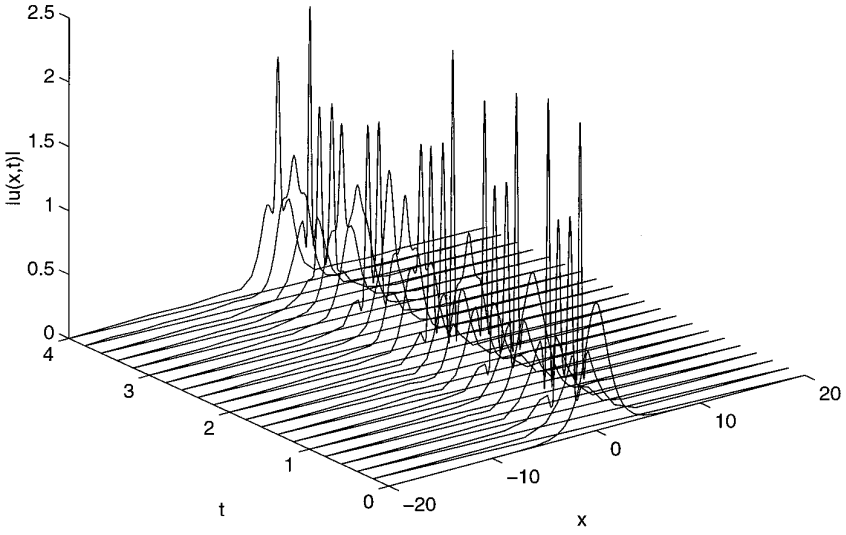

FIG. 9. Bound state of three solitons from $t = 0$ to $t = 4$ at time intervals of 0.2—five-point finite differences—Equidistribution principle (3.4b) with $N = 41$, $\alpha = 0.005$, $\beta = 100$, $n$steps $= 10$.

$$\begin{aligned} &\rho_x(0,t) = 0,\ T_x(0,t) = 0, \qquad t \geq 0 \\ &\rho_x(1,t) = 0,\ T(1,t) = f(t) \end{aligned} \tag{6.16}$$

where

$$\begin{aligned} f(t) &= 0.2 + t/2 \times 10^{-4}, \qquad t \leq 2 \times 10^{-4} \\ &= 1.2, \qquad\qquad\quad t \geq 2 \times 10^{-4}. \end{aligned}$$

The heat source at $x = 1$ generates a flame front which propagates from right to left at a relatively high speed. The density and temperature computed using version 2 with the parameter values $N = 51$, $\alpha = 0.001$, $\beta = 1000$, and $n$steps $= 10$ are compared in Figs. 12 and 13 with a reference solution obtained with 151 adaptive grid points. The time interval of interest is (0, 0.006) and the solution is graphed at intervals of 0.0006. Some computational statistics are presented in Table VII. In [15], a solution to this problem is computed using an adaptive mesh strategy which combines two mechanisms: a Lagrangian method based on minimizing the time rate of change of the solution in the moving coordinates and a refinement procedure (i.e., grid points are added or deleted after every time step)

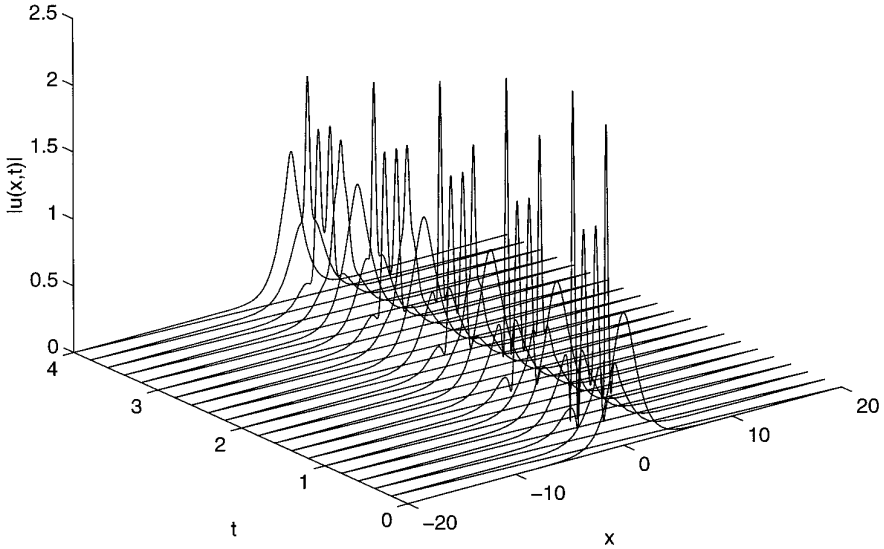

FIG. 10. Bound state of three solitons from $t = 0$ to 4 at time intervals of 0.2—five-point finite differences; equidistribution principle (3.4b) with $N = 101$, $\alpha = 0.005$, $\beta = 100$, $n$steps $= 10$.

TABLE VI

Single Step Reaction with Diffusion—Computational Statistics

Version	STEPS	FNS	JACS	CPU (s)
1	601	3596	190	4
2	561	3406	191	4
3	542	3266	167	4

which equidistributes a mesh function based on the first- and second-order derivatives of the solution. A numerical solution to the flame propagation problem is proposed which makes use of a maximum of 56 grid points. Our algorithm is probably not as efficient as this adaptive mesh strategy (the number of time steps taken is significantly larger in our case), but is considerably simpler.

7. CONCLUDING REMARKS

Although a great deal of interest has developed in spatial remeshing techniques over the past several years, only a few methods are available yet in the form of library components of MOL packages. In this paper, we investigate a class of methods based on equidistribution principles, and particularly, the method published by Sanz-Serna and Christie [20] and the extension proposed by Revilla [16]. The following points are discussed:

(a) The grid movement is directly attached to the equidistribution in space of an a priori chosen functional which is based on the second-order derivative of the solution. This criterion is easy to implement and to work with, i.e., the adaptive algorithm can be coded once and for all and the parameter tuning consists essentially in the choice of the number of grid points N and the scaling factor α.

(b) Following a method of line formulation, spatial discretization is required to transform the PDE problem into a set of DAEs. In contrast to the comments of Sanz-Serna and Christie and Revilla, we have found that the selection of a spatial discretization scheme has a significant influence on the efficiency of the solution procedure. In the test examples considered, the use of higher-order finite differences allows the number of grid points, and consequently the computational load, to be significantly reduced.

(c) Better performance has been obtained by updating the grid periodically, i.e., after a fixed number of integration steps, rather than by updating the grid alternatively, i.e., at each time step taken by the time integrator. Time integration is efficiently performed using the implicit RK solver RADAU5 [7].

(d) Numerical experiments have been carried out with five different test examples, including several soliton solutions of the CSE, a model of a single step reaction with diffusion, and a model of flame propagation. These examples illustrate the effectiveness of our space/time adaptive procedure.

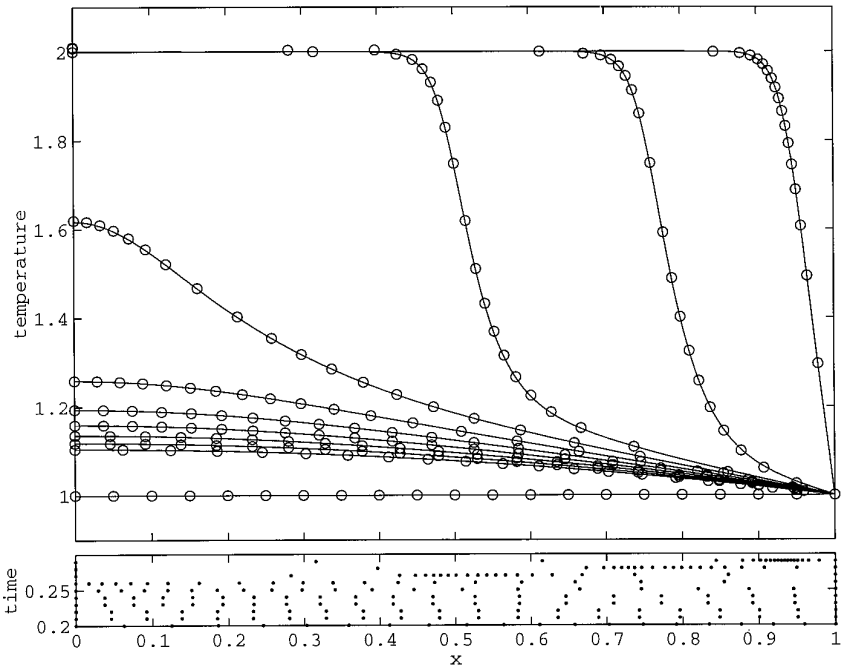

FIG. 11. Temperature from $t = 0.2$ to 0.29 at time intervals of 0.01—five-point finite differences; equidistribution principle (3.4a) with $N = 21$, $\alpha = 0.05$, $\beta = 1000$, $n$steps $= 10$.

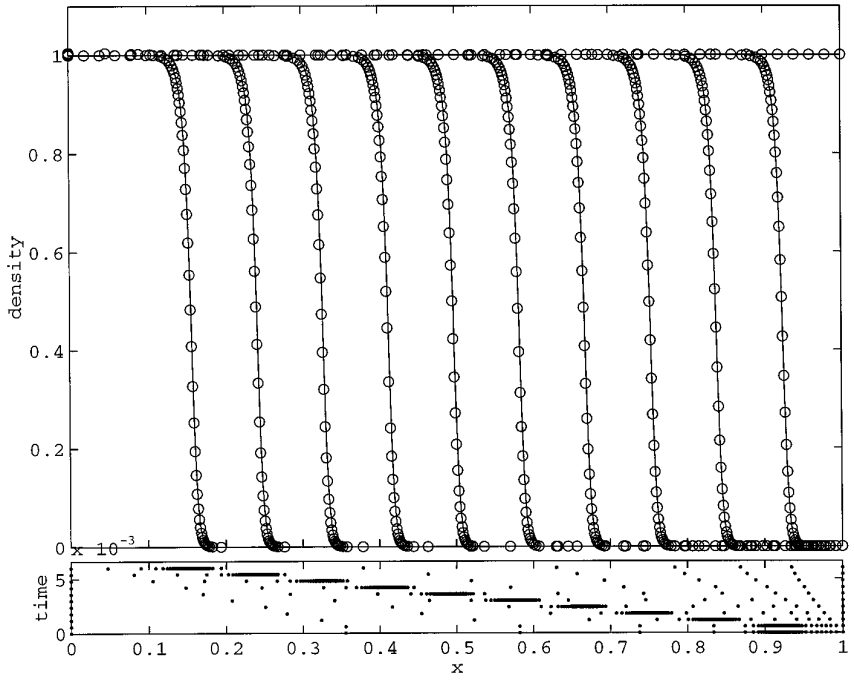

FIG. 12. Flame density from $t = 0$ to $t = 0.006$ at time intervals of 0.0006—five-point finite differences; equidistribution principle (3.4a) with $N = 51$, $\alpha = 0.001$, $\beta = 1000$, $n$steps $= 10$.

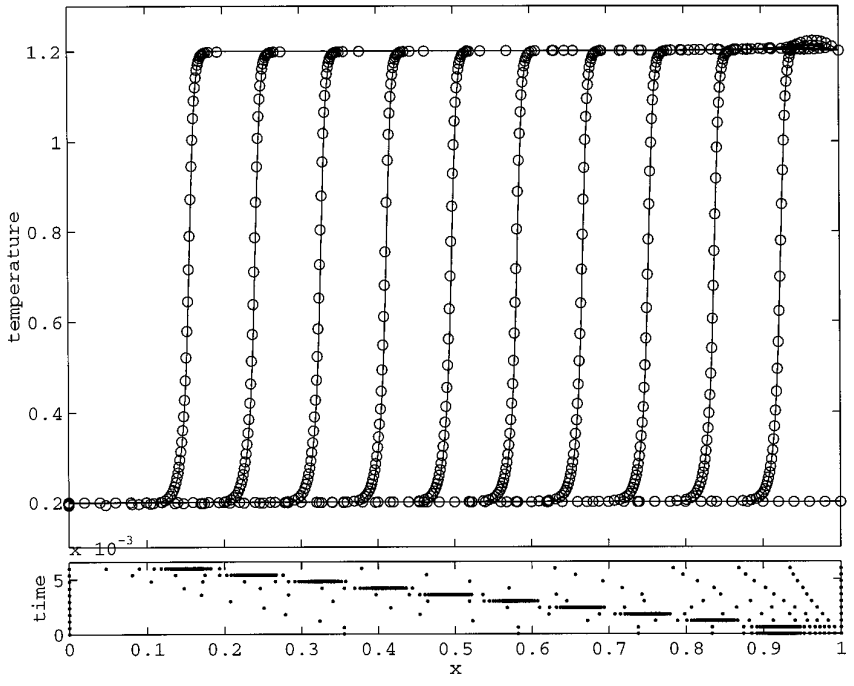

FIG. 13. Flame temperature from $t = 0$ to 0.006 at time intervals of 0.0006—five-point finite differences; equidistribution principle (3.4a) with $N = 51$, $\alpha = 0.001$, $\beta = 1000$, $n$steps $= 10$.

TABLE VII

Flame Propagation—Computational Statistics

Version	STEPS	FNS	JACS	CPU (s)
1	2576	15895	928	904
2	2096	13473	947	700
3	2117	13641	968	713

(e) However, our numerical scheme suffers from numerical instabilities which show up in the case of a bound state of three solitons. At this stage, more work is required in order to understand the source of this instability.

Some of the examples presented in this paper could probably be solved more efficiently using other adaptive grid strategies. For instance, the model of flame propagation has been solved very efficiently by Petzold in [15]. The major merits of our algorithm are its simplicity of implementation, its ease of use, and its capability of treating a variety of PDE problems.

As a result of this study, a spatial remeshing routine has been implemented in standard Fortran and is available on request from the authors. This routine can easily be linked to existing PDE packages like DSS/2 [21].

ACKNOWLEDGMENTS

We are very grateful for the referees' constructive criticisms and comments on the preliminary version of this paper. A. Vande Wouwer is supported by the Belgian National Research Fund (F.N.R.S.) as a Senior Research Assistant.

REFERENCES

1. S. Adjerid and J. E. Flaherty, A moving-mesh finite element method with local refinement for parabolic partial differential equations, *Comput. Meth. Appl. Mech. Eng.* **55,** 3 (1986).
2. J. B. Blom, J. M. Sanz-Serna, and J. G. Verwer, On simple moving grid methods for one-dimensional evolutionary partial differential equations, *J. Comp. Phys.* **74,** 191 (1988).
3. H. A. Dwyer, R. J. Kee, and B. R. Sanders, Adaptive grid method for problems in fluid mechanics and heat transfer, *AIAA J.* **18,** 1205 (1980).
4. B. Fornberg, Generation of finite difference formulas on arbitrarily spaced grid, *Math. Comp.* **51,** 699 (1988).
5. R. M. Furzeland, J. G. Verwer, and P. A. Zegeling, A numerical study of three moving-grid methods for one-dimensional partial differential equations which are based on the method of lines, *J. Comp. Phys.* **89,** 349 (1990).
6. D. F. Griffiths, A. R. Mitchell, and J. Ll. Morris, A numerical study of the nonlinear Schrödinger equation, *Comput. Methods Appl. Mech. Eng.* **45,** 177 (1984).
7. E. Hairer and G. Wanner, *Solving Ordinary Differential Equations.* II. *Stiff and Differential-Algebraic Problems* (Springer-Verlag, Berlin, 1991).
8. D. F. Hawken, J. J. Gottlieb, and J. S. Hansen, Review of some adaptive node-movement techniques in finite-element and finite difference solutions of partial differential equations, *J. Comp. Phys.* **95,** 254 (1991).
9. B. M. Herbst, J. Ll. Morris, and A. R. Mitchell, Numerical experience with the nonlinear Schrödinger equation, *J. Comp. Phys.* **60,** 282 (1985).
10. A. C. Hindmarsh, ODEPACK: A systematized collection of ODE solvers, in *Scientific Computing*, edited by R. S. Stepleman (IMACS, North-Holland, Amsterdam, 1983), p. 55.
11. W. Huang and R. D. Russell, A moving collocation method for solving time dependent partial differential equations, *Appl. Num. Math,* to appear.
12. J. W. Miles, An envelope soliton problem, *SIAM J. Appl. Math.* **41,** 227 (1981).
13. K. Miller and R. N. Miller, Moving finite elements, I, *SIAM J. Numer. Anal.* **18,** 1019 (1981).
14. K. Miller, Moving finite elements, II, *SIAM J. Numer. Anal.* **18,** 1033 (1981).
15. L. R. Petzold, Observation on an adaptive moving grid method for one-dimensional systems of partial differential equations, *Appl. Numer. Math.* **3,** 347 (1987).
16. M. A. Revilla, Simple time and space adaptation in one-dimensional evolutionary partial differential equation, *Int. J. Numer. Methods Eng.* **23,** 2263 (1986).
17. J. M. Sanz-Serna and V. S. Manoranjan, A method for the integration in time of certain partial differential equations, *J. Comp. Phys.* **52,** 273 (1983).
18. J. M. Sanz-Serna, Methods for the numerical solution of the nonlinear Schroedinger equation, *Math. Comp.* **43,** 21 (1984).
19. J. M. Sanz-Serna and J. G. Verwer, Conservative and nonconservative schemes for the solution of the nonlinear Schrödinger equation, *IMA J. Numer. Anal.* **6,** 25 (1986).
20. J. M. Sanz-Serna and I. Christie, A simple adaptive technique for nonlinear wave problems, *J. Comp. Phys.* **67,** 348 (1986).
21. W. E. Schiesser, *The Numerical Method of Lines* (Academic Press, San Diego, 1991).
22. W. A. Strauss, The nonlinear Schroedinger equation, in *Contemporary Developments in Continuum Mechanics and Partial Differential Equations*, edited by G. M. de la Penha and L. A. Medeiros (North-Holland, New York, 1978), pp. 452–465.
23. A. B. White, On the numerical solution of initial/boundary-value problems in one space dimension, *SIAM J. Numer. Anal.* **19,** 683 (1982).
24. G. B. Whitham, *Linear and Nonlinear Waves* (Wiley, New York, 1974).