Applied Mathematics and Computation 219 (2012) 3486–3495

Contents lists available at SciVerse ScienceDirect

Applied Mathematics and Computation

journal homepage: www.elsevier.com/locate/amc

Adaptive numerical method for Burgers-type nonlinear equations

Ali R. Soheili [a,b], Asghar Kerayechian [b], Noshin Davoodi [b,*]

[a] The Center of Excellence on Modelling and Control Systems, Ferdowsi University of Mashhad, Mashhad, Iran
[b] Department of Applied Mathematics, School of Mathematical Sciences, Ferdowsi University of Mashhad, Mashhad, Iran

ARTICLE INFO

Keywords:
Finite difference method
Adaptive moving mesh method
Burgers-type equations

ABSTRACT

In this paper a moving mesh partial differential equation (MMPDE) approach will be applied to some types of nonlinear partial differential equations. The method is described in detail, and a number of computational examples are presented using different monitor functions applied to the Burgers-type equations.

1. Introduction

In this paper, we are concerned with the numerical solution of time-dependent partial differential equations (PDEs) which involve large solution variations, (such as boundary layers or moving wave fronts), large gradients or discontinuities which often complicate the numerical solution of this equations. One of this equations is viscous Burgers' equation whose solutions have steep fronts (physically corresponding to shock waves in a fluid) travelling toward the grid end. Since a steep typically occurs on increasingly small length scales as well as time scales, it would be essential to use an adaptive mesh in the numerical simulation.

Two types of mesh adaptation have been commonly used, mesh refinement [3] and mesh movement [6]. With the former approach, mesh points are added as the length scale is getting smaller whereas in the latter approach a fixed number of mesh points are moved to resolve the increasingly small length scale. In this paper, we are interested in the moving mesh solution of problems and focus particularly on the MMPDE (moving mesh PDE) method developed in [8].

For the numerical solution of time-dependent differential equations, adaptive methods can be divided into two categories, including static and dynamic. For static methods, the discrete solution and equation are initially defined on a given mesh. During the computation, a new mesh that might have a different number of nodes from the old one is constructed, based on properties of a certain function which measures the goodness of the approximation. The solution is then interpolated from the old mesh to the new mesh, and a new discrete approximation to the solution is defined on the new mesh. The redistribution of the old nodes, the addition of new nodes, and the interpolation of the dependent variables from the old mesh to the new mesh are done at fixed time. In the dynamical moving mesh method, a mesh equation that involves node speeds is employed to move a mesh having a fixed number of nodes in such a way that the nodes remain concentrated in regions of rapid variation of the solution. The mesh equation and the original differential equation are often solved simultaneously for the physical solution and the mesh. Unlike static methods [11,14], interpolation of dependent variables from the old mesh to the new mesh is unnecessary.

An outline of the paper is as follows: Section 2 provides an introduction to moving mesh methods in one spatial dimension (1D), and several smooth monitor functions are studied.

In Section 3, we describe a selection of one-dimensional reaction-convection diffusion equation defined on a bounded domain as:

* Corresponding author.
E-mail addresses: soheili@ferdowsi.um.ac.ir (A.R. Soheili), davoodi.math@gmail.com (N. Davoodi).

0096-3003/$ - see front matter © 2012 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.amc.2012.09.019

1. 1D Burgers' equation,

$$u_t = \varepsilon u_{xx} - uu_x, \quad a \leqslant x \leqslant b, \quad t > 0 \tag{1.1}$$

where $\varepsilon > 0$ is an arbitrary physical parameter.

2. Burgers'–Fisher equation with variable and constant coefficients,

$$u_t - u_{xx} + \alpha(t)uu_x = \beta(t)u(1-u), \quad a \leqslant x \leqslant b \tag{1.2}$$

where $\alpha(t), \beta(t)$ are arbitrary functions of t.

3. Burgers'–Huxley equation,

$$u_t + \alpha u^\delta u_x - u_{xx} = \beta u(1-u^\delta)(u^\delta - \gamma), \quad 0 \leqslant x \leqslant 1 \tag{1.3}$$

where α, β, δ and γ are given constants. The mesh movement strategy and the PDE discretization for moving mesh are discussed in Section 4. And, in Section 5, numerical results have been presented for different monitor functions.

2. Moving mesh method in one dimension

Let x and ξ denote the physical and computational coordinates, respectively and consider a one-to-one transformation between these two coordinates:

$$\begin{aligned} &\Omega_c \equiv [0,1] \to \Omega \equiv [a,b],\\ &x = x(\xi,t), \qquad \xi \in [0,1],\\ &x(0,t) = a, \quad x(1,t) = b. \end{aligned}$$

A corresponding moving mesh can be described as

$$\mathrm{T}_h(t): \quad x_i(t) = x(\xi_i,t), \qquad i = 0,1,\cdots,N.$$

Suppose that a uniform mesh on Ω_c is given by

$$\mathrm{T}_h^c: \quad \xi_i = \frac{i}{N}, \qquad i = 0,1,\cdots,N,$$

where N is a certain positive integer and the corresponding mesh in the physical domain can be denoted by

$$a = x_0 < x_1(t) < x_2(t) < \cdots < x_{N-1}(t) < x_N = b.$$

The equidistribution principle (EP), [8], is one of the most important concepts in development of moving mesh methods. For a chosen monitor function $M(x,t) > 0$, mathematically, a fine mesh is a mesh to satisfy the following EP for all values of time t:

$$\int_a^{x_i(t)} M(x,t)dx = \frac{i}{N}\theta(t) = \xi_i\theta(t). \tag{2.1}$$

where $\theta(t) = \int_0^1 M(x,t)dx$. By differentiating the above equation we obtain

$$\frac{\partial}{\partial\xi}\left(M\frac{\partial x}{\partial\xi}\right)(\xi,t) = 0, \tag{2.2}$$

where $x(0,t) = a, x(1,t) = b$. Usually the mesh must be satisfied the above EP equation at the later time $t+\tau$ ($\tau = \Delta t$) instead of t, since $\theta(t)$ has been omitted in this form of EP. So, the mesh must satisfy

$$\frac{\partial}{\partial\xi}\{M(x(\xi,t+\tau),t+\tau)\frac{\partial}{\partial\xi}x(\xi,t+\tau)\} = 0,$$

the parameter τ is called a relaxation time. Recently, though, Soheili and Stockie [15] showed that for solutions that have a natural time scale which changes significantly over time, a constant value for τ can result in bad mesh adaptation. When τ is increased, the stiffness decreases, but the adaptation becomes less sharp (the mesh points may in fact 'lag behind'). When τ is decreased, the results are more accurate, but the system becomes stiffer. They propose a time dependent relaxation parameter $\tau(t)$ which is specific for the chosen MMPDE. This new approach is very effective, although it is still specific for self-similar blow-up problems only.

Using the expansions of $M(x(\xi,t+\tau),t+\tau)$ and $\frac{\partial}{\partial\xi}x(\xi,t+\tau)$ and dropping higher order term, various MMPDEs is obtained in [8]. In this paper, we apply modified MMPDE5 for moving mesh methods:

$$\frac{\partial x}{\partial t} = \frac{1}{\tau M}\frac{\partial}{\partial\xi}(M\frac{\partial x}{\partial\xi}), \tag{2.3}$$

which is supplemented with the boundary conditions

$$x(0,t)=a,\quad x(1,t)=b. \tag{2.4}$$

The monitor function (M) is used to guide the mesh redistribution. It would be very important to choose a suitable monitor function. It depends on the solution arc-length, curvature, and optimal mesh monitor functions [10].

- *Optimal mesh monitor function:*

$$M=\left(1+\frac{1}{\alpha}|u_{xx}|^2\right)^{\frac{1}{3}},\quad \alpha=\max\left\{1,\left(\frac{1}{b-a}\int_a^b |u_{xx}|^{\frac{2}{3}}dx\right)^3\right\}, \tag{2.5}$$

- *Arc-length mesh monitor function:*

$$M=(1+|u_x|^2)^{\frac{1}{2}}, \tag{2.6}$$

- *Curvature mesh monitor function:*

$$M=(1+|u_{xx}|^2)^{\frac{1}{4}}. \tag{2.7}$$

If u is not smooth, the approximation of derivatives are often non-smooth, especially the higher order ones. So the discrete monitor function can often change abruptly and unnecessarily slow down the computation. To obtain a smoother mesh and also make the MMPDE easier to integrate, it is common practice in the context of moving mesh methods to smooth the monitor function. For example, for a given M (viewed as a function of ξ through some coordinate transformation) a mesh density function having higher regularity or a smoother monitor function, $\widetilde{M}$, can be obtained as the solution of the boundary value problem

$$\begin{cases}(I-\beta^{-2}\frac{d^2}{d\xi^2})\widetilde{M}=M, & \forall\xi\in(0,1)\\ \frac{d\widetilde{M}}{d\xi}(0)=\frac{d\widetilde{M}}{d\xi}(1)=0,\end{cases} \tag{2.8}$$

where $\beta>0$ is a parameter and $I$is the identity operator. For more details see [9,10].

3. Description of the models

The Burgers' equations appear in various areas of applied mathematics, such as modeling of fluid dynamics, turbulence, boundary layer behavior, shock wave formation, and traffic flow. As it is known, firstly the Burgers' Eq. (1.1) was introduced to describe turbulence in one space dimension and has been used in several other physical contexts, including sound waves in viscous media [4,5].

The Fisher equation [17,19] is a nonlinear partial differential equation of second order of the form

$$u_t-u_{xx}=\beta(t)u(1-u),\qquad a\leqslant x\leqslant b. \tag{3.1}$$

Fisher proposed this equation as a model for the propagation of a mutant gene with $u(x,t)$ denoting the density of advantageous. This equation is encountered in chemical kinetics, population dynamics, flame propagation, autocatalytic chemical reactions and branching Brownian motion processes.

The Burgers–Fisher Eq. (1.2) has important applications in various fields of financial mathematics, gas dynamic, traffic flow, applied mathematics and physics applications [12,13]. This equation shows a prototypical model for describing the interaction between the reaction mechanism, convection effect, and diffusion transport [16].

The Huxley equation [20] is a nonlinear partial differential equation of second order of the form

$$u_t-u_{xx}=\beta u(1-u^\delta)(u^\delta-\gamma),\qquad 0\leqslant x\leqslant 1. \tag{3.2}$$

This equation is an evolution equation that describes the nerve propagation in biology from which molecular and cellular biology (CB) properties can be calculated.

The Burgers–Huxley Eq. (1.3) being a nonlinear partial differential equation is of high importance for describing the interaction between reaction mechanisms, convection effects, and diffusion transports. Since there exists no general technique for finding analytical solutions of nonlinear diffusion equations so far, numerical solutions of nonlinear differential equations are of great importance in physical problems. There are many researchers who used various numerical techniques to obtain numerical solution of the Burgers–Huxley equation. Wang et al. [18] studied the solitary wave solutions of the generalized Burgers–Huxley equation. In the past few years, various powerful mathematical methods such as Adomian decomposition method [12], homotopy analysis method, the tanh-coth method [21], variational iteration method [1,2], have been used in attempting to solve this equation.

In addition to these nonlinear evolution equations, the equations (1.1)–(1.3) will be investigated.

4. Numerical solutions by the moving mesh method

We employ the method of lines approach in which these equations are discretized with second, third and fourth order spatial accuracy using central finite difference schemes to obtain the discritization equation for the solution where the "dot" refers to the time derivative.

1. 1D Burgers' equation,

$$u_t = \varepsilon u_{xx} - uu_x, \qquad a \leqslant x \leqslant b, \quad t > 0, \tag{4.1}$$

where α and β are arbitrary constants.
The Burgers' equation can be written in the computational coordinate form:

$$\dot{u} - \frac{u_\xi}{x_\xi}\dot{x} = \frac{\varepsilon}{x_\xi}\left(\frac{u_\xi}{x_\xi}\right)_\xi - u\frac{u_\xi}{x_\xi}.$$

Central finite difference can then be used to discretize the above equation on the uniform computational mesh T_h^c. This yields

$$\frac{du_i}{dt} - \frac{(u_{i+1}-u_{i-1})}{(x_{i+1}-x_{i-1})}\frac{dx_i}{dt} = \frac{2\varepsilon}{(x_{i+1}-x_{i-1})}\left[\frac{(u_{i+1}-u_i)}{(x_{i+1}-x_i)} - \frac{(u_i-u_{i-1})}{(x_i-x_{i-1})}\right] - u_i\frac{(u_{i+1}-u_{i-1})}{(x_{i+1}-x_{i-1})}, \quad i = 1,\ldots,N-1, \tag{4.2}$$

where $x_i \approx x(\xi_i, t), u_i \approx u(x(\xi_i, t), t)$.

2. Burgers–Fisher equation with variable and constant coefficients,

$$u_t - u_{xx} + \alpha(t)uu_x = \beta(t)u(1-u), \qquad a \leqslant x \leqslant b, \tag{4.3}$$

where $\alpha(t), \beta(t)$ are arbitrary functions of t.
The Burgers–Fisher equation can be written in the computational coordinate form:

$$\dot{u} - \frac{u_\xi}{x_\xi}\dot{x} - \frac{1}{x_\xi}\left(\frac{u_\xi}{x_\xi}\right)_\xi + \alpha(t)u\frac{u_\xi}{x_\xi} = \beta(t)u(1-u). \tag{4.4}$$

discretization of the above equation on the uniform computational mesh gives

$$\frac{du_i}{dt} - \frac{(u_{i+1}-u_{i-1})}{(x_{i+1}-x_{i-1})}\frac{dx_i}{dt} - \frac{2}{(x_{i+1}-x_{i-1})}\left[\frac{(u_{i+1}-u_i)}{(x_{i+1}-x_i)} - \frac{(u_i-u_{i-1})}{(x_i-x_{i-1})}\right] + \alpha(t)u_i\frac{(u_{i+1}-u_{i-1})}{(x_{i+1}-x_{i-1})} = \beta(t)u_i(1-u_i), \quad i = 1,\ldots,N-1. \tag{4.5}$$

3. Burgers–Huxley equation,

$$u_t + \alpha u^\delta u_x - u_{xx} = \beta u(1-u^\delta)(u^\delta - \gamma), \qquad 0 \leqslant x \leqslant 1, \tag{4.6}$$

where α, β, δ and γ are given constants.
The Burgers–Huxley equation can be written in the computational coordinate form:

$$\dot{u} - \frac{u_\xi}{x_\xi}\dot{x} + \alpha u^\delta\frac{u_\xi}{x_\xi} - \frac{1}{x_\xi}\left(\frac{u_\xi}{x_\xi}\right)_\xi = \beta u(1-u^\delta)(u^\delta - \gamma). \tag{4.7}$$

with discretization form:

$$\frac{du_i}{dt} - \frac{(u_{i+1}-u_{i-1})}{(x_{i+1}-x_{i-1})}\frac{dx_i}{dt} + \alpha u_i^\delta\frac{(u_{i+1}-u_{i-1})}{(x_{i+1}-x_{i-1})} - \frac{2}{(x_{i+1}-x_{i-1})}\left[\frac{(u_{i+1}-u_i)}{(x_{i+1}-x_i)} - \frac{(u_i-u_{i-1})}{(x_i-x_{i-1})}\right] = \beta u_i(1-u_i^\delta)(u_i^\delta - \gamma), \quad i = 1,\ldots,N-1. \tag{4.8}$$

Semi-discretization of MMPDE (2.3) on the uniform mesh T_h^c gives, for i = 1,...,N − 1,

$$\frac{dx_i}{dt} = \frac{1}{\widetilde{M}_i\tau\Delta\xi^2}\left[\frac{\widetilde{M}_{i+1}+\widetilde{M}_i}{2}(x_{i+1}-x_i) - \frac{\widetilde{M}_i+\widetilde{M}_{i-1}}{2}(x_i-x_{i-1})\right], \tag{4.9}$$

where $\widetilde{M}_i = \widetilde{M}(x_i, t)$ and the boundary conditions (2.4) become

$$\frac{dx_0}{dt} = 0, \quad \frac{dx_N}{dt} = 0. \tag{4.10}$$

In practice, the monitor function M is always associated with the underlying solution u or/and its derivatives, but without loss of generality, we assume that $\tilde{M} = \tilde{M}(u)$ as defined (2.5)–(2.7).
Consider a central finite difference discretization of (2.8) on a uniform mesh of N points,

$$\tilde{M}_i = \frac{1}{\beta^2\Delta\xi^2}M_{i+1} + (1 - \frac{2}{\beta^2\Delta\xi^2})M_i + \frac{1}{\beta^2\Delta\xi^2}M_{i-1}, \quad i = 1, \ldots, N-1, \tag{4.11}$$

which is an averaging if β is chosen such that $\beta^2\Delta\xi^2 \geqslant 2$ ($M_i = M(x_i, t)$). In particular, when $\beta^2\Delta\xi^2 = 4$ and the boundary condition is taken properly, (4.11) reduces to the weighted averaging, i.e.,

$$\begin{cases} \tilde{M}_i & := \frac{1}{4}M_{i-1} + \frac{1}{2}M_i + \frac{1}{4}M_{i+1}, \quad i = 1, \ldots, N-1, \\ \tilde{M}_1 & := \frac{1}{2}M_1 + \frac{1}{2}M_2, \\ \tilde{M}_N & := \frac{1}{2}M_{N-1} + \frac{1}{2}M_N, \end{cases} \tag{4.12}$$

where the symbol := stands for the operation in which the right-hand side terms are calculated and the final value is saved to the variable on the left-hand side.
Eqs. (4.2), or (4.5), or discretization of (4.7) with (4.8), supplemented with the boundary conditions (4.9), form a coupled system of $2N$, or $2N-2$ ordinary differential equations (depends on the boundary conditions) for the physical solutions $u_0(t), \ldots, u_N(t)$ and mesh $x_0(t), \ldots, x_N(t)$.

5. Numerical experiments

In this section, the numerical results of computational solutions and mesh trajectories of the Burgers-type equations are presented. Matlab ODE solver (ode15s) has been used to integrate the resulting system of ordinary differential equations. In all numerical examples, the numerical solutions correspond to the curvature monitor function. Other monitor functions have relatively the same solutions.

Example 1. *1D Burgers' equation.* For the first example, we test our adaptive mesh algorithm with the 1D Burgers' Eq. (1.1), subject to the boundary conditions

$$u(0,t) = u(1,t) = 0,$$

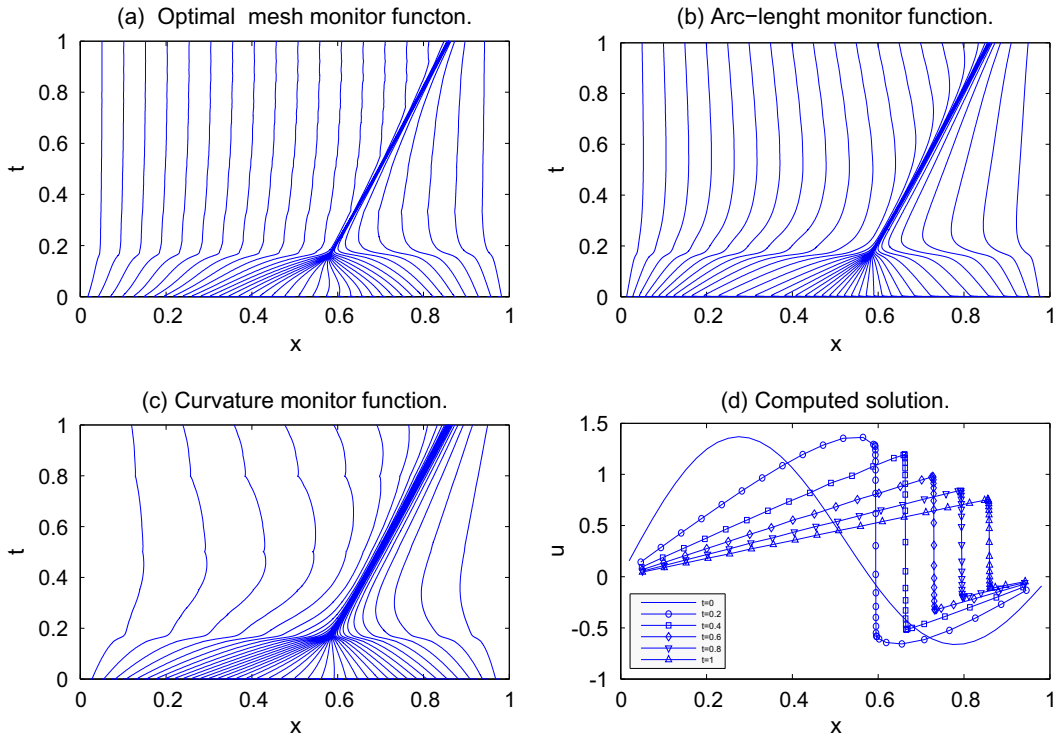

Fig. 1. Example 1. (a–c) The corresponding mesh trajectories with 40 mesh points using different monitor functions. (d) Computed solution obtained at $t = 0$, 0.2, 0.4, 0.6, 0.8, and 1 with curvature monitor function.

and initial condition (see [10])

$$u(x,0) = \sin(2\pi x) + \frac{1}{2}\sin(\pi x),$$

The solution and mesh trajectories of the Burgers' equation computed for $\varepsilon = 10^{-4}$ are shown in Fig. 1. In our computation, an initial uniform mesh with $N = 40$ and the value $\tau = 10^{-4}$ are used. The mesh trajectories show how the mesh points respond quickly to the change in the solution. Fig. 1(a–c) shows how differently the nodes are distributed according to optimal, arc-length, and curvature mesh monitor functions, after applying a few iterations smoothing scheme, respectively.

Example 2 (*Burgers–Fisher equation*). For the second example, we consider the Burgers–Fisher Eq. (1.2), with constant coefficients $\alpha(t) = a$, and $\beta(t) = \frac{2ac-a^2}{4}$, where c is an arbitrary constant. The exact solution of this equation subject to the boundary conditions,

$$u(-1,t) = \frac{1}{2} - \frac{1}{2}\tanh\left[\frac{a}{4}(-1-ct)\right],$$
$$u(0,t) = \frac{1}{2} - \frac{1}{2}\tanh\left[\frac{a}{4}(-ct)\right],$$

and the initial condition,

$$u(x,-0.2) = \frac{1}{2} - \frac{1}{2}\tanh\left[\frac{a}{4}(x+0.2c)\right],$$

is (see [7]):

$$u(x,t) = \frac{1}{2} - \frac{1}{2}\tanh\left[\frac{a}{4}(x-ct)\right].$$

The numerical results are obtained with $a = 24, c = 8$, and $\tau = 10^{-6}$, and 60 discrete points (see Fig. 2).

Since the maximum L^2-error occurs in the regions of steep gradient, we present the L^2-error at these times.

In Table 1, L^2-norm of errors are listed for $t = -0.1, -0.05, -0.04, -0.035, -0.03$. Although computed error by the arc-length monitor function in some region is less than others monitor functions, but the maximum L^2-error that occurs in sharp gradient regions is larger.

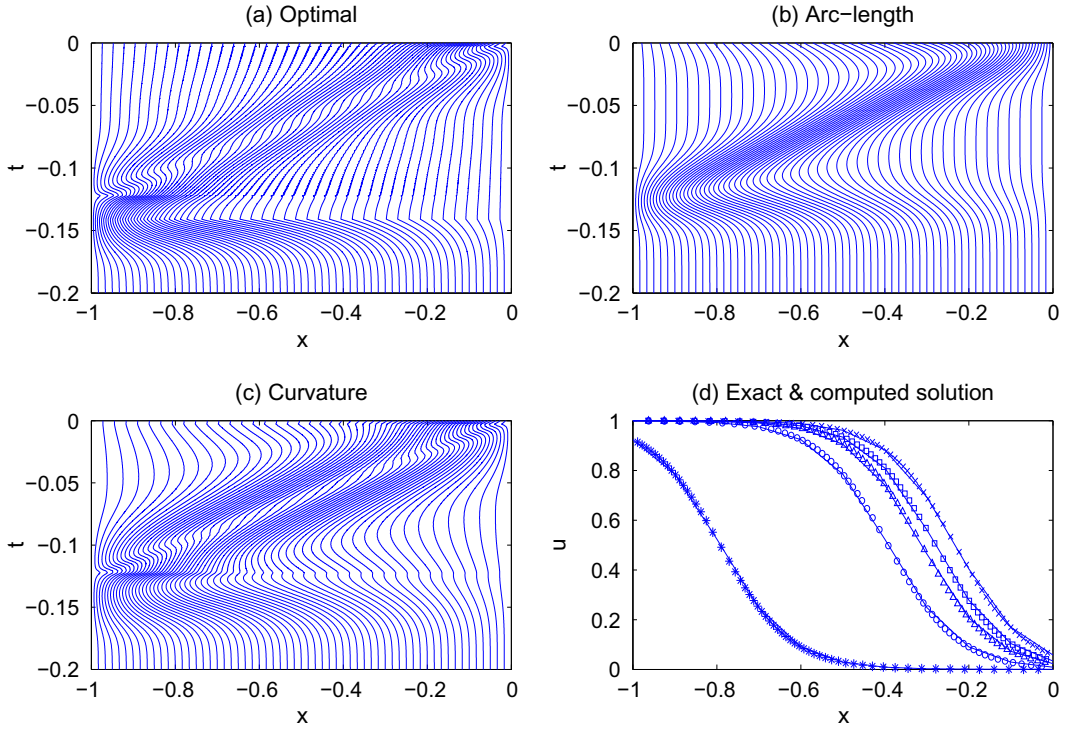

Fig. 2. Example 2. (a–c) The corresponding mesh trajectories with $N = 60$ mesh points using different monitor functions. (d) $(*, \circ, \triangle, \square, \times)$ and solid lines denote numerical and exact solutions at $t = -0.1, -0.05, -0.04, -0.035, -0.03$, respectively corresponding to the curvature monitor function.

Table 1
Example 2. L^2-error for equidistributing mesh with $N = 60$.

t	$t = -0.1$	$t = -0.05$	$t = -0.04$	$t = -0.035$	$t = -0.03$
Mesh for optimal M	2.1e−003	2.9e−003	3.4e−003	3.7e−003	4.2e−003
Mesh for arc-length M	2.7e−003	8.0e−003	7.9e−003	7.6e−003	7.1e−003
Mesh for curvature M	2.1e−003	2.4e−003	2.4e−003	2.4e−003	2.5e−003

Table 2
Example 3. L^2-error for equidistributing mesh with $N = 40$.

t	$t = -0.05$	$t = -0.025$	$t = 0$	$t = 0.025$	$t = 0.05$
Mesh for optimal M	2.2e−003	4.0e−003	5.3e−003	3.5e−003	1.2e−003
Mesh for arc-length M	1.8e−003	4.1e−003	6.6e−003	6.8e−003	2.3e−003
Mesh for curvature M	2.2e−003	4.0e−003	5.2e−003	3.6e−003	9.6e−004

Example 3. *Burgers–Fisher equation with variable coefficients.*

In Eq. (1.2), If we let $\alpha(t) = 20$ and $\beta(t) = -1 + 3\sin(t)$, then the exact solution is (see [7]):

$$u(x,t) = \frac{\cosh[x + \int(1+\beta(t))] + \sinh[x + \int(1+\beta(t))] - \sqrt{5}}{-4\cosh[x + \int(1+\beta(t))] + 6\sinh[x + \int(1+\beta(t))]}. \tag{5.1}$$

We may obtain the boundary and initial conditions from the exact solution.

Table 2 shows the L^2-errors of function at t= −0.05, −0.025, 0, 0.025, and 0.05.

This table indicates that in some times, the arc-length monitor function may produce an adaptive mesh that is inferior than those generated using the optimal and curvature monitor functions.

Example 4 (*Burgers–Huxley equation*). Here we apply the adaptive mesh algorithm to Burgers–Huxley equation with the boundary conditions

$$u(0,t) = \left(\frac{\gamma}{2} + \frac{\gamma}{2}\tanh(-a_1 a_2 t)\right)^{\frac{1}{\delta}},$$
$$u(1,t) = \left(\frac{\gamma}{2} + \frac{\gamma}{2}\tanh(a_1(1 - a_2 t))\right)^{\frac{1}{\delta}},$$

and the initial condition

$$u(x,0) = \left(\frac{\gamma}{2} + \frac{\gamma}{2}\tanh(a_1 x)\right)^{\frac{1}{\delta}},$$

where

$$a_1 = \frac{-\alpha\delta + \delta\sqrt{\alpha^2 + 4\beta(1+\delta)}}{4(1+\delta)}\gamma,$$

and

$$a_2 = \frac{\gamma\alpha}{1+\delta} - \frac{(1+\delta-\alpha)(-\alpha + \sqrt{\alpha^2 + 4\beta(1+\delta)})}{2(1+\delta)}.$$

The exact solution of this equation is (see [12])

$$u(x,t) = \left(\frac{\gamma}{2} + \frac{\gamma}{2}\tanh(a_1(x - a_2 t))\right)^{\frac{1}{\delta}}.$$

In this example, we consider $\gamma = 2, \alpha = 3, \beta = 60$, and $\delta = 1$, and $\tau = 10^{-4}$.

Table 3
Example 4. L^2-error for equidistributing mesh with $N = 60$.

t	$t = 0.2$	$t = 0.3$	$t = 0.4$	$t = 0.5$	$t = 0.6$
Mesh for optimal M	1.26e−002	1.60e−002	1.96e−002	2.32e−002	2.04e−002
Mesh for arc-length M	1.83e−002	2.39e−002	2.95e−002	3.51e−002	3.20e−002
Mesh for curvature M	1.15e−002	1.38e−002	1.61e−002	1.85e−002	1.69e−002

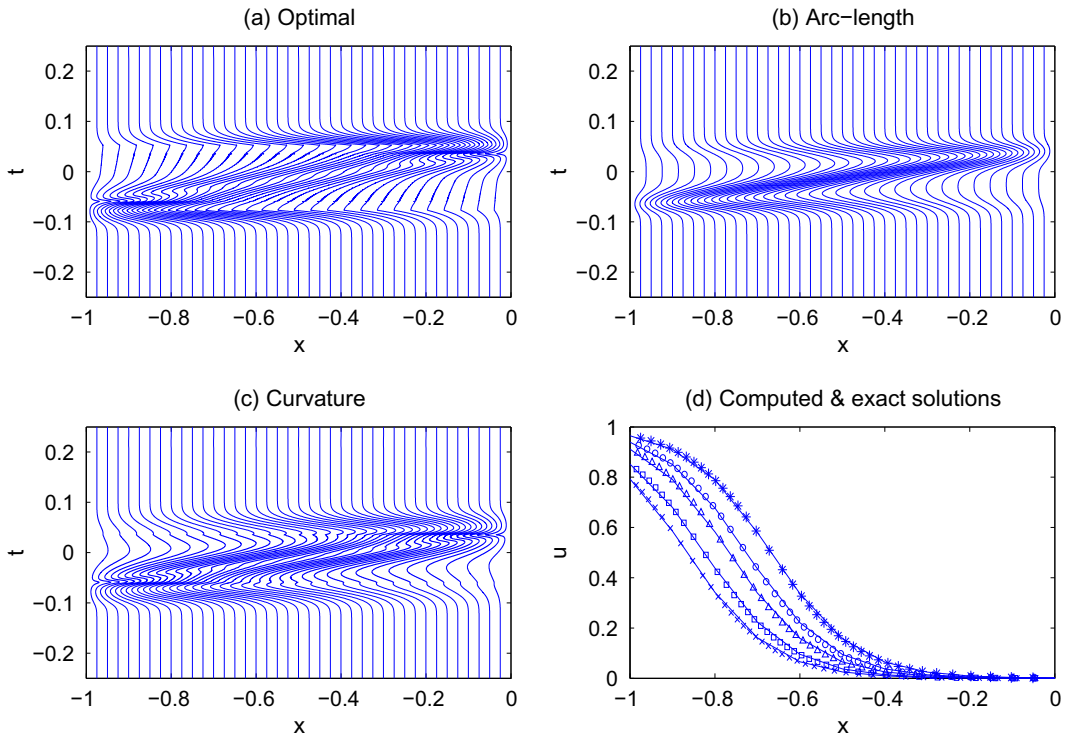

Fig. 3. Example 3. (a–c) The corresponding mesh trajectories with $N = 40$ mesh points using different monitor functions. (d) Numerical ($\times, \square, \triangledown, \circ, *,$) and exact solutions (solid line) at $t = -0.05, -0.025, 0, 0.025, 0.05$, respectively corresponding to the curvature monitor function.

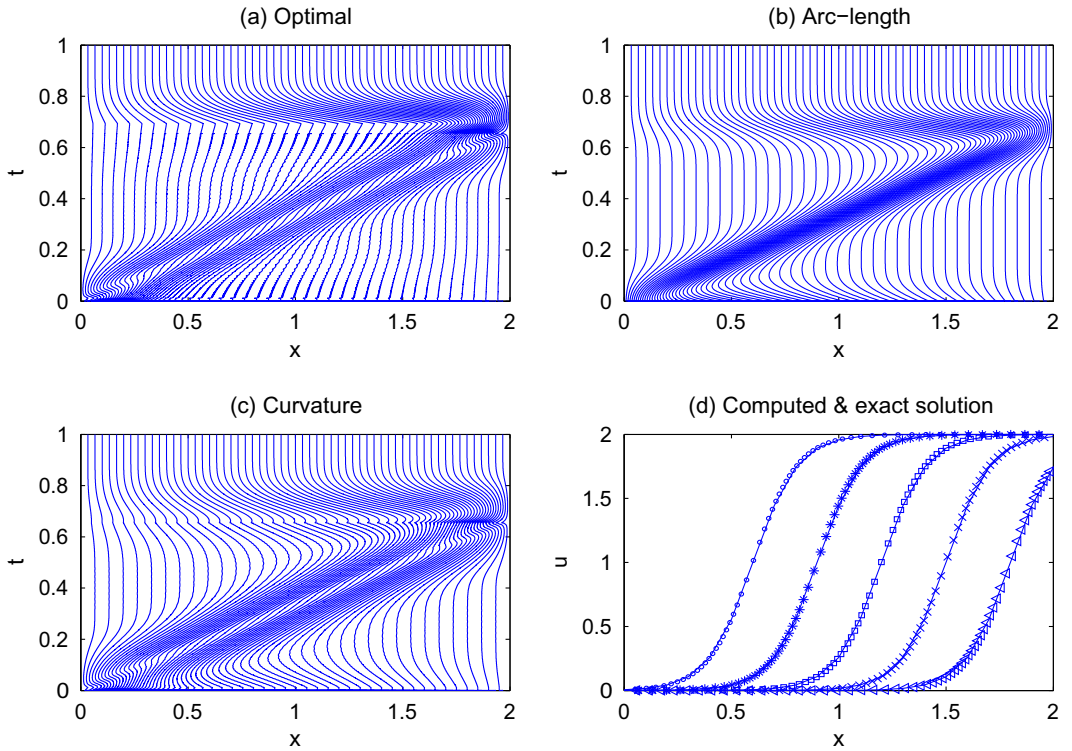

Fig. 4. Example 4. (a–c) The corresponding mesh trajectories with $N = 60$ mesh points using different monitor functions. (d). Numerical ($\circ, *, \square, \times, \triangleleft,$) and exact solutions (solid line) at $t = 0.2, 0.3, 0.4, 0.5, 0.6$, respectively corresponding to the curvature monitor function.

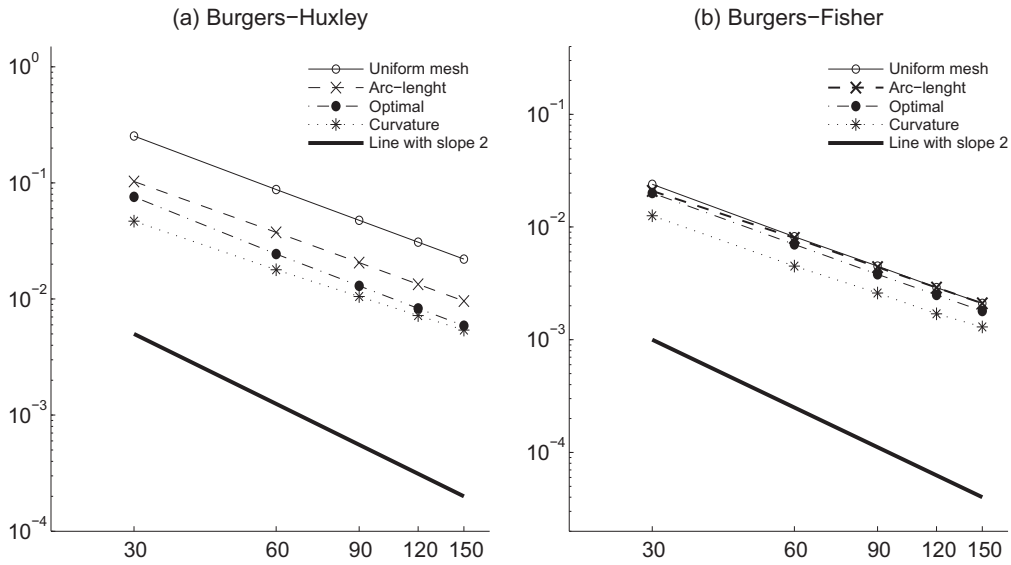

Fig. 5. The maximum norm of L^2-norm are plotted as functions of N for various monitor functions for Example 4, and Example 2 in (a) and (b), respectively.

From the Table 3, It is observed that in region with large solution variations, the arc-length monitor function works worse than the optimal monitor functions and both of them work worse than the curvature monitor function.

Although, in general we can not say which monitor function works better, in all of these examples, the curvature monitor function works better than others.

The ability of the adaptive mesh method to capture and follow the moving steep fronts is clearly demonstrated in Figs. 1–4.

6. Error bounds

For easy reference as well as comparison purposes, we list here corresponding error bounds for equidistributing meshes from the monitor functions described in the preceding sections.

$$\begin{aligned} &\ln E = a\ln N + b, \text{where} \quad a < 0, \\ &E = e^{\ln N^a + b} = N^a e^b = \alpha(\tfrac{1}{N})^{|a|}, \end{aligned} \tag{6.1}$$

$$\frac{1}{N} \leqslant \beta \max_i (x_i - x_{i-1}) = \beta H$$

so,

$$\lim_{N\to\infty} |u_N - u(x)| \leqslant \mu H^{|a|} = O(H^{|a|}).$$

Fig. 5 shows the maximum norm of the L^2-norm error as a function of N for a uniform mesh and for the meshes equidistributing the optimal, arc-length, and curvature monitor functions. The numerical results confirm that the convergence orders are roughly $O(N^{-2})$ as $N \to \infty$. The results also show that all adaptive meshes produce smaller error than a uniform mesh does, while the curvature monitor function perform better than the optimal and the arc-length monitor functions. As can be seen from these examples, the advantages of the adaptive mesh methods are quite obvious.

References

[1] B. Batiha, M.S.M. Noorani, I. Hashim, Application of variational iteration method to the generalized Burgers–Huxley equation, Chaos, Solitons Fractals 36 (3) (2008) 660–663.
[2] B. Batiha, M.S.M. Noorani, I. Hashim, Numerical simulation of the generalized Huxley equation by He's variational iteration method, Appl. Math. Comput. 186 (2) (2007) 1322–1325.
[3] M. Berger, R.V. Kohn, A rescaling algorithm for the numerical calculation of blowing-up solutions, Commun. Pure Appl. Math. (1988) 841–863.
[4] J.M. Burgers, A mathematical model illustrating the theory of turbulence, Adv. Appl. Mech. 1 (1948) 171–199.
[5] J.M. Burgers, The Nonlinear Diffusion Equation, Reidel, Boston, 1974.
[6] C.J. Budd, W. Huang, R.D. Russell, Moving mesh methods for problems with blow-up, SIAM J. Sci. Comput. 17 (1996) 305–327.
[7] B.K. Chen, Y. Li, H.L. Chen, B.H. Wang, Exp-function method for solving the Burgers–Fisher equation with variable coefficients, eprint arXiv: 1004.1815,04/2010

[8] W. Huang, Y. Ren, R.D. Russell, Moving mesh partial differential equations (MMPDEs) based upon the equidistribution principle, SIAM J. Numer. Anal. 31 (1994) 709–730.
[9] W. Huang, R.D. Russell, Analysis of moving mesh partial differential equations with spatial smoothing, SIAM J. Numer. Anal. 34 (1997) 1106–1126.
[10] W. Huang, R.D. Russell, Adaptive Moving Mesh Methods, Springer, New York, 2011.
[11] J.M. Hyman, Shengtai Li, L.R. Petzold, An adaptive moving mesh method with static rezoning for partial differential equations, Computers Math. Appl. 46 (10–11) (2003) 1511–1524.
[12] H.N.A. Ismail, K. Raslan, A.A.A. Rabboh, Adomian decomposition method for Burger's–Huxley and Burger's–Fisher equations, Appl. Math. Comput. 159 (1) (2004) 291–301.
[13] D. Kaya, S.M. El-Sayed, A numerical simulation and explicit solutions of the generalized Burgers–Fisher equation, Appl. Math. Comput. 152 (2) (2003) 403–413.
[14] R. Marlow, M.E. Hubbard, P.K. Jimack, Moving mesh methods for solving parabolic partial differential equations, Computers Fluids 46 (1) (2011) 353–361.
[15] A.R. Soheili, J.M. Stockie, A moving mesh method with variable mesh relaxation time, Appl. Numer. Math. 58 (3) (2008) 249–263.
[16] X. Wang, Y. Lu, Exact solutions of the extended Burger–Fisher equation, Chin. Phys. Lett. 7 (4) (1990) 145.
[17] X.Y. Wang, Exact and explicit solitary wave solutions for the generalized Fisher equation, Phys. Lett. A 131 (4/5) (1988) 277–289.
[18] X.Y. Wang, Z.S. Zhu, Y.K. Lu, Solitary wave solutions of the generalised Burgers–Huxley equation, J. Phys. A 23 (3) (1990) 271–274.
[19] A.M. Wazwaz, A. Georguis, An analytic study of Fisher's equation by using Adomian decomposition method, Appl. Math. Comput. 154 (2004) 609–620.
[20] A.M. Wazwaz, Travelling wave solutions of generalized forms of Burgers, Burgers–KdV and Burgers–Huxley equations, Appl. Math. Comput. 169 (2005) 639–656.
[21] A.M. Wazwaz, Analytic study on Burgers, Fisher, Huxley equations and combined forms of these equations, Appl. Math. Comput. 195 (2) (2008) 754–761.