Journal of Computational Physics **170,** 871–892 (2001)
doi:10.1006/jcph.2001.6770, available online at http://www.idealibrary.com on IDEAL®

# An Error Indicator Monitor Function for an *r*-Adaptive Finite-Element Method

Weiming Cao,* Weizhang Huang,† and Robert D. Russell‡

*Division of Mathematics and Statistics, University of Texas at San Antonio, San Antonio, Texas 78249; †Department of Mathematics, University of Kansas, Lawrence, Kansas 66045; and ‡Department of Mathematics and Statistics, Simon Fraser University, Burnaby, British Columbia V5A 1S6, Canada
E-mail: wcao@math.utsa.edu, huang@math.ukans.edu, rdr@cs.sfu.ca

Received March 22, 2000; revised January 16, 2001

An *r*-adaptive finite-element method based on moving-mesh partial differential equations (PDEs) and an error indicator is presented. The error indicator is obtained by applying a technique developed by Bank and Weiser to elliptic equations which result in this case from temporal discretization of the underlying physical PDEs on moving meshes. The construction of the monitor function based on the error indicator is discussed. Numerical results obtained with the current method and the commonly used method based on solution gradients are presented and analyzed for several examples. 

*Key Words:* moving-mesh method; adaptive finite-element method; mesh adaptation; a posteriori error estimate.

## 1. INTRODUCTION

For problems exhibiting large variations in spatial and temporal scales, such as those with boundary or internal layers, shock waves, and blowup of solutions, adaptive methods are indispensable for their efficient numerical solution. The three major types of adaptive finite-element methods are the *h*-, *p*-, and *r*-methods. For the *h*-method, the mesh is refined or coarsened by adding or deleting grid points, while the adaptivity of the *p*-method is achieved by changing the degree of the polynomial approximation used in each element. For the *r*-method, or the moving-mesh method, the mesh connectivity is kept unchanged but the grid points are shifted throughout the region as needed to best approximate the solution globally.

There has been extensive study of the *h*- and *p*-methods, and they have been shown to be reliable and efficient for the finite-element solution of partial differential equations (PDEs), particularly steady-state problems. The *r*-method has been less popular, largely because of the difficulty in developing a general and robust moving-mesh method in higher



0021-9991/01 $35.00

dimensions. Nevertheless, there are distinct potential advantages to the $r$-method: e.g., the relative ease of coding in comparison with mesh subdivision, which requires complicated tree data structures; no need of interpolation between different levels of mesh refinement, which can cause extra numerical dissipation [15]; the ease of incorporating the method in existing codes based on fixed grids; and the simplicity in principle of computing the mesh using continuous time integration. Indeed, continuously changing the positions of grid points is naturally consistent with the evolutionary features of time-dependent problems. Moving-mesh methods have been shown to be very successful for large classes of one-dimensional problems (e.g., see [21, 27]) and for some higher dimensional problems [11, 13–15, 31].

There are several ways to accomplish $r$-adaptivity. In one dimension, most of the procedures rely on the so-called equidistribution principle [10, 21, 24]. However, the situation is not so straightforward in higher dimensions. Miller [27] proposed a moving finite-element method which relocates the grid points by minimizing the residual (see [4] for a detailed description). Liao and co-workers developed a moving-mesh method based on deformation mappings (e.g., see [31]). Huang and Russell [25] developed a moving-mesh method based on a set of parabolic PDEs, so-called moving-mesh PDEs (MMPDEs). The method is formulated on a commonly used variational framework and involves minimization of a quadratic functional describing mesh properties such as concentration, alignment, and orthogonality.

A key issue for the moving-mesh strategy is the selection of a so-called monitor function to use in the variational formulation which will properly control the mesh properties and interconnect the mesh and physical solution [11, 32]. A common practice has been to use the gradient of the numerical solution, so that the mesh is concentrated in regions where the solution changes rapidly. This has proven successful, for instance, in solving a number of nontrivial reaction–diffusion, convection–diffusion, and fluid flow problems [11, 13, 15]. Nevertheless, as has often been pointed out (e.g., see Babuška and Rheinboldt [3]), a more natural and general approach than using gradients to locate the regions needing high resolution is to define the monitor functions directly in terms of error estimates. Indeed, it is common to employ a posteriori error estimates with $h$- and $p$-refinement to solve steady-state problems by finite-element methods. For time-dependent problems, it is also possible to derive error estimates; however, as evident from the analysis of Johnson and co-workers, it is much more challenging to generate efficient and reliable error estimates, due to the coupling of errors in the space and time directions—see [17, 26]. The difficulty is compounded here by the introduction of a convection term from the mesh movement, which makes classical error estimates for elliptic problems less applicable (see [33, 34]). Limited work has been done using a posteriori error estimates in the context of mesh movement for one-dimensional problems [1, 8], but to our knowledge, such strategies have not been attempted in higher dimensions. (While our concern is parabolic problems, it is worth noting that global error estimation for hyperbolic problems is also complicated by the combination of local time and space discretization errors [29, 30], although in certain cases success in solving the error estimation problem globally is achieved [22].)

The main purpose of this paper is to consider an $r$-adaptive finite-element method based on a moving-mesh PDE approach to solving parabolic PDEs, where the monitor function is defined in terms of an error indicator. The idea behind the method is straightforward: We first discretize the parabolic problem in time. At each time level, we solve elliptic equations, for which an error estimate for the numerical solution of the discretized problem

is available. This error indicator $\bar{e}(\boldsymbol{x}, t)$ is calculated by the a posteriori error estimation technique developed as in [5, 6, 16, 28]. The monitor function $G(\boldsymbol{x}, t)$ (see Section 2) is defined in terms of this error indicator, e.g., by

$$G(\boldsymbol{x}, t) = \sqrt{1 + \alpha(|\bar{e}(\boldsymbol{x}, t)|/\|\bar{e}(\cdot, t)\|_\Omega)^2}\, I,$$

where $\alpha$ is a parameter balancing the relative costs of solving the moving-mesh PDE and the physical PDE. The moving-mesh PDE is then solved to determine an updated adaptive mesh for the next time level. Finally, the physical problem is integrated to get the numerical solution at this new time level. Compared to a moving-mesh method with the monitor function defined in terms of the gradient of numerical solutions, the present approach appears to be more robust since it automatically locates the regions where higher numerical resolution is needed. In addition, this approach generally gives more accurate results.

There are some limitations to this moving-mesh approach. For one, the error indicator only takes into account the local errors arising from the spatial discretization, instead of the global error from both the space and time discretizations—our experience indicates that the strategy is most successful when these are balanced. Also, it is generally impossible to perform error control without the capability to change the number of mesh points and thereby the mesh topology. These important issues are discussed in later sections and are topics of our current research.

An outline of the paper is as follows. In Section 2 we give a brief description of the moving-mesh method based on moving-mesh PDEs. In Section 3 we introduce the general model problem to be considered and the finite-element method for moving-meshes. In Section 4 we describe the a posteriori error estimation technique for elliptic equations and construct the monitor function using the error estimate. In Section 5, some numerical experiments are presented to compare the present approach and that based on using solution gradients. Finally, Section 6 contains conclusions and remarks.

## 2. MOVING-MESH METHOD BASED ON MOVING-MESH PDEs

We assume that the underlying physical problem is defined on a simply connected open domain $\Omega \subset R^2$. After prescribing a (fixed) computational domain $\Omega_c \subset R^2$ and a corresponding mesh on it, we define a moving mesh on $\Omega$ as the image of the mesh on $\Omega_c$ through a time-dependent mapping $\boldsymbol{x} = \boldsymbol{x}(\boldsymbol{\xi}, t)$. In this sense, generating an adaptive moving mesh on $\Omega$ is equivalent to determining a time-dependent mapping $\boldsymbol{x} = \boldsymbol{x}(\boldsymbol{\xi}, t)$.

Following [25], we define $\boldsymbol{x} = \boldsymbol{x}(\boldsymbol{\xi}, t)$ as the inverse mapping of the solution $\boldsymbol{\xi} = \boldsymbol{\xi}(\boldsymbol{x}, t)$ of the parabolic equation

$$\frac{\partial \boldsymbol{\xi}}{\partial t} = \frac{1}{\tau} \nabla \cdot (G^{-1} \nabla \boldsymbol{\xi}), \tag{1}$$

supplemented with appropriate boundary and initial conditions. Here, $\tau > 0$ is a parameter used to control the smoothness of mesh movement in time, and the monitor function $G = G(\boldsymbol{x}, t)$ is a two-by-two symmetric positive definite matrix which provides control of various mesh properties, particularly mesh concentration and alignment. In general, smaller $\tau$ results in prompter mesh adaptation to changes in the monitor function, while larger $\tau$ produces slower (smoother) mesh movement in time.

In practice, it is more convenient to directly compute the mapping $\boldsymbol{x}(\boldsymbol{\xi}, t)$ instead of its inverse $\boldsymbol{\xi}(\boldsymbol{x}, t)$, because it gives explicit locations of the mesh points. Interchanging the roles of the variables $\boldsymbol{x}$ and $\boldsymbol{\xi}$, (1) can be written as [23]

$$\frac{\partial \boldsymbol{x}}{\partial t} = \frac{1}{\tau}\left(a_{11}\frac{\partial^2 \boldsymbol{x}}{\partial \xi^2} + a_{12}\frac{\partial^2 \boldsymbol{x}}{\partial \xi\, \partial \eta} + a_{22}\frac{\partial^2 \boldsymbol{x}}{\partial \eta^2} + b_1\frac{\partial \boldsymbol{x}}{\partial \xi} + b_2\frac{\partial \boldsymbol{x}}{\partial \eta}\right), \tag{2}$$

where

$$J = \det\left(\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\xi}}\right), \quad \boldsymbol{a}^1 = \frac{1}{J}\begin{bmatrix} y_\eta \\ -x_\eta \end{bmatrix}, \quad \boldsymbol{a}^2 = \frac{1}{J}\begin{bmatrix} -y_\xi \\ x_\xi \end{bmatrix},$$
$$a_{ij} = \boldsymbol{a}^i \cdot G^{-1}\boldsymbol{a}^j,$$
$$b_i = -\boldsymbol{a}^i \cdot \left(\frac{\partial G^{-1}}{\partial \xi}\boldsymbol{a}^1 + \frac{\partial G^{-1}}{\partial \eta}\boldsymbol{a}^2\right).$$

This system of nonlinear parabolic PDEs is referred to as the moving-mesh PDE [25].

The overall effect of the monitor function $G$ on the resulting generated meshes is complicated, depending on various factors such as the geometries of $\Omega$ and $\Omega_c$ and the boundary correspondence between them. Nevertheless, the eigensystem of $G$ plays a crucial descriptive role. More specifically, if $\lambda_1$ and $\lambda_2$ are the eigenvalues of $G$, and $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ are the corresponding eigenvectors, then $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ control mainly the directions of mesh concentration, while $\lambda_1$ and $\lambda_2$ determine the concentration strength along these directions. To achieve a higher mesh concentration along the $\boldsymbol{v}_1$ direction in certain regions, one needs large $\lambda_1$ in that region (see [12] for details).

Given the monitor function, the moving-mesh PDE (2) is solved numerically for $\boldsymbol{x} = \boldsymbol{x}(\boldsymbol{\xi}, t)$ in conjunction with the physical PDE. Since the positions of mesh points need not be determined very precisely, it is usually unnecessary to solve the MMPDE to high accuracy. Here, (2) is discretized with linear finite elements in space, and the resulting ODE system is integrated using a backward Euler method, with the parameter $\tau = 1$.

Once the meshes $\Omega_h(t_n)$ and $\Omega_h(t_{n+1})$ on $\Omega$ corresponding to times $t_n$ and $t_{n+1}$, respectively, are obtained, the mesh $\Omega_h(t)$ for $t \in (t_n, t_{n+1})$ is defined via linear interpolation as follows: The meshes $\Omega_h(t_{n+1})$ and $\Omega_h(t_n)$ have the same connectivities as the computational mesh $\Omega_{c,h}$, so for each element $K_c \in \Omega_{c,h}$, there exist two corresponding elements $K(t_n) \in \Omega_h(t_n)$ and $K(t_{n+1}) \in \Omega_h(t_{n+1})$. The vertices of element $K(t)$ are defined by

$$\boldsymbol{x}_i(t) = \frac{t - t_n}{t_{n+1} - t_n}\boldsymbol{x}(\boldsymbol{\xi}_i, t_{n+1}) + \frac{t_{n+1} - t}{t_{n+1} - t_n}\boldsymbol{x}(\boldsymbol{\xi}_i, t_n),$$

where $\boldsymbol{x}(\boldsymbol{\xi}, t_n)$ and $\boldsymbol{x}(\boldsymbol{\xi}, t_{n+1})$ are the approximations of the mapping $\boldsymbol{x} = \boldsymbol{x}(\boldsymbol{\xi}, t)$ at time levels $t_n$ and $t_{n+1}$, respectively, and $\{\boldsymbol{\xi}_i\}$ denotes the set of vertices of $K_c$. All elements $K(t)$ defined this way constitute the mesh $\Omega_h(t)$, which is needed for the integration of the physical PDEs with multistage integrators.

## 3. MOVING FINITE-ELEMENT APPROXIMATION OF PHYSICAL PDEs

We now describe the finite-element discretization of the physical PDE on the moving meshes. For simplicity, the description is given only for a scalar model problem, but it is straightforward to generalize it to systems of PDEs.

The model problem is

$$D(\boldsymbol{x},t)\frac{\partial u}{\partial t}=\nabla\cdot(a(\boldsymbol{x},t)\nabla u)+f(\boldsymbol{x},t,u,\nabla u),\quad \text{in } \Omega\times(t_0,T] \tag{3}$$

with boundary conditions

$$\begin{aligned} u&=d(\boldsymbol{x},t),\quad \text{on } \Gamma_D,\\ a\frac{\partial u}{\partial \vec{n}}&=g(\boldsymbol{x},t),\quad \text{on } \Gamma_N, \end{aligned} \tag{4}$$

where $D(\boldsymbol{x},t)>0$, $a(\boldsymbol{x},t)\geq a_0>0$, and $\Gamma_D$ and $\Gamma_N$ are disjoint sets whose union is $\partial\Omega$. It is assumed that there exists a unique solution $u=u(x,t)$ for given initial conditions.

For the discretization of (3), we use Rothe's approach, or the approach of horizontal method of lines. Specifically, (3) is discretized first in time and then in space. This is different from the commonly used method-of-lines approach, where the physical PDEs are discretized first in space and then in time. A main advantage of the former approach over the latter is that error estimation techniques developed for elliptic problems can be adopted and illustrated more easily. But we should also point out that the two approaches are mathematically equivalent provided that the same spatial and temporal discretization schemes are used.

With Rothe's approach, we first transform (3) from the physical coordinates to the computational ones. Let $\hat{u}(\boldsymbol{\xi},t)=u(\boldsymbol{x}(\boldsymbol{\xi},t),t)$, $\hat{D}(\boldsymbol{\xi},t)=D(\boldsymbol{x}(\boldsymbol{\xi},t),t)$, and $\hat{a}(\boldsymbol{\xi},t)=a(\boldsymbol{x}(\boldsymbol{\xi},t),t)$. By the chain rule we rewrite (3) as

$$\hat{D}\frac{\partial \hat{u}}{\partial t}=\hat{\nabla}\cdot(\hat{a}\nabla\hat{u})+f(\boldsymbol{x},t,\hat{u},\hat{\nabla}\hat{u})+\hat{D}\left(\frac{\partial \boldsymbol{x}}{\partial t}\cdot\hat{\nabla}\hat{u}\right), \tag{5}$$

where

$$\hat{\nabla}=\left(\frac{\partial\boldsymbol{\xi}}{\partial\boldsymbol{x}}\right)^T\nabla_\xi=\left[\left(\frac{\partial\boldsymbol{x}(\boldsymbol{\xi},t)}{\partial\boldsymbol{\xi}}\right)^{-1}\right]^T\nabla_\xi$$

and $\nabla_\xi$ is the gradient operator with respect to $\boldsymbol{\xi}$.

A multistage singly diagonally implicit Runge–Kutta method (SDIRK) is employed for the temporal discretization of (5) because of its high accuracy and good stability (e.g., see [20]). First, we rewrite (5) as

$$\hat{D}\frac{\partial\hat{u}}{\partial t}=F(t,\hat{u}),\quad t\in(t_0,T], \tag{6}$$

where

$$F(t,\hat{u})=\hat{\nabla}\cdot(\hat{a}\nabla\hat{u})+f(\boldsymbol{x},t,\hat{u},\hat{\nabla}\hat{u})+\hat{D}\left(\frac{\partial\boldsymbol{x}}{\partial t}\cdot\hat{\nabla}\hat{u}\right).$$

Let $t_0<t_1<\cdots<t_N=T$ be a partition of $[t_0,T]$, let $\delta t_n=t_{n+1}-t_n$, and let $\hat{u}^{(n)}(\boldsymbol{\xi})$ be an approximation of $\hat{u}(\boldsymbol{\xi},t_n)$. Applying the $s$-stage SDIRK to (6), we have

$$\begin{cases} \hat{D}(t_{n,i})\hat{k}_i=F\big(t_{n,i},\hat{u}^{(n)}+\delta t_n\sum_{j=1}^{i-1}a_{ij}\hat{k}_j+\gamma\delta t_n\hat{k}_i\big),\quad 1\leq i\leq s,\\ \hat{u}^{(n+1)}=\hat{u}^{(n)}+\delta t_n\sum_{i=1}^{s}b_i\hat{k}_i, \end{cases} \tag{7}$$

where $t_{n,i} = t_n + c_i \delta t_n$ and $a_{ij}$, $b_i$, $c_i$ $(1 \le i \le s, 1 \le j < i)$, and $\gamma$ are scheme constants. Introducing

$$\hat{v}_i = \hat{u}^{(n)} + \delta t_n \sum_{j=1}^{i-1} a_{ij} \hat{k}_j, \quad \hat{u}_i = \hat{v}_i + \gamma \delta t_n \hat{k}_i,$$

the $i$th stage equation can be simplified to

$$\hat{D}(t_{n,i}) \frac{\hat{u}_i - \hat{v}_i}{\gamma \delta t_n} = F(t_{n,i}, \hat{u}_i). \tag{8}$$

Letting $u_i(\boldsymbol{x}) = \hat{u}_i(\boldsymbol{\xi}(\boldsymbol{x}, t_{n,i}))$ and $v_i(\boldsymbol{x}) = \hat{v}_i(\boldsymbol{\xi}(\boldsymbol{x}, t_{n,i}))$, we transform (8) back into the physical domain and obtain

$$\begin{aligned} D(\boldsymbol{x}, t_{n,i}) \frac{u_i - v_i}{\gamma \delta t_n} &= \nabla \cdot (a(\boldsymbol{x}, t_{n,i}) \nabla u_i) + f(\boldsymbol{x}, t_{n,i}, u_i, \nabla u_i) \\ &\quad + D(\boldsymbol{x}, t_{n,i}) \left( \frac{\partial \boldsymbol{x}}{\partial t}(\boldsymbol{\xi}(\boldsymbol{x}, t_{n,i}), t_{n,i}) \cdot \nabla u_i \right). \end{aligned} \tag{9}$$

This is a second-order elliptic equation. The boundary conditions for $u_i$ can be readily obtained from (4) as

$$\begin{cases} u_i = d(\boldsymbol{x}, t_{n,i}), & \text{on } \Gamma_D, \\ a(\boldsymbol{x}, t_{n,i}) \dfrac{\partial u_i}{\partial \vec{n}} = g(\boldsymbol{x}, t_{n,i}), & \text{on } \Gamma_N. \end{cases} \tag{10}$$

After finding $u_i$, we compute $\hat{u}_i = u_i(\boldsymbol{x}(\boldsymbol{\xi}, t_{n,i}), t_{n,i})$ and $\hat{k}_i = (\hat{u}_i - \hat{v}_i)/(\gamma \delta t_n)$. The approximate solution $\hat{u}^{(n+1)}$ at $t_{n+1}$ is then obtained after all $\hat{k}_i$ $(1 \le i \le s)$ have been calculated.

It remains to describe the finite-element discretization for (9) supplemented with (10). To simplify notation, we will omit writing the dependence on $t_{n,i}$ in functions $D$, $a$, and $\frac{\partial \boldsymbol{x}}{\partial t}$. Let $H_D^1(\Omega)$ be the subspace of $H^1(\Omega)$ whose elements vanish on $\Gamma_D$. Taking the $L^2(\Omega)$-inner product of (9) with test function $\phi \in H_D^1(\Omega)$, we obtain the weak formulation

$$\mathcal{A}(u_i, \phi) = 0, \quad \forall \phi \in H_D^1(\Omega), \tag{11}$$

where

$$\begin{aligned} \mathcal{A}(u, \phi) = \int_\Omega \Bigg[ \left( D \frac{u - v_i}{\gamma \delta t_n} - D \frac{\partial \boldsymbol{x}}{\partial t} \cdot \nabla u - f(\boldsymbol{x}, t, u, \nabla u) \right) \phi \\ + a \nabla u \cdot \nabla \phi \Bigg] d\boldsymbol{x} - \int_{\Gamma_N} g \phi \, d\Gamma. \end{aligned} \tag{12}$$

Recall that $\Omega_h(t_{n,i})$ is the mesh at time $t_{n,i}$ defined by linear interpolation between $\Omega_h(t_n)$ and $\Omega_h(t_{n+1})$. We denote the standard element by $\hat{K}$ (viz., the unit square for quadrilateral elements and the unit triangle for triangular elements) and an arbitrary element in $\Omega_h(t_{n,i})$ by $K$. Let $F_K$ be the mapping from $\hat{K}$ onto $K$. Then the approximation subspace based on mesh $\Omega_h(t_{n,i})$ can be described as

$$\mathcal{S}^h(t_{n,i}) = \{ v \in H^1(\Omega) \mid v|_K \circ F_K \in P(\hat{K}), \forall K \in \Omega_h(t_{n,i}) \},$$

where $P(\hat{K})$ is a given set of polynomials on $\hat{K}$. In our applications, we choose $P(\hat{K})$ as the set of linear functions; i.e., we use only linear elements.

Let $\mathcal{S}_D^h(t_{n,i}) = \mathcal{S}^h(t_{n,i}) \cap H_D^1(\Omega)$. Then the finite-element approximation $u_{h,i} \in \mathcal{S}^h(t_{n,i})$ of the solution $u_i$ of (9) is required to satisfy the Dirichlet boundary conditions in (10) and

$$\mathcal{A}(u_{h,i}, \phi) = 0, \quad \forall \phi \in \mathcal{S}_D^h(t_{n,i}). \tag{13}$$

The system of nonlinear algebraic equations is solved by Newton's iteration, with the resulting linear systems solved by BiCGStab2 [19], preconditioned with an incomplete LU decomposition.

For problems with varying time scales, $\delta t_n$ should be selected dynamically. This is achieved with a standard approach as follows: Assume that the $s$-stage SDIRK method (7) is of order $p$. Let $\bar{b}_i (1 \le i \le s)$ be a set of parameters of an embedded $q$th-order method associated with scheme (7) (e.g., see [20]). Let $\bar{p} = \min(p, q)$. Then $\delta t_{n+1}$ is chosen to satisfy

$$\delta t_{n+1} = \delta t_n \min\left(2, \max\left(0.1, 0.8\left(atol \Big/ \left\| \sum_{i=1}^{s} (b_i - \bar{b}_i)\hat{k}_{h,i} \right\|_{\ell^2}\right)^{1/(\bar{p}+1)}\right)\right), \tag{14}$$

where *atol* is a prescribed error tolerance, $\|\cdot\|_{\ell^2}$ is the vector $\ell^2$-norm, and $\hat{k}_{h,i}$ is the $i$th stage function value corresponding to the spatially discretized version of (7).

## 4. A POSTERIORI ERROR ESTIMATION

In this section, we present in more details our strategy for obtaining the error estimates and the monitor function. In the integration of the time-dependent PDEs, there are two main types of errors, local and global. The local errors result from the spatial and temporal discretizations of the underlying PDEs, while the global error measures the accumulation of these effects, i.e., the actual difference between the exact solution and the numerical solution. In general, it is very difficult to estimate the global error for the parabolic PDEs even if spatial errors are ignored since it depends on the (problem dependent) accumulated effects of these local errors during the numerical integration. General numerical ODE integrators only attempt to control local errors, with the assumption that the corresponding global errors do not grow prohibitively. Similarly, our strategy will be to control the spatial local error with mesh adaptation and the temporal local error with time step-size selection. The successes and limitations of the approach are discussed in Section 5.

To obtain an error indicator for the space discretization, we use the type III error estimation technique developed by Bank and co-workers [5, 6] for elliptic problems (and independently by Oden and co-workers as the implicit element residual method [16, 28]). The analysis in [5] cannot be applied directly to (9) since the diffusion coefficient is proportional to the time step size. Nevertheless, a recent study of steady-state reaction–diffusion and convection–diffusion problems by Verfürth [33, 34] has shown that the estimated error obtained with a similar method is of the same magnitude in the energy norm as the real one, with a factor depending weakly on the diffusion coefficient. On the basis of this result, we expect that

the error estimator developed in [5] will provide a reasonably accurate local error indicator which in turn can be used for mesh movement.

Let $u_{h,i}$ be the finite-element solution in (13) and the local error be $e_i = u_i - u_{h,i}$. Define the gradient operator $A$ of $\mathcal{A}$ as

$$A(w,\phi) = \frac{\delta \mathcal{A}}{\delta u}(u_{h,i},\phi)w = \int_\Omega \left[ \left( D\frac{w}{\gamma \delta t_n} - D\frac{\partial \boldsymbol{x}}{\partial t}\cdot\nabla w - \frac{\partial f}{\partial u}(\boldsymbol{x},t,u_{h,i},\nabla u_{h,i})w \right.\right.$$
$$\left.\left. - \frac{\partial f}{\partial \nabla u}(\boldsymbol{x},t,u_{h,i},\nabla u_{h,i})\nabla w \right)\phi + a\nabla w\cdot\nabla\phi \right] d\boldsymbol{x}.$$

Then $e_i \in H_D^1(\Omega)$ satisfies

$$A(e_i,\phi) \approx \mathcal{A}(u_i,\phi) - \mathcal{A}(u_{h,i},\phi) = -\mathcal{A}(u_{h,i},\phi). \tag{15}$$

Following [5], we determine an easily computable local error indicator $\check{e}_i$ approximating $e_i$. For this purpose, we first introduce some notation. For each element $K \in \Omega_h(t_{n,i})$, let $(\cdot,\cdot)_K$ be the $L^2$-inner product over $K$, $\mathcal{S}_K^Q$ be the space of functions which are the pullbacks of quadratic polynomials in $\hat{K}$ under the mapping from $\hat{K}$ to $K$, and

$$\check{\mathcal{S}}_K = \left\{ v \in \mathcal{S}_K^Q \;\middle|\; v = 0 \text{ at the vertices of } K \right\}.$$

For any element side $s$ of the mesh $\Omega_h(t_{n,i})$, we also denote by $\langle\cdot,\cdot\rangle_s$ the $L^2$-inner product over $s$. Denote by $E_I$ the set of interior element sides in $\Omega_h(t_{n,i})$, and by $E_N$ the set of boundary sides on $\Gamma_N$. Let $\vec{n}$ denote one of the unit normal vectors to $s$ for $s \in E_I$ and the outward unit normal vector to $s$ for $s \in E_N$. Further, for any $s \in E_I$, let $[v]_s$ denote the jump of $v$ across $s$ along the $\vec{n}$ direction. It is not difficult to see that when $a$ is continuous, the jump $[a\frac{\partial u}{\partial \vec{n}}]_s$ is independent of the orientation of $\vec{n}$. Let

$$\begin{aligned} r &= f(t_{n,i},\boldsymbol{x},u_{h,i},\nabla u_{h,i}) - D\frac{u_{h,i}-v_{h,i}}{\gamma\delta t_n} + D\frac{\partial \boldsymbol{x}}{\partial t}\cdot\nabla u_{h,i} - \nabla(a\nabla u_{h,i}), \\ r_b &= a\frac{\partial u_{h,i}}{\partial \vec{n}} - g. \end{aligned} \tag{16}$$

It follows that

$$\mathcal{A}(u_{h,i},\phi) = -(r,\phi) - \langle r_b,\phi\rangle_{\Gamma_N} - \sum_{K\in\Omega_h(t_{n,i})}\sum_{s\in\partial K\cap E_I}\left\langle \left[a\frac{\partial u_{h,i}}{\partial \vec{n}}\right]_s,\phi\right\rangle_s. \tag{17}$$

The local error indicator $\check{e}_i$ is defined piecewise in $\Omega_h(t_{n,i})$ such that $\check{e}|_K \in \check{\mathcal{S}}_K$ satisfies

$$A_K(\check{e}_i,\phi) = (r,\phi)_K + \sum_{s\in\partial K\cap E_N}\langle r_b,\phi\rangle_s + \frac{1}{2}\sum_{s\in\partial K\cap E_I}\left\langle \left[a\frac{\partial u_{h,i}}{\partial \vec{n}}\right]_s,\phi\right\rangle_s, \quad \forall\phi\in\check{\mathcal{S}}_K. \tag{18}$$

Note that $A_K(\cdot,\cdot)$ is similar to $A(\cdot,\cdot)$ except that the integration is taken only over the element $K$. For each element, the above equation contains either three or four unknowns (for triangles and quadrilaterals, respectively) associated with the midpoints of the element sides.

Note that the integration from $t_n$ to $t_{n+1}$ involves $s$ steady-state equations only. To reduce the overhead cost of error estimation, we apply the above procedure to one of these stages. If $\delta t_n$ is small, or if the coefficients $D$, $a$, and $f$ do not change much for different stages, then all $\check{e}_i$'s will be close to each other. However, since this error indicator is used to calculate the adaptive mesh for the following time step, it is preferable to use the one for the time closest to the next time step, i.e., for the last stage. Moreover, if the SDIRK scheme is stiffly accurate, we have $\hat{u}^{n+1} = \hat{u}_s$. In other words, the numerical solution in the last stage is the solution at the new time level [20]. Thus, we calculate $\check{e}_s$ for the last stage value $\hat{u}_s$.

To construct the monitor function for mesh movement, we first calculate the energy norm of the error function $\check{e}_s$ over each element; i.e., we define a piecewise constant function $\bar{e}(\cdot, t_n)$ by

$$\bar{e}(\boldsymbol{x}, t_n) = [(D(t_{n,i})\check{e}_s, \check{e}_s)_K + \gamma \delta t_n (a \nabla \check{e}_s, \nabla \check{e}_s)_K]^{1/2}, \quad \forall \boldsymbol{x} \in K. \tag{19}$$

The monitor function is defined as

$$G(\boldsymbol{x}, t_n) = \sqrt{1 + \alpha (\bar{e}(\boldsymbol{x}, t_n)/\|\bar{e}(\cdot, t_n)\|_\Omega)^2}\, I, \tag{20}$$

where $\|\bar{e}(\cdot, t_n)\|_\Omega^2 = \sum_K (\bar{e}(t_n), \bar{e}(t_n))_K$ so that $\|\bar{e}\|_\Omega$ is the energy norm over $\Omega$ of the error indicator $\check{e}$, $I$ is the two-by-two identity matrix, and $\alpha$ is an intensity parameter used to emphasize or deemphasize the influence of the error function on the mesh concentration. For larger $\alpha$ the mesh distribution is more closely influenced by $\bar{e}(t_n)$, which generally results in more computational effort being expended in solving the MMPDEs. Smaller $\alpha$ gives less variation in $G$, resulting in less mesh adaptation.

For comparison, we also use a monitor function defined using the gradient of the numerical solutions. Although not as commonly used or recommended for $h$-refinement [3], gradients have always been a popular choice for moving-mesh methods due to their simplicity and the relative sensitivity of moving-mesh equation to the use of higher derivative terms in the monitor function [9]. Specifically, the monitor function is defined as

$$G = \sqrt{1 + \alpha_g (|\nabla u_h|/\|\nabla u_h\|_\Omega)^2}\, I, \tag{21}$$

where $\nabla u_h$ is the gradient of the numerical solution, $\|\nabla u_h\|_\Omega^2 = \sum_K (\nabla u_h, \nabla u_h)_K$, and $\alpha_g$ is a parameter controlling the influence of the gradient on the mesh concentration. Larger $\alpha_g$ produces stronger mesh concentration in regions of large $|\nabla u_h|$ and requires more computational effort in solving the MMPDEs.

The purpose of scaling by the $L^2$-norm of $\bar{e}(t)$ or $\nabla u_h(t)$ in defining the monitor functions in (20) and (21) is to make choosing $\alpha$ and $\alpha_g$ easier and general. This treatment is similar to the one used in [7, 8] for one-dimensional problems, where the argument is made that under suitable conditions the control parameters can be optimally chosen. While in general the optimal choice of $\alpha$ and $\alpha_g$ is clearly problem dependent, the numerical solutions obtained with our moving-mesh method are relatively insensitive to them, and we see from our numerical experiments in Section 5 that taking $\alpha$ and $\alpha_g$ in the range of 1 to 100 usually produces a reasonable balance between the costs in solving MMPDEs and physical PDEs. So while the choice of $\alpha$ is not insignificant, it is a secondary effect and does not qualitatively alter the comparison between (20) and (21).

As a common practice with moving-mesh methods based on MMPDEs, the monitor function $G(t_n)$ is smoothed. We use a simple smoothing method of local averaging. More

precisely, for a nonnegative integer $M$, the monitor function $G^{(M)}(t_n) = G^{(M)}(\boldsymbol{x}, t_n)$ is a piecewise linear polynomial with the value at any grid point $\boldsymbol{x}$ defined as

$$G^{(m+1)}(\boldsymbol{x}, t_n) = \frac{1}{|\omega(\boldsymbol{x})|} \int_{\omega(\boldsymbol{x})} G^{(m)}(\mathbf{y}, t_n)\, d\mathbf{y}, \quad \text{for } m = 0, 1, \ldots, M-1, \tag{22}$$

where $\omega(\boldsymbol{x})$ is the union of the elements having $\boldsymbol{x}$ as a vertex and $|\omega(\boldsymbol{x})|$ is its area. The starting value is $G^{(0)} = G(t_n)$. In our computation, we take $M = 6$, for which experience has shown that the approach performs well [13, 23].

## 5. NUMERICAL EXAMPLES

In this section, we present some numerical results obtained with the $r$-adaptive finite-element method which uses the error indicator developed in the previous sections. The examples are selected to demonstrate the feasibility of the method, especially in predicting the location of large solution error regions.

In our computations, a two-stage second-order SDIRK scheme is used for time integration. The corresponding embedded scheme is of first order. The parameters are [2]

$$\gamma = (2 - \sqrt{2})/2, \quad a_{21} = 1 - \gamma, \quad c_1 = \gamma, \quad c_2 = 1,$$
$$b_1 = 1 - \gamma, \quad b_2 = \gamma, \quad \hat{b}_1 = 1, \quad \hat{b}_2 = 0.$$

EXAMPLE 1. Our first example involves the linear parabolic equation

$$\frac{\partial u}{\partial t} = \nabla^2 u + f(t, \boldsymbol{x}) \tag{23}$$

defined on the unit square $(0, 1) \times (0, 1)$. The right-hand side $f(t, \boldsymbol{x})$ and the initial and the Dirichlet boundary conditions are chosen so that there is an exact solution,

$$u(t, \boldsymbol{x}) = \tanh\left[15\left(x - \frac{1}{2}\right)\right] \tanh\left[15\left(y - \frac{1}{2}\right)\right].$$

This time-independent solution is chosen so that reliability of the error estimation procedure can be verified. This simple model problem is also used to compare the performances of the moving-mesh methods based on the error indicator and solution gradients.

An initial $40 \times 40$ mesh with uniform rectangular elements is used in all the computations. The problem is integrated with a fixed step size 0.01 until $t = 1$, at which time the change in the numerical solution $u_h$ between two subsequent time steps is below $10^{-6}$ in the $L^2$-norm.

We first examine the error indicator on a fixed mesh. Surface plots of the energy norm distribution for the true error $u(t) - u_h(t)$ and the error indicator $\check{e}(t)$ at $t = 1$ are displayed in Fig. 1. The energy norms over $\Omega$ for $u(t) - u_h(t)$ and for $\check{e}(t)$ at $t = 1$ are $3.096 \times 10^{-2}$ and $2.139 \times 10^{-2}$, respectively. Although the magnitude of the estimated local error differs from that of the true global error, $\check{e}(t)$ locates very well the regions of large global error where high resolution is most needed.

Next, we test the moving-mesh techniques based on the error indicator in (20) and gradient function in (21). The maximum norm, $L^2$-norm, and energy norm of $u(t) - u_h(t)$ at $t = 1$ are summarized in Table I for solutions obtained using different values of the intensity parameters $\alpha$ and $\alpha_g$. From Table I, one can see that for the moving-mesh method based

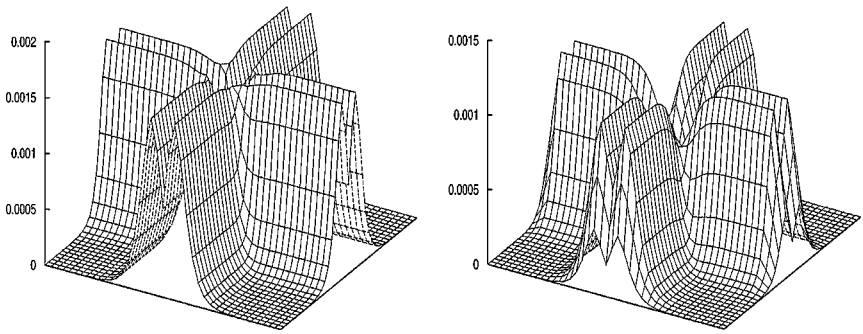


**FIG. 1.** Example 1: Energy norm distribution of true error $u(t) - u_h(t)$ (left) and error indicator $\check{e}(t)$ (right) for the solution on a fixed mesh at $t = 1$.

on the error indicator, larger $\alpha$ results in better mesh adaptation and smaller errors. For the moving-mesh method based on gradients, this is not quite true, and large $\alpha_g$ may even result in larger errors.

Due to the relative simplicity of the problem, the numerical solutions obtained with moving-meshes are only slightly more accurate than those obtained with a fixed mesh having the same number of elements, and indeed, the adaptive algorithm may not even bring about improved efficiency, but for more challenging problems a fixed-mesh computation can be prohibitively expensive or completely unrealistic [13, 15]. Also, it is possible to substantially reduce the overhead in solving MMPDEs, e.g., by using two-level grids [23]. The point here is that the approach using the error indicator gives qualitative improvement over that using gradients.

In Fig. 2 we plot the adaptive mesh at $t = 1$ for the cases $\alpha = 50$ and $\alpha_g = 50$, and in Fig. 3 we plot the true error. Using the monitor function based on the error indicator, the regions of large error are correctly located, and the mesh points are appropriately concentrated. This is in contrast to the adaptive mesh obtained with the monitor function based on solution gradients, where the concentration or adaptation does not always occur in the regions of large solution errors. As a consequence, the numerical accuracy may not improve since more points are taken away from the regions needing higher resolution. Indeed, when $\alpha_g$ is increased from 50 to 500, the pointwise error of the numerical solution increases; see Table I.

**TABLE I**

**Norms of the Error $u - u_h$ at $t = 1$**

| | $\|u - u_h\|_\infty$ | $\|u - u_h\|_{L^2}$ | $\|u - u_h\|_e$ |
|---|---|---|---|
| Fixed mesh | 1.30768*e*-02 | 4.32257*e*-03 | 3.09652*e*-02 |
| Moving-mesh monitor (20) | | | |
| $\alpha = 10$ | 4.87727*e*-03 | 1.62814*e*-03 | 1.86536*e*-02 |
| $\alpha = 50$ | 4.22896*e*-03 | 1.37534*e*-03 | 1.68747*e*-02 |
| $\alpha = 100$ | 4.11860*e*-03 | 1.33783*e*-03 | 1.65705*e*-02 |
| $\alpha = 500$ | 4.02447*e*-03 | 1.30868*e*-03 | 1.63113*e*-02 |
| Moving-mesh monitor (21) | | | |
| $\alpha_g = 10$ | 5.04127*e*-03 | 2.14485*e*-03 | 2.00205*e*-02 |
| $\alpha_g = 50$ | 4.98980*e*-03 | 2.10109*e*-03 | 1.82722*e*-02 |
| $\alpha_g = 100$ | 5.12903*e*-03 | 2.16794*e*-03 | 1.78881*e*-02 |
| $\alpha_g = 500$ | 5.43788*e*-03 | 2.45696*e*-03 | 1.76697*e*-02 |

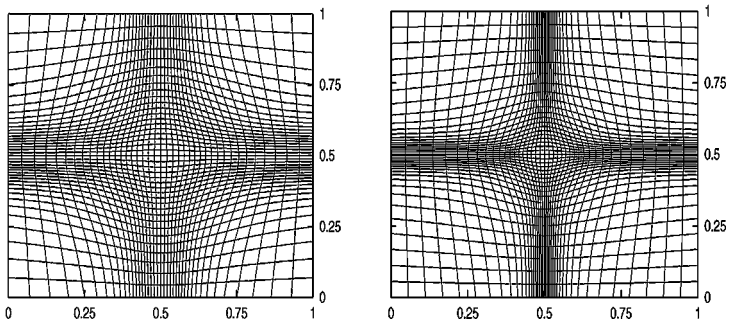


**FIG. 2.** Example 1: Adaptive meshes obtained with monitor function based on error indicator $\check{e}(t)$ (left) and gradient $|\nabla u_h|$ (right).

Fortunately, for many problems, regions with large gradients are adjacent to those where large higher order solution derivatives occur and the numerical solution has poorer accuracy. By smoothing the monitor functions based on gradients, the higher mesh concentration regions often overlap with these regions of large errors. This helps explain why in most applications moving-mesh methods based on solution gradients are able to effectively improve the solution accuracy and consequently are often used.

Finally, we note that while the solution accuracy is not overly sensitive to the choices of these parameters $\alpha$ or $\alpha_g$, one should not choose excessively large values, since the computational work in solving the moving-mesh PDEs will increase accordingly. In our experience, values between 1 and 100 usually produce good balance between the quality of mesh concentration and the cost of solving the MMPDEs.

EXAMPLE 2. The second example is the well-known Burgers equation

$$\frac{\partial u}{\partial t} = \nu \nabla^2 u - u u_x - u u_y, \quad \text{in } \Omega \times (0.25, 1.25], \tag{24}$$

where $\Omega$ is the unit square $(0, 1) \times (0, 1)$. The initial and the Dirichlet boundary conditions are chosen such that the exact solution is

$$u(\boldsymbol{x}, t) = [1 + e^{(x+y-t)/(2\nu)}]^{-1}. \tag{25}$$

We consider the case with a moderately small diffusion coefficient $\nu = 0.005$.

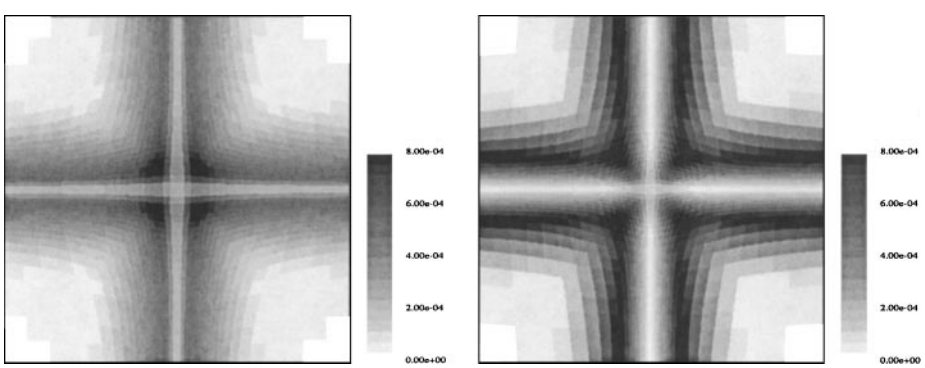


**FIG. 3.** Example 1: Contour plots of energy norm distribution of true error $u(t) - u_h(t)$ at $t = 1$ for solutions obtained with monitor functions based on error indicator (left) and solution gradient (right).

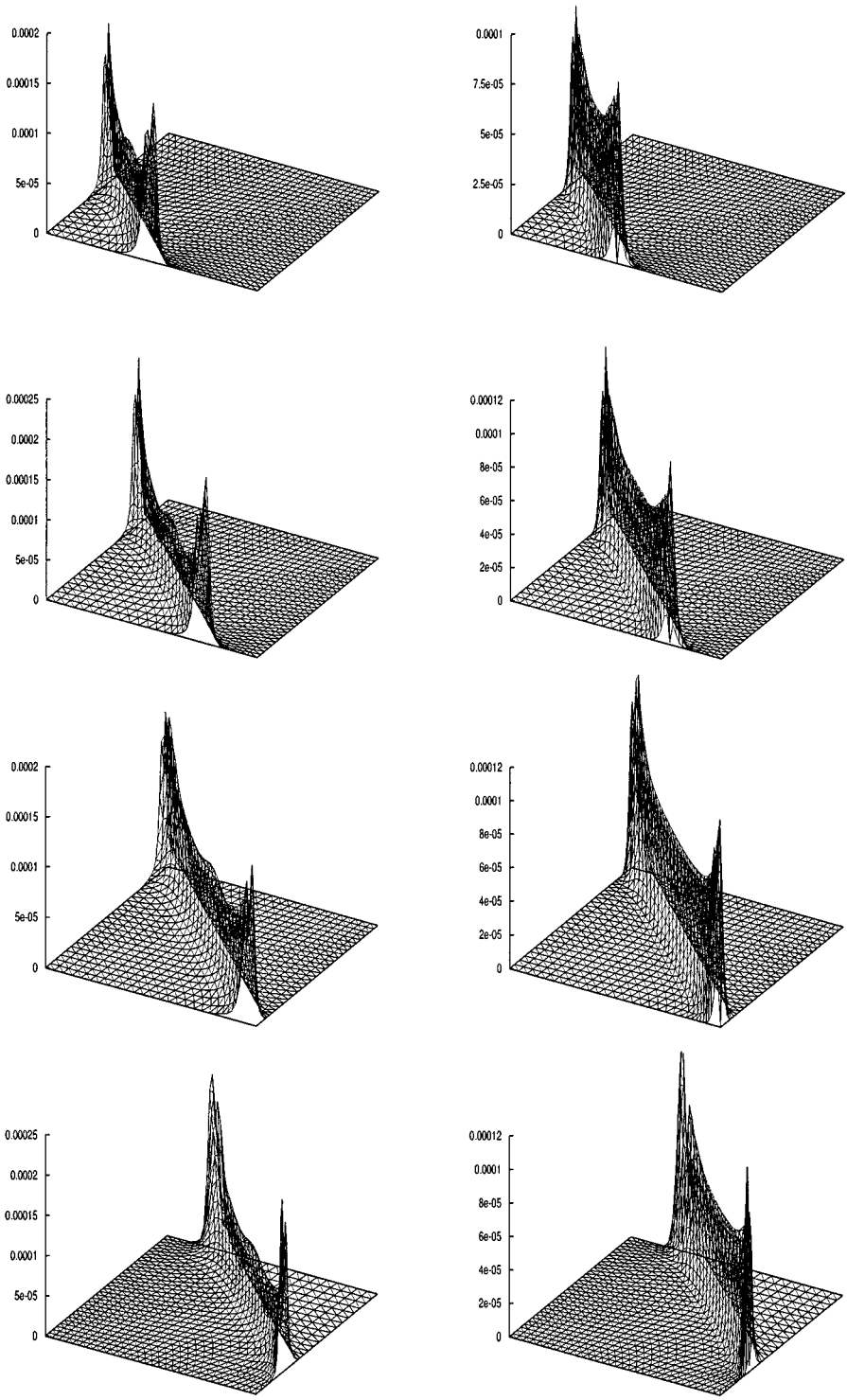


**FIG. 4.** Example 2: Energy norm distribution of true error $u(u) - u_h(t)$ (left) and error indicator $\check{e}(t)$ (right) at $t = 0.5, 0.75, 1.0, 1.25$ (from top to bottom).

An adaptive initial mesh consisting of 2048 triangular elements is used in this example. The time step size is fixed at $\delta t = 0.01$. The parameter $\alpha = 50$ in (20).

The energy norm distribution of the true error $u(t) - u_h(t)$ and the local error indicator $\check{e}(t)$ are displayed in Fig. 4 for four different times. Note that $\check{e}(t)$ indicates the regions where the true error is large and higher mesh concentration is needed. Figure 5 shows how the mesh is correctly concentrated in regions with correspondingly large errors.

For comparison, we also solve this problem using a corresponding fixed uniform mesh and a moving mesh obtained with monitor function (21) based on the gradient of the numerical solution (with $\alpha_g = 50$). The energy norms of the solution errors are plotted in the left diagram of Fig. 7. The solution based on moving meshes obtained using an error indicator is better than that using the solution gradient, while both are more accurate than the solution obtained on a fixed uniform mesh (though not substantially, as for Example 1 the problem is reasonably easy). To examine the difference between the two mesh adaptation cases, in Fig. 6 we magnify the mesh and the error indicator $\check{e}(t)$ around the midpoint of the physical domain. The figure again confirms the observation made in Example 1: the monitor function based on the local error indicator more accurately pinpoints the locations of regions needing higher resolution than that based on the solution gradient.

For this calculation the mesh lines are aligned with the direction of the wave front, although this is not a major factor in the success of the moving-mesh method. To demonstrate this point, we tested the problem with the same parameter setting using mesh triangles oriented with the hypotenuse at an angle 45°, which is orthogonal to the direction of the wave front of the solution (25). The right diagram of Fig. 7 displays the energy norm of

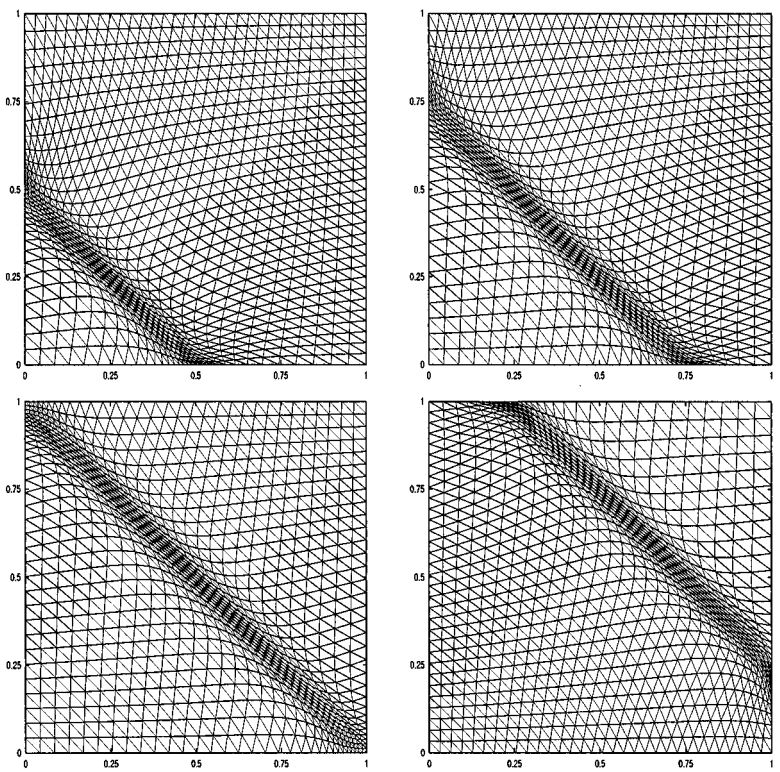


**FIG. 5.** Example 2: Moving-mesh based on error indicator at $t = 0.5, 0.75, 1.0, 1.25$.

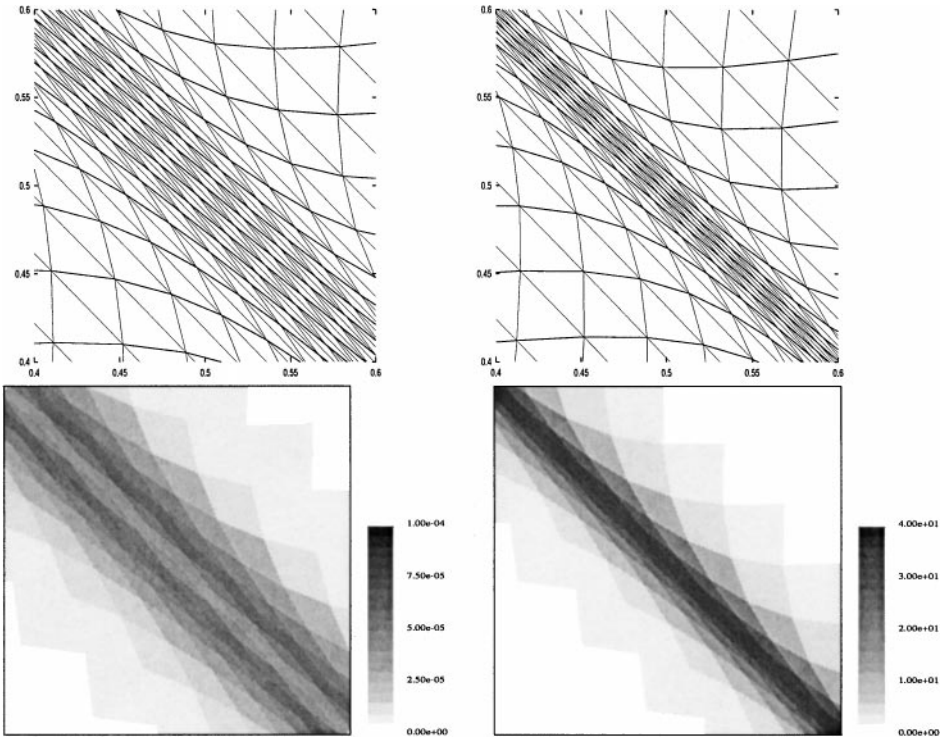


**FIG. 6.** Example 2: Top: Closeup of moving meshes at $t = 1$ obtained with monitor function based on error indicator (left) and gradient of numerical solution (right). Bottom: Closeup of corresponding $\overline{e}(t)$ (left) and $|\nabla u_h(t)|$ (right).

the true error $u(t) - u_h(t)$ using the fixed-mesh and the two moving-mesh methods. The relative improvement in accuracy using the moving-mesh methods is similar to that in the earlier case, and the meshes aligned with the wave front produce somewhat more accurate solutions than those that are not aligned. So, while this is not a focus of our comparison, in principle one may strive to improve the mesh alignment, e.g., by edge swapping, to produce better results.

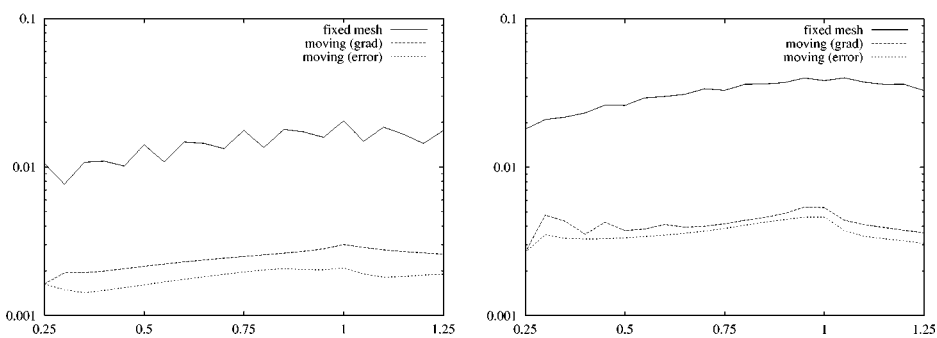


**FIG. 7.** Example 2: Energy norm of error $u(t) - u_h(t)$ obtained with moving meshes and fixed mesh. Left: triangular mesh with hypotenuse oriented at 135°. Right: Triangular mesh with hypotenuse oriented at 45°.

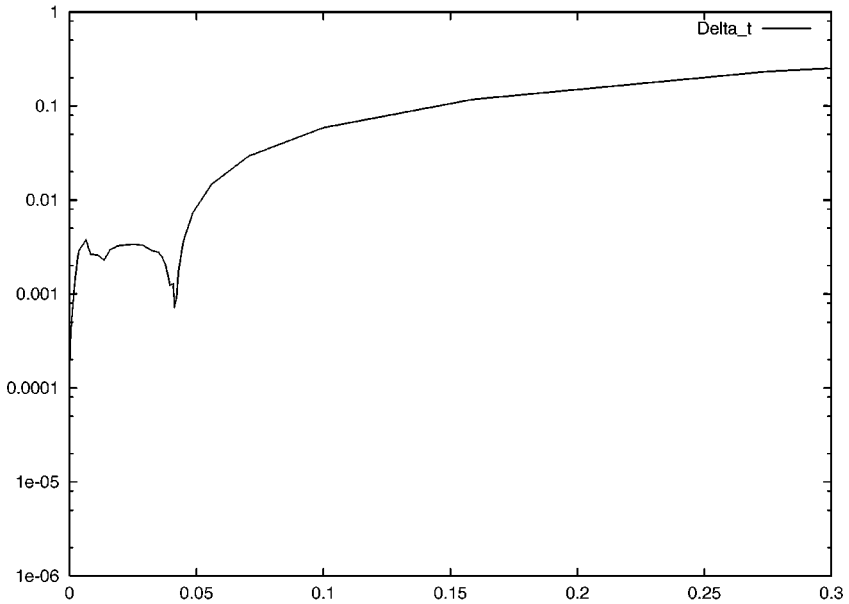


**FIG. 8.** Time step sizes $\delta t$ selected with (14) for Example 3.

EXAMPLE 3. The third example is a nonlinear reaction–diffusion equation

$$\frac{\partial u}{\partial t} = \nabla^2 u + \frac{1}{\epsilon} u(1 - u^2) \tag{26}$$

defined on $\Omega = (0, 1) \times (0, 1)$. We choose $\epsilon = 10^{-3}$ as in [18]. On all of the boundary segments, homogeneous Neumann conditions are imposed for all time.

The initial conditions are

$$u(0, \boldsymbol{x}) = \begin{cases} 1, & \text{if } \left(x - \frac{1}{3}\right)\left(x - \frac{2}{3}\right)\left(y - \frac{1}{3}\right)\left(y - \frac{2}{3}\right) > 0, \\ -1, & \text{otherwise .} \end{cases}$$

This problem was used by Erikson and Johnson in [18] to test their adaptive method based on local refinement and for a posteriori error estimation. The solution is very sensitive to perturbations. Indeed, small perturbations at the cross points $(\frac{i}{3}, \frac{j}{3})$, $i, j = 1$ or 2, may lead to different solution paths, and in [18] a nonsymmetric solution develops because of the nonsymmetric local refinement.

We solve this problem by the moving-mesh method based on the error indicator with $\alpha = 5$ in (20). The time integration is implemented with the same SDIRK method as in the previous examples but with variable step size. The initial time step size is chosen as $\delta t_0 = 10^{-5}$, and later time step sizes are selected by (14) with an error tolerance $atol = 10^{-3}$. See Fig. 8 for a plot of the step sizes selected by this scheme. The problem is symmetric with respect to the diagonal lines as well as the horizontal and vertical lines passing through $(\frac{1}{2}, \frac{1}{2})$. To preserve this symmetry, we use a uniform initial triangular mesh obtained by inserting both diagonals to the elements of a $40 \times 40$ uniform rectangular mesh.

Figure 9 displays the numerical solution at four different times. The corresponding moving mesh and the contour plot of the energy norm distribution of $\check{e}(t)$ are plotted in Fig. 10. Note that the moving mesh conforms to the regions with large error distribution, and the symmetry in the solution pattern is preserved with our $r$-adaptive strategy.

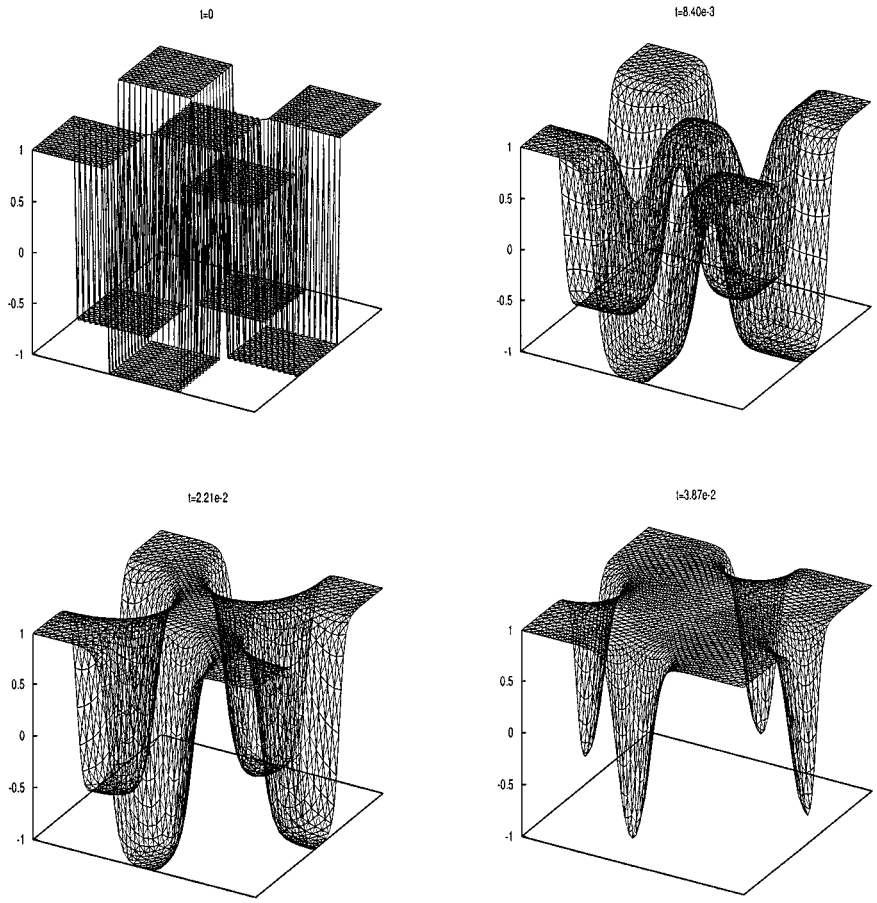


**FIG. 9.** Example 3: Solution $u(t)$ at four time instants, $t = 0, 0.00840, 0.0221, 0.0387$.

EXAMPLE 4. Finally, we consider a coupled nonlinear reaction–diffusion system modeling a combustion process [1, 25],

$$\begin{cases} \dfrac{\partial u}{\partial t} - \nabla^2 u = -\dfrac{R}{\alpha\delta} u e^{\delta(1-1/T)}, \\ \dfrac{\partial T}{\partial t} - \dfrac{1}{Le}\nabla^2 T = \dfrac{R}{\delta Le} u e^{\delta(1-1/T)}, \end{cases}$$

where $u$ and $T$ represent, respectively, the dimensionless species concentration and the temperature of a chemical which is undergoing a one-step reaction. The physical domain is $\Omega = (-1, 1) \times (-1, 1)$. The initial and boundary conditions are

$$\begin{aligned} u|_{t=0} = T|_{t=0} = 1, \quad & \text{in } \Omega, \\ u|_{\partial\Omega} = T|_{\partial\Omega} = 1, \quad & \text{for } t > 0, \end{aligned} \tag{27}$$

and the physical parameters are set to $Le = 0.9, \alpha = 1, \delta = 20$, and $R = 5$.

This problem has several interesting features; e.g., the temperature $T$ rises from 1 to approximately $1 + \alpha$ at the center of $\Omega$ in a very short period of time and the solutions $T$ and $u$ have sharp wave fronts moving toward the boundary $\partial\Omega$. These make adaptive

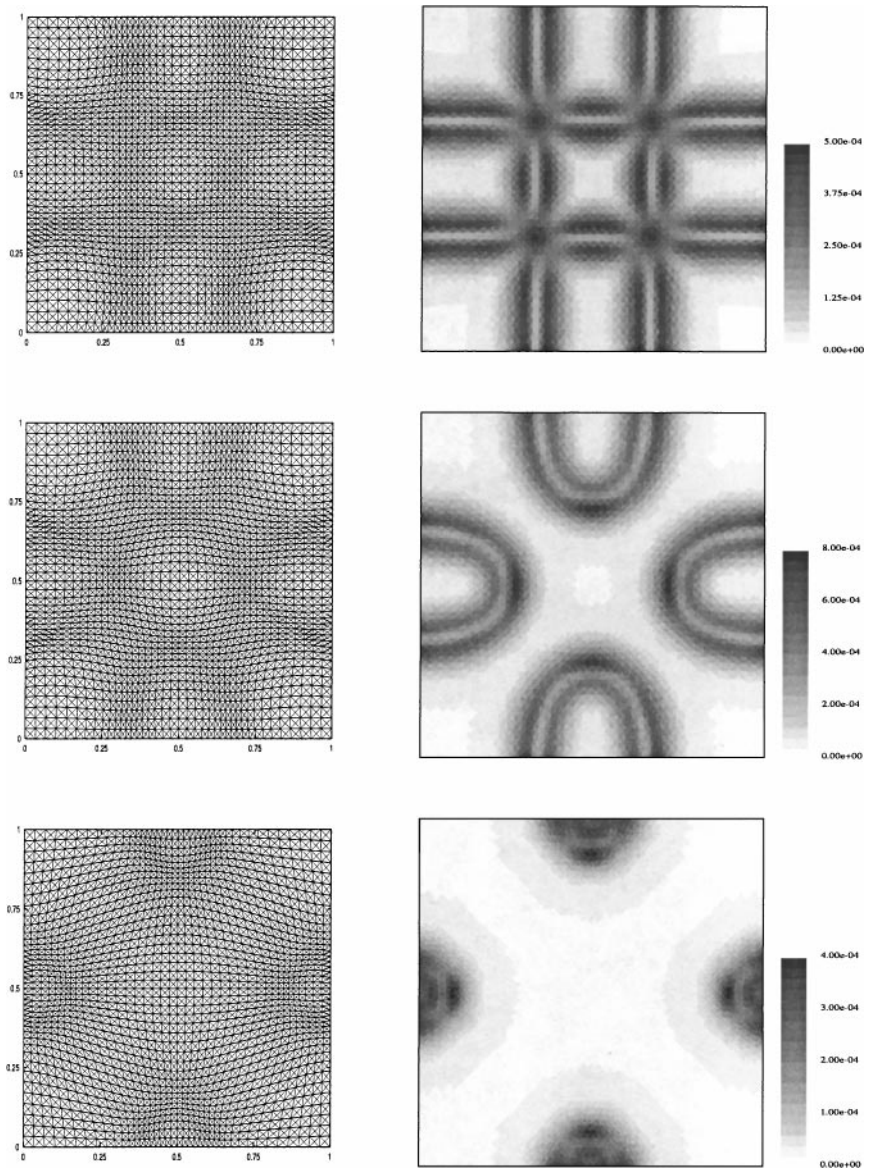


**FIG. 10.** Example 3: Moving mesh and contour plot of energy norm distribution of error estimator $\check{e}(t)$ at $t = 0.00840, 0.0221, 0.0387$ (from top to bottom).

methods (in both the spatial and temporal directions) crucial for accurate simulation of the physical process.

For this problem, we use the same initial mesh as in Example 3. The time integration uses a variable step size determined by $atol = 10^{-5}$ and $\delta t_0 = 10^{-4}$. In defining the monitor function, we take $\alpha = 50$ in (20).

Figure 11 displays the moving mesh and the solution $T$ at four different times. The resulting step size plotted in Fig. 12 illustrates the importance of a variable time step size selection strategy to efficiently solve this type of problem. Once again, this monitor function performs somewhat better than the gradient monitor function (21), although we do not give

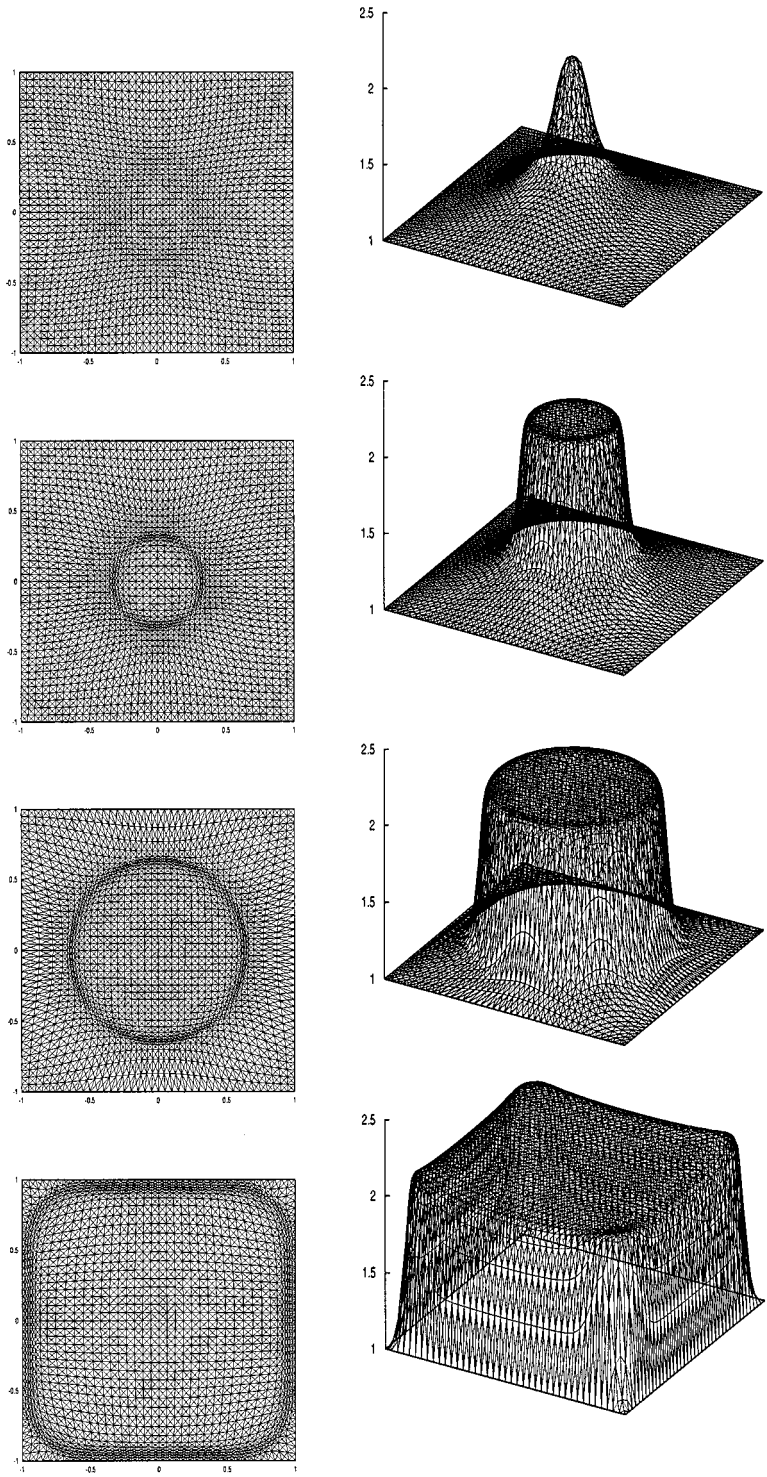

**FIG. 11.** Example 4: Moving mesh and temperature $T$ at $t = 0.257, 0.262, 0.270, 0.288$.

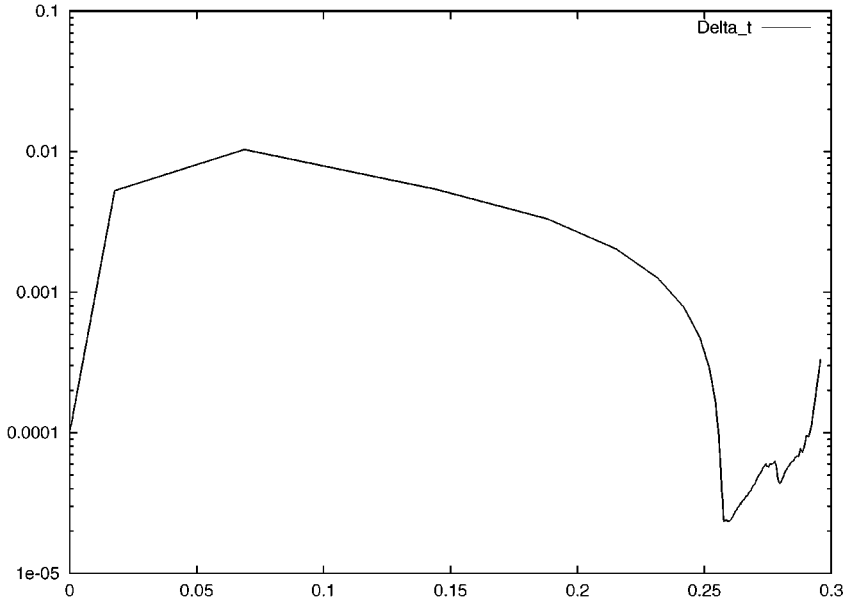


**FIG. 12.** Time step sizes $\delta t$ selected with (14) for Example 4.

detailed results here. Also, for this problem, using no spatial adaptivity would necessitate a much finer mesh to achieve comparable accuracy.

## 6. CONCLUSIONS AND REMARKS

We have presented an $r$-adaptive finite-element method based on moving-mesh PDEs and an error indicator for solving parabolic problems. The basic idea behind the method is to define the monitor function for mesh movement as a function of an a posteriori estimate of the local spatial approximation error. The estimation is done by applying a technique developed in [5, 6, 16, 28] for elliptic problems (which result here from temporal discretization of the underlying physical PDEs). The numerical results demonstrate that the error indicator accurately predicts the regions of large solution variation. Comparison between monitor functions based on the error indicator and on solution gradients has been made. The numerical results show that while the method based on solution gradients is simpler and easier to implement, the one based on an error indicator more accurately pinpoints regions needing higher mesh concentration and is generally more robust. Some guidelines in choosing the parameter in the monitor function definition are provided.

It is worth pointing out that the error indicator used here for mesh movement is only an approximation to the local spatial discretization error at a given time. This local approximation can give a reasonable indication of the magnitude of the true error where it is largest and more mesh concentration is needed; in our experience this approximation tends to be less reliable when the spatial and temporal discretization errors are of substantially different size. To better understand this, it would be desirable to extend the analysis of Verfürth [33, 34] to elliptic PDEs of the form (9) having convection terms due to the mesh movement. For at the end of the day, the ability to estimate global errors for mesh movement algorithms depends on estimating both the temporal and spatial discretization errors and understanding how they can accumulate.

While the robustness of moving-mesh methods based on MMPDEs has been established [13, 15, 23], several limitations warrant future investigation. First, as a fundamental feature of the $r$-adaptive finite-element method, the number of grid points is fixed. The $r$-adaptive method seeks in principle the optimal mesh within the given mesh topology. Thus, an error estimator may provide accurate relative distribution of mesh points, but it is generally impossible to keep the error below a certain magnitude without changing the topology of the meshes. Second, the adaptive approach used here attempts to minimize the errors for time-dependent problems using tools developed for steady-state problems. In other words, the errors in the spatial and temporal directions are treated separately. So while this approach is simple and reasonably robust, in its present form it is generally not able to provide good global error control. Achieving this requires varying the number of grid points and time step sizes. Along these lines, we are in the process of developing an algorithm which will incorporate the techniques presented here as well as the features of both $h$- and $r$-methods.

## ACKNOWLEDGMENTS

This work was supported in part by NSERC (Canada) Grant OGP-0008781 and NSF (USA) Grant DMS-9626107.

## REFERENCES

1. S. Adjerid and J. E. Flaherty, A moving finite element method with error estimation and refinement for one-dimensional time dependent partial differential equations, *SIAM J. Numer. Anal.* **23**, 778 (1986).
2. R. Alexander, Diagonally implicit Runge–Kutta methods for stiff O.D.E.'s, *SIAM J. Numer. Anal.* **14**, 1006 (1977).
3. I. Babuška and W. C. Rheinboldt, Adaptive approaches and reliability estimations in finite element analysis, *Comput. Methods Appl. Mech. Eng.* **18**, 519 (1979).
4. M. J. Baines, *Moving Finite Elements* (Oxford Univ. Press, Oxford, 1994).
5. R. E. Bank and A. Weiser, Some a posteriori error estimators for elliptic differential equations, *Math. Comput.* **44**, 283 (1985).
6. R. E. Bank and K. Smith, A posteriori error estimates based on hierarchical bases, *SIAM J. Numer. Anal.* **30**, 921 (1993).
7. G. Beckett and J. A. Mackenzie, On a uniform accurate finite difference approximation of a singularly perturbed reaction–diffusion problem using grid equidistribution, *J. Comput. Appl. Math.*, to appear.
8. G. Beckett, J. A. Mackenzie, A. Ramage, and D. M. Sloan, On the numerical solution of one-dimensional PDEs using adaptive methods based on equidistribution, submitted for publication.
9. J. G. Blom and J. G. Verwer, *On the use of the Arclength and Curvature Monitor in a Moving-grid Method Which is Based on the Method of Lines*, Report NM-N8902, CWI, Amsterdam (1989).
10. C. de Boor, *Good Approximation by Splines with Variable Knots II*, Springer lecture Notes Series 363 (Springer-Verlag, Berlin, 1973).
11. J. U. Brackbill, An adaptive grid with direction control, *J. Comput. Phys.* **108**, 38 (1993).
12. W. Cao, W. Huang, and R. D. Russell, A study of monitor functions for two dimensional adaptive mesh generation, *SIAM J. Sci. Comput.* **20**, 1978 (1999).
13. W. Cao, W. Huang, and R. D. Russell, An $r$-adaptive finite element method based upon moving-mesh PDEs, *J. Comput. Phys.* **149**, 221 (1999).
14. N. Carlson and K. Miller, Design and application of a gradient-weighted moving finite element code. II. In two dimensions, *SIAM J. Sci. Comput.* **19**, 766 (1998).
15. H. Ceniceros and T. Hou, An efficient dynamically adaptive method for potentially singular solutions, *J. Comput. Phys.*, to appear.

16. L. F. Demkowicz, J. T. Oden, and T. Stroubolis, Adaptive finite elements for flow problems with moving boundaries. I. Variational principles and a posterior estimates, *Comput. Methods Appl. Mech. Eng.* **46**, 217 (1984).
17. K. Eriksson, D. Estep, P. Hansbo, and C. Johnson, Introduction to adaptive methods for differential equations, in *Acta Numerica* (Cambridge Univ. Press, Cambridge, 1995), pp. 105–158.
18. K. Eriksson and C. Johnson, Adaptive finite element methods for parabolic problems V. long-time integration, *SIAM J. Numer. Anal.* **32**, 1743 (1995).
19. M. H. Gutknecht, Variants of BICGSTAB for matrices with complex spectrum, *SIAM J. Sci. Comput.* **14**, 1020 (1993).
20. E. Hairer and G. Wanner, *Solving Ordinary Differential Equations* (Springer-Verlag, Berlin, 1987), Vols. I and II.
21. 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 PDEs, *J. Comput. Phys.* **95**, 254 (1991).
22. P. Houston, J. Mackenzie, E. Suli, and G. Warnecke, A posteriori error analysis for numerical approximations of Friedrichs systems, *Numer. Math.* **82**, 433 (1999).
23. W. Huang, Practical aspects of formulation and solution of moving mesh partial differential equations, *J. Comput. Phys.*, to appear.
24. W. Huang, Y. Ren, and R. D. Russell, Moving mesh partial differential equations (MMPDEs) based upon the equidistribution principle, *SIAM J. Numer. Anal.* **31**, 709 (1994).
25. W. Huang and R. D. Russell, Moving mesh strategy based upon a gradient flow equation for two dimensional problems, *SIAM J. Sci. Comput.* **20**, 998 (1999).
26. C. Johnson, Adaptive finite element methods for diffusion and convection problems, *Comput. Methods Appl. Mech. Eng.* **82**, 301 (1990).
27. K. Miller, Moving finite elements II, *SIAM J. Numer. Anal.* **18**, 1033 (1981).
28. J. T. Oden, L. F. Demkowicz, T. Stroubolis, and P. Devloo, Adaptive methods for problems in solid and fluid mechanics, in *Accuracy Estimates and Adaptive Refinements in Finite Element Computations*, edited by I. Babuška, O. C. Zienkiewicz, J. Gago, and E. R. de A. Oliveira (Wiley, Chichester, 1986), pp. 249–280.
29. J. T. Oden, L. F. Demkowicz, W. Rachowicz, and T. Westerman, A posteriori error analysis in finite elements: The element residual method for symmetrizable problems with applications to compressible Euler and Navier–Stokes equations, *Comput. Methods Appl. Mech. Eng.* **82**, 183 (1990).
30. A. Safjan, L. F. Demkowicz, and J. T. Oden, Adaptive finite element methods for hyperbolic systems with application to transient acoustics, *Int. J. Numer. Methods Eng.* **32**, 677 (1991).
31. B. Semper and G. Liao, A moving grid finite-element method using grid deformation, *Numer. Meth. Partial Differential Equations* **11**, 603 (1995).
32. J. F. Thompson, Z. A. Warsi, and C. W. Mastin, *Numerical Grid Generation* (North-Holland, New York, 1985).
33. R. Verfürth, A posteriori error estimators for a singularly perturbed reaction–diffusion equation, *Numer. Math.* **78**, 479 (1998).
34. R. Verfürth, A posteriori error estimators for convection–diffusion equation, *Numer. Math.* **80**, 641 (1998).