SIAM J. SCI. COMPUT.
Vol. 20, No. 3, pp. 998–1015

MOVING MESH STRATEGY BASED ON A GRADIENT FLOW EQUATION FOR TWO-DIMENSIONAL PROBLEMS*

WEIZHANG HUANG† AND ROBERT D. RUSSELL‡

Abstract. In this paper we introduce a moving mesh method for solving PDEs in two dimensions. It can be viewed as a higher-dimensional generalization of the moving mesh PDE (MMPDE) strategy developed in our previous work for one-dimensional problems [W. Huang, Y. Ren, and R. D. Russell, *SIAM J. Numer. Anal.*, 31 (1994), pp. 709–730]. The MMPDE is derived from a gradient flow equation which arises using a mesh adaptation functional in turn motivated from the theory of harmonic maps. Geometrical interpretations are given for the gradient equation and functional, and basic properties of this MMPDE are discussed. Numerical examples are presented where the method is used both for mesh generation and for solving time-dependent PDEs. The results demonstrate the potential of the mesh movement strategy to concentrate the mesh points so as to adapt to special problem features and to also preserve a suitable level of mesh orthogonality.

Key words. moving meshes, mesh adaptation, gradient equation

AMS subject classifications. 65M50, 65M20, 35K05

PII. S1064827596315242

1. Introduction.

During the last decade, mesh adaptation has become an indispensable tool in the numerical solution of partial differential equations (PDEs). This is especially true in areas such as fluid dynamics, hydraulics, combustion, and heat transfer. For problems in these areas, small node separations are often required over a portion of the physical domain to resolve large solution variations there. Numerical solution of these problems using uniform meshes is formidable, even with the use of supercomputers when the systems involve two or more spatial dimensions, since the number of mesh nodes required in each dimension becomes very large. On the other hand, it has been amply demonstrated that significant improvements in accuracy and efficiency can be gained by adapting a smaller number of mesh nodes during the solution process so that they remain concentrated locally about regions of large solution variations (e.g., see [6, 20, 30, 36]). Yet while mesh adaptation is now an established area of intense research activity in many areas of science and engineering, it is also a rapidly changing one in need of clarification of its basic principles.

Roughly speaking, there are two kinds of adaptive methods, static and dynamic. In a static or local refinement method, mesh points are moved at fixed time levels, where points can also be added to or removed from the mesh in order to obtain an approximate solution with the desired level of accuracy. The local refinement method is conceptually easy to apply and usually reliable, requiring little tuning of parameters. It also allows an easier error analysis than a dynamic method. Many local refinement methods have been developed, such as in [2, 5, 17]. The principal disadvantages of the local refinement method are its complicated data structure and its

*Received by the editors June 1, 1996; accepted for publication (in revised form) September 12, 1997; published electronically January 5, 1999. This work was supported by the Natural Science and Engineering Research Council of Canada grant OGP-0008781, the University of Kansas General Research Fund 3427-20-0038, NSF grant 9626107, and EPSCoR grant OSR-9255223.
http://www.siam.org/journals/sisc/20-3/31524.html
†Department of Mathematics, University of Kansas, Lawrence, KS 66045 (huang@math.ukans.edu).
‡Department of Mathematics and Statistics, Simon Fraser University, Burnaby, BC V5A 1S6, Canada (rdr@cs.sfu.ca).

being forced to either use a one-step time integrator or continually restart a multistep method. Expertise is required in the programming of a local refinement method, and considerable technical effort is necessary to treat the complicated communications between elements in the same as well as different levels of refinement.

For a dynamic or moving mesh method, the mesh points move continuously in the space-time domain and concentrate in regions where the solution is steep. An explicit rule, called a moving mesh equation, is designed to do this. It is often very difficult to formulate a moving mesh strategy which can efficiently perform the mesh adaptation while moving the mesh in an orderly fashion. Producing an analysis of the error is also difficult because the mesh movement is strongly coupled (in a highly nonlinear fashion) to the physical PDE. However, when working properly, a moving mesh method usually requires considerably fewer mesh points than a static method to produce commensurate accuracy and allows significantly larger time steps without causing instability. The moving mesh equation can often be easily designed to mimic basic properties of the underlying physical PDE, and as a result, the qualitative behavior of the solution can be efficiently captured (e.g., see [9, 10]). Moreover, a moving mesh method which generates a structured mesh is straightforward to combine with the existing structured CFD codes. We anticipate that its ease of programming will allow the moving mesh strategy to find widespread acceptance in the science and engineering communities.

Our aim is to present a new mesh movement strategy for use in two spatial dimensions. Although a number of one-dimensional moving mesh methods have been developed in the last 15 years, such as those in [1, 6, 15, 26, 32, 33], very few have been applied in two spatial dimensions. The moving finite element method (MFE) of Miller and Miller [32] can in theory be used in any dimension. One disadvantage, however, is that certain sensitive penalty functions are necessary in order to avoid singularity of the mass matrix. Nevertheless, this method bears further study in two and higher dimensions, and in particular the recent works by Carlson and Miller [12] and Baines [6] have provided a better understanding of the MFE. The method in [21] determines the mesh speed by directly differentiating the steady variational mesh equation in [8] with respect to time, but it is necessary to periodically enforce satisfaction of the original mesh equation to combat a tendency of the node concentration to relax.

The space-time finite elements recently developed in [3, 28] are other methods warranting special attention. For these, the finite element formulation corresponding to the physical PDE is expressed over its space-time domain, allowing deformation of the spatial domain with respect to time to be automatically taken into account. A major issue has been how to update the mesh as the domain changes shape. Several strategies for the mesh movement have been considered, viz., use of the equation of linear elasticity in [28]. However, these mesh movement strategies have had difficulties controlling the deformation of elements, and a remeshing procedure must be used when this deformation becomes too large. More generally, it has been unclear how mesh adaptation should be combined with these mesh movement strategies.

Recently, we have developed a so-called *moving mesh PDE (MMPDE)* approach (see [22, 23, 24, 25]). In one dimension, the approach has led to a unifying framework describing the previous methods, provided a new theoretical underpinning, and generated very reliable methods. Here we develop a new moving mesh method along the lines of the MMPDE approach in [24] but which can be applied in two dimensions. With this approach, a continuous moving mesh equation which explicitly involves the mesh speed is employed to move the mesh points in such a way that they are con-

centrated in regions where the physical solution is steep. Since the mesh movement obeys a time-dependent partial differential equation (the MMPDE), this approach facilitates a better understanding of moving mesh methods, a comparison of their basic properties, and ultimately the development of useful new algorithms.

The fundamental idea is to assume that the steady mesh point distribution is determined by a steady mesh generator and then obtain an MMPDE by introducing mesh speeds into the mesh generator. We describe how to obtain this MMPDE using the gradient flow equation in the next section. The approach can, under suitable conditions, be used to derive our original one-dimensional MMPDEs introduced in [24], but it also applies readily for two dimensions. In section 3, we lay the groundwork for the new class of moving mesh methods by introducing a functional for mesh adaptation and orthogonality and directional control which is motivated from the theory of harmonic maps. A geometrical interpretation of the functional is also given. In section 4, the basic MMPDE is derived using the gradient flow equation method and the functional developed in sections 2 and 3. Several two-dimensional numerical examples are presented in section 5, followed by conclusions and comments in section 6. Finally, an appendix is given for the brief definition of harmonic maps between Riemannian manifolds and a few related existence results.

2. Gradient flow equation. Consider the problem of generating a moving mesh for a simply connected, two-dimensional physical domain Ω_p. It is useful to regard Ω_p and a computational (logical) domain Ω_c as images of one another under the invertible map

$$x = x(\xi, \eta, t), \quad y = y(\xi, \eta, t) \tag{1}$$

and its inverse

$$\xi = \xi(x, y, t), \quad \eta = \eta(x, y, t), \tag{2}$$

where $\vec{x} = (x, y)^T \in \Omega_p$ and $\vec{\xi} = (\xi, \eta)^T \in \Omega_c$ are the physical and computational coordinates, and t is time. A mesh covering Ω_p is obtained by partitioning Ω_c (often uniformly) and then mapping it to the physical domain through (1). In this sense, generating a mesh is mathematically equivalent to generating an invertible continuous coordinate transformation. Hereafter, the terms mesh, map, and coordinate transformation will be used synonymously.

With the MMPDE approach developed in [24, 25], the time-dependent coordinate transformation is determined by solving a parabolic PDE called a moving mesh PDE or MMPDE. It explicitly involves the mesh speed and continuously moves the mesh in such a way that the mesh points are concentrated in regions where the physical solution is steep. This MMPDE is obtained by introducing mesh speed into a steady mesh generation equation. In this section, we will give a geometric illustration of the MMPDE approach and generalize the one-dimensional case studied in [24, 25] to two dimensions.

Suppose that there is a steady mesh generation strategy formulated as the variational problem of minimizing $I[\xi, \eta]$ over a certain function space. Since the fastest descent direction of $I[\xi, \eta]$ in the function space is the opposite direction of the first-order functional derivative $\frac{\delta I}{\delta \xi}$ (e.g., see [13]), the gradient flow equation

$$\frac{\partial \xi}{\partial t} = -\frac{\delta I}{\delta \xi}, \quad \frac{\partial \eta}{\partial t} = -\frac{\delta I}{\delta \eta} \tag{3}$$

defines a flow which converges to the equilibrium state as $t \to \infty$. In practice, it is often more suitable to use different descent directions than the fastest one and to allow the user to adjust the time scale of the mesh equation. Hence, we define our moving mesh equation as

$$\frac{\partial \xi}{\partial t} = -\frac{P}{\tau}\frac{\delta I}{\delta \xi}, \quad \frac{\partial \eta}{\partial t} = -\frac{P}{\tau}\frac{\delta I}{\delta \eta}, \tag{4}$$

where $\tau > 0$ is a user-prescribed parameter which controls the time scale, and P is an operator on the underlying function space which is chosen to have positive spectrum.

It is interesting to consider the one-dimensional analogue of (4). In one dimension, De Boor's equidistribution principle, which has been a popular tool for mesh generation and adaptation, is used to form the steady mesh generator in [24]. The idea behind this principle is to choose the coordinate transformation by equidistributing *a monitor function* $G(x) > 0$ which provides some measure of the computational difficulty in the solution of the physical PDE. (Although for simplicity we assume here that the monitor is time independent, the MMPDE approach is generally used for time-dependent monitor functions; e.g., see [25].) The equidistribution equation can be obtained by minimizing

$$I[\xi] = \frac{1}{2}\int_0^1 \frac{1}{G}\left(\frac{\partial \xi}{\partial x}\right)^2 dx, \tag{5}$$

where for convenience we assume that both the physical and computational domains are the unit interval. The Euler–Lagrange equation for (5) is

$$\frac{\delta I}{\delta \xi} \equiv -\frac{\partial}{\partial x}\left(\frac{1}{G}\frac{\partial \xi}{\partial x}\right) = 0. \tag{6}$$

Taking $P = \left(\frac{G}{\xi_x}\right)^2 I$, where I is the identity operator and $\xi_x = \frac{\partial \xi}{\partial x}$, we obtain the one-dimensional MMPDE

$$\frac{\partial \xi}{\partial t} = \frac{G^2}{\tau \xi_x^2}\frac{\partial}{\partial x}\left(\frac{1}{G}\frac{\partial \xi}{\partial x}\right), \tag{7}$$

or after exchanging the roles of dependent and independent variables,

$$\frac{\partial x}{\partial t} = \frac{1}{\tau}\frac{\partial}{\partial \xi}\left(G\frac{\partial x}{\partial \xi}\right), \tag{8}$$

which is exactly MMPDE 5 developed in [24]. Similarly, MMPDEs 3, 4, and 6 in [24] can be obtained by defining appropriate operators for P.

Obviously, the above general procedure works in any dimension provided that a functional for steady mesh adaptation is given. However, the extension of equidistribution to two dimensions is neither straightforward nor unique. In addition to adaptation and smoothness, control of other mesh properties such as skewness and overlap must be incorporated into high-dimensional formulations. In fact, a number of two-dimensional generalizations of equidistribution have been developed with varying degrees of success in the last decade by researchers using the variational approach for mesh generation and adaptation (e.g., see [8, 30, 36]). In the following section, we present a variational approach for obtaining a steady mesh generation differential equation, and an MMPDE is then obtained from it using a gradient flow equation.

3. Steady mesh generation equation. We define our mesh adaptation functional in a general form as

$$I[\xi,\eta]=\frac{1}{2}\int_{\Omega_p} dxdy\left[\nabla\xi^T G_1^{-1}\nabla\xi+\nabla\eta^T G_2^{-1}\nabla\eta\right], \tag{9}$$

where G_1 and G_2 are given symmetric positive definite matrices (called *monitor functions*) and $\nabla=(\frac{\partial}{\partial x},\frac{\partial}{\partial y})^T$. The Euler–Lagrange equation for $I[\xi,\eta]$,

$$\nabla\cdot\left(G_1^{-1}\nabla\xi\right)=0,\quad \nabla\cdot\left(G_2^{-1}\nabla\eta\right)=0, \tag{10}$$

determines the coordinate transformation for the mesh generation and adaptation. Choosing appropriate monitor functions is a key to the variational mesh adaptation approach.

A major reason for defining our mesh adaptation functional in the form (9) is its close relation to harmonic maps (defined in the appendix). In fact, it is easy to see that (9) defines an energy functional when

$$G_1=G_2=\frac{1}{\sqrt{g}}G, \tag{11}$$

where G is a symmetric positive definite matrix and $g=\det(G)$. The corresponding Euler–Lagrange equation (10), together with appropriate boundary conditions such as Dirichlet boundary conditions, define a harmonic map $\xi=\xi(x,y)$, $\eta=\eta(x,y)$: $\Omega_p\to\Omega_c$. The existence and uniqueness of the harmonic map is guaranteed provided that the boundary of Ω_c is convex. Further, the two-dimensional harmonic map defined by (10) and (11) is a homotopy equivalent to a diffeomorphism [34]. It is unclear whether or not these existence results extend to the general case where G_1 and G_2 differ, but in our limited experience little difficulty has been encountered in finding a numerical solution of the equation using smooth monitor functions.

Dvinsky [16] is apparently the first to note the practical importance of the fact that no constraints on the metric tensor $G=(g_{ij})$ are necessary to guarantee invertibility of the harmonic map and that such a map defined from the Euler–Lagrange equation can be used specifically for mesh adaptation. This metric tensor can thus be chosen so that mesh points concentrate in regions of rapid variation of the physical solution. A mesh so obtained enjoys desirable properties of harmonic maps, particularly regularity, or smoothness. In our formulation (9), the monitor functions Dvinsky uses satisfy (11) with

$$G=I+\frac{f(F)}{\|\nabla F\|^2}\nabla F\cdot\nabla F^T, \tag{12}$$

where $f(F)$ is a function of the distance from a given point to the given curve $F(x,y)=0$ such that $f(F)$ increases as the distance decreases and goes to zero as the distance increases.

Brackbill [7] is motivated by Dvinsky's work and Winslow's earlier work on mesh generation [37]. He also incorporates an efficient directional control into the mesh adaptation, thereby improving both the accuracy and efficiency of the procedure. The monitor functions he employs can be expressed in the form

$$\begin{aligned} G_1&=w\left[(1-\gamma)I+\gamma S(\vec{v}^1)\right]^{-1},\\ G_2&=w\left[(1-\gamma)I+\gamma S(\vec{v}^2)\right]^{-1}, \end{aligned} \tag{13}$$

where w is a mesh concentration function, γ is a user-defined dimensionless parameter, and the symmetric matrix $S(\vec{v})$ is defined by

$$S(\vec{v}) = \frac{1}{\|\vec{v}\|^2} \begin{bmatrix} v_2^2 & -v_1 v_2 \\ -v_1 v_2 & v_1^2 \end{bmatrix} \tag{14}$$

for any nonzero vector $\vec{v} = (v_1, v_2)^T$. The corresponding mesh functional is

$$\begin{aligned} I[\xi, \eta] = {} & \frac{\gamma}{2} \int_{\Omega_p} dxdy \frac{1}{w} \left(\|\nabla \xi\|^2 + \|\nabla \eta\|^2 \right) \\ & + \frac{(1-\gamma)}{2} \int_{\Omega_p} dxdy \frac{1}{w} \left(\frac{\|\nabla \xi \times \vec{v}^1\|^2}{\|\vec{v}^1\|^2} + \frac{\|\nabla \eta \times \vec{v}^2\|^2}{\|\vec{v}^2\|^2} \right). \end{aligned} \tag{15}$$

The first and second terms serve, respectively, to control mesh adaptation and the level of alignment of the normals to mesh surfaces of constant ξ and η with corresponding vector fields $\vec{v}^1$ and $\vec{v}^2$ which are chosen for their problem-dependent physical significance.

Motivated by the work of Dvinsky and Brackbill, we define the monitor functions as

$$\begin{aligned} & G_1 = \frac{1}{\sqrt{\tilde{g}_1}} \tilde{G}_1, \quad G_2 = \frac{1}{\sqrt{\tilde{g}_2}} \tilde{G}_2, \\ & \tilde{G}_1 = \left[(1-\gamma_1-\gamma_2) G^{-1} + \frac{\gamma_1}{2} \|G^{-1}\|_F S(\nabla \tilde{\xi}) + \frac{\gamma_2}{2} \|G^{-1}\|_F S(\vec{v}^1) \right]^{-1}, \\ & \tilde{G}_2 = \left[(1-\gamma_1-\gamma_2) G^{-1} + \frac{\gamma_1}{2} \|G^{-1}\|_F S(\nabla \tilde{\eta}) + \frac{\gamma_2}{2} \|G^{-1}\|_F S(\vec{v}^2) \right]^{-1}, \end{aligned} \tag{16}$$

where $\sqrt{\tilde{g}_1} = \det(\tilde{G}_1)$, $\sqrt{\tilde{g}_2} = \det(\tilde{G}_2)$, G is a symmetric positive definite matrix used for mesh adaptation (an arclength-like monitor function $G = I + \nabla u \ \nabla u^T$ is used in our computation), $\|G^{-1}\|_F$ is the Frobenius norm of G^{-1}, $\tilde{\xi} = \tilde{\xi}(x, y)$ and $\tilde{\eta} = \tilde{\eta}(x, y)$ define a given reference mesh used for orthogonality control, $\vec{v}^1 = \vec{v}^1(x, y)$ and $\vec{v}^2 = \vec{v}^2(x, y)$ are prescribed vector fields used for flow directional control, and γ_1 and γ_2 ($\gamma_1 + \gamma_2 < 1$) are two user-defined dimensionless parameters. It is not difficult to see from (13) and (15) that the functional (9) with the monitor functions (16) provides a mechanism to control the level of alignment of the normals of the mesh surfaces of constant ξ and η both with the prescribed vector fields $\vec{v}^1$ and $\vec{v}^2$ and with vector fields $\nabla \tilde{\xi}$ and $\nabla \tilde{\eta}$ defined by the reference mesh. In many applications the reference mesh can be chosen to be orthogonal or nearly so, and the control of the level of mesh alignment with it can be interpreted as the control of the mesh orthogonality or skewness.

It is instructive to give a different geometric interpretation of the functional $I[\xi, \eta]$ in (9). Noting that

$$\begin{aligned} \nabla \xi^T G_1^{-1} \nabla \xi &= \frac{1}{J^2 g_1} \left(\vec{x}_\eta^T G_1 \vec{x}_\eta \right), \\ \nabla \eta^T G_2^{-1} \nabla \eta &= \frac{1}{J^2 g_2} \left(\vec{x}_\xi^T G_2 \vec{x}_\xi \right), \end{aligned} \tag{17}$$

where $J = x_\xi y_\eta - x_\eta y_\xi$ is the Jacobian of the coordinate transformation, $g_1 = \det(G_1)$,

and $g_2 = \det(G_2)$, the functional $I[\xi, \eta]$ can be transformed into

$$\tilde{I}[x, y] = \frac{1}{2} \int_{\Omega_c} d\xi d\eta \left\{ \frac{1}{Jg_1} \vec{x}_\eta^T G_1 \vec{x}_\eta + \frac{1}{Jg_2} \vec{x}_\xi^T G_2 \vec{x}_\xi \right\}. \tag{18}$$

From the calculus of variations, it is known that minimizing $I[\xi, \eta]$ with respect to ξ and η as functions of x and y is equivalent to minimizing $\tilde{I}[x, y]$ with respect to x and y as functions of ξ and η provided that both coordinate transformations exist and are invertible. For the arclength-like monitor function

$$G_1 = G_2 = \frac{1}{\sqrt{1 + \|\nabla u\|^2}} \left(I + \nabla u \nabla u^T \right), \tag{19}$$

where u is the physical solution, (18) becomes

$$\tilde{I}[x, y] = \frac{1}{2} \int_{\Omega_c} \frac{d\xi d\eta}{J\sqrt{1 + \|\nabla u\|^2}} \left(x_\xi^2 + x_\eta^2 + y_\xi^2 + y_\eta^2 + u_\xi^2 + u_\eta^2 \right), \tag{20}$$

so the solution and the coordinate transformation are smooth with respect to the weighted integral of the squared derivatives in the computational domain.

It is not difficult to see that the mesh equation (10) defines a harmonic map for the monitor function (16) if $\gamma_1 = \gamma_2 = 0$. However, it is not obvious that the choice (16) for the monitor function is in general either unique or optimal in any sense. Based on (13) we have also chosen

$$\begin{aligned} G_1 &= w \left[(1 - \gamma_1 - \gamma_2) I + \gamma_1 S(\nabla \tilde{\xi}) + \gamma_2 S(\vec{v}^1) \right]^{-1}, \\ G_2 &= w \left[(1 - \gamma_1 - \gamma_2) I + \gamma_1 S(\nabla \tilde{\eta}) + \gamma_2 S(\vec{v}^2) \right]^{-1} \end{aligned} \tag{21}$$

with $w = \sqrt{1 + \|\nabla u\|^2}$, and preliminary experiments have shown that using either (16) or (21) with the MMPDE presented in the next section leads to comparable results. In this case, the map is not harmonic even when $\gamma_1 = \gamma_2 = 0$, as G_1 and G_2 cannot be written into the form $G_1 = G_2 = \frac{1}{\sqrt{g}} G$ for some symmetric positive definite matrix G.

4. The moving mesh method and solution procedure. Our basic MMPDE is developed from the gradient flow equation of functional $I[\xi, \eta]$. It is defined as

$$\frac{\partial \xi}{\partial t} = -\frac{1}{\tau \sqrt{\tilde{g}_1}} \frac{\delta I}{\delta \xi}, \qquad \frac{\partial \eta}{\partial t} = -\frac{1}{\tau \sqrt{\tilde{g}_2}} \frac{\delta I}{\delta \eta} \tag{22}$$

or

$$\frac{\partial \xi}{\partial t} = \frac{1}{\tau \sqrt{\tilde{g}_1}} \nabla \cdot \left(G_1^{-1} \nabla \xi \right), \qquad \frac{\partial \eta}{\partial t} = \frac{1}{\tau \sqrt{\tilde{g}_2}} \nabla \cdot \left(G_2^{-1} \nabla \eta \right), \tag{23}$$

where $\tau > 0$ is a user-defined parameter for adjusting the time scale of the mesh equation and $\tilde{g}_1$ and $\tilde{g}_2$ are, respectively, the determinants of matrices $\tilde{G}_1$ and $\tilde{G}_2$ in (16). For actual computation, we interchange the dependent and independent variables. From (23) we obtain

$$\begin{aligned} \frac{\partial \vec{x}}{\partial t} = &- \frac{\vec{x}_\xi}{\tau \sqrt{\tilde{g}_1} J} \left\{ + \frac{\partial}{\partial \xi} \left[\frac{1}{Jg_1} \left(\vec{x}_\eta^T G_1 \vec{x}_\eta \right) \right] - \frac{\partial}{\partial \eta} \left[\frac{1}{Jg_1} \left(\vec{x}_\xi^T G_1 \vec{x}_\eta \right) \right] \right\} \\ &- \frac{\vec{x}_\eta}{\tau \sqrt{\tilde{g}_2} J} \left\{ - \frac{\partial}{\partial \xi} \left[\frac{1}{Jg_2} \left(\vec{x}_\eta^T G_2 \vec{x}_\xi \right) \right] + \frac{\partial}{\partial \eta} \left[\frac{1}{Jg_2} \left(\vec{x}_\xi^T G_2 \vec{x}_\xi \right) \right] \right\}. \end{aligned} \tag{24}$$

Note that $g_1 = g_2 = 1$ for the monitor function choice (16). In general, MMPDE (24) must be solved with appropriate boundary conditions. For simplicity, we use Dirichlet boundary conditions (i.e., the boundary points are fixed) in our computation. It is known (see the appendix) that the gradient flow equation for an energy functional has a unique solution when supplemented with Dirichlet boundary conditions. However, we emphasize that in many applications it is necessary to adapt the boundary points themselves to the physical solution in order to obtain satisfactory results. Various boundary condition strategies are currently under investigation.

We are now prepared to give a complete description of the full moving mesh method. We assume that the underlying time-dependent physical PDE is of the form

$$u_t = f(t, x, y, u, u_x, u_y, u_{xx}, u_{xy}, u_{yy}), \qquad t > 0,\ (x, y) \in \Omega_p, \tag{25}$$

subject to appropriate boundary and initial conditions, where u is the (possibly vector-valued) physical solution and the underlying problem is well posed. Using the MMPDE approach, we replace (25) with an extended problem involving a continuous coordinate transformation equation MMPDE (24) and the physical PDE expressed in the computational coordinate. There are three user-defined parameters for mesh movement and quality control: the mesh orthogonality control parameter γ_1, the directional control parameter γ_2, and the temporal smoothing parameter τ. The parameters γ_1 and γ_2 are dimensionless and hence relatively problem independent and easy to choose. The temporal smoothing parameter τ controls the time scale of the MMPDE and can itself be made a function of time and/or the position in the domain. Ideally, it should be chosen such that the time scale of the MMPDE is commensurate with that of the physical PDE, although this is a difficult task in general. Experience in one dimension [25] indicates that for most problems the computation is relatively insensitive to the choice of τ, and we expect this to also be the case in two dimensions.

The reference mesh $(\tilde{\xi}, \tilde{\eta})$, being used to ensure control of mesh skewness, should have reasonable orthogonality properties but generally is not intended to adapt to the physical solution. Such a mesh is usually fairly straightforward to generate, for instance, by using Thompson's boundary fitted elliptic mesh generator, an algebraic mesh generator, or an elliptic or hyperbolic orthogonal mesh generator (see [36]). In many cases, an analytical reference mesh is available. For example, the identity map $(\tilde{\xi}, \tilde{\eta}) = (x, y)$ can be used when the physical and computational domains coincide.

Alignment of the mesh with the vector fields associated with a certain physical flow has long been known to be an efficient means of improving the accuracy of calculations, e.g., see [7, 14, 18, 29, 35]. Such alignment can be very useful when there is a natural anisotropy in the problem, e.g., for problems in magnetized plasmas, or more generally for problems having dominant flow directions [7, 18]. There is not a general guide for selecting the reference vector field, as it is usually problem dependent (see [7] for more details).

The physical and mesh PDEs have to be solved simultaneously or alternately for the physical solution and the mesh. Simultaneous computation has commonly been used in the method of lines approach for one-dimensional moving mesh methods. However, it is less straightforward in two dimensions, where the size of the system consisting of the physical and mesh PDEs may become very large. Our purpose here is to examine the feasibility of the MMPDE approach in general and also as a technique with which to supplement the strategy for existing physical PDE solvers (for which the moving mesh approach would be used externally from such software). Consequently, in this paper these equations are integrated alternately in time. The procedure is given below.

The alternate solution procedure. Given the physical solution u^n, the mesh $\vec{x}^n \equiv (x^n, y^n)$, and the time stepsize Δt_n at time $t = t_n$.

(i) Compute the monitor functions $G_1^n = G_1(t_n, \vec{x}^n, u^n)$ and $G_2^n = G_2(t_n, \vec{x}^n, u^n)$.

(ii) Compute the new mesh $\vec{x}^{n+1}$ by integrating the MMPDE from $t = t_n$ to $t = t_n + \Delta t_n$, using $\vec{x}^n$ as the initial mesh and keeping the monitor functions G_1^n and G_2^n constant in time during the integration.

(iii) Compute the physical solution u^{n+1} by integrating the physical PDE from $t = t_n$ to $t = t_n + \Delta t_n$, using the mesh $\vec{x}(t) = \vec{x}^n + \frac{(t-t_n)}{\Delta t_n}(\vec{x}^{n+1} - \vec{x}^n)$ and mesh speed $\dot{\vec{x}}(t) = (\vec{x}^{n+1} - \vec{x}^n)/\Delta t_n$.

(iv) Choose Δt_{n+1} as the next time stepsize predicted during the physical PDE integration.

It is still unclear to us how reliable this alternate solution method for the extended system is for general problems, but it does have the advantages that two smaller systems are solved and that the MMPDE calculation is easily combined with existing PDE solvers (such as VODPK described below). In principle it allows for one to efficiently utilize special properties of the physical PDE and of the MMPDE. Further, we have not observed any instability of the method for our test examples, at least for relatively small values of τ. Obviously, simultaneous solution methods and other more efficient alternate solution methods deserve more investigation.

In the computations presented here the method of lines approach is employed for the numerical solution of both the physical and moving mesh PDEs. The ODE solver VODPK [4] and central finite differences are used for time integration and spatial discretizations, respectively. The discretizations are standard and straightforward (e.g., see [25, 36]), so they are only briefly described. For simplicity, it is assumed that the computational domain Ω_c is a rectangle and a uniform mesh $\{(\xi_i, \eta_j), i = 0, \ldots, N_1, j = 0, \ldots, N_2\}$ is used. Denote the mesh points by $(x_{ij}(t), y_{ij}(t))$ and the corresponding approximate solution by $u_{ij}(t)$. We illustrate the spatial discretization of MMPDE (24) for the term $\frac{\partial}{\partial \xi}[\frac{1}{Jg_1}(\vec{x}_\eta^T G_1 \vec{x}_\eta)]$. It is approximated at the point (ξ_i, η_j) by the central finite difference

$$\begin{aligned}\frac{\partial}{\partial \xi}\left[\frac{1}{Jg_1}\left(\vec{x}_\eta^T G_1 \vec{x}_\eta\right)\right]\Big|_{(\xi_i,\eta_j)} \approx\ & \frac{1}{J_{i+\frac{1}{2},j} g_{1,i+\frac{1}{2},j}} \vec{x}^T_{\eta,i+\frac{1}{2},j} G_{1,i+\frac{1}{2},j} \vec{x}_{\eta,i+\frac{1}{2},j} \\ & - \frac{1}{J_{i-\frac{1}{2},j} g_{1,i-\frac{1}{2},j}} \vec{x}^T_{\eta,i-\frac{1}{2},j} G_{1,i-\frac{1}{2},j} \vec{x}_{\eta,i-\frac{1}{2},j},\end{aligned} \tag{26}$$

where

$$\begin{aligned} G_{1,i+\frac{1}{2},j} &= \frac{1}{2}\left(G_{1,i+1,j} + G_{1,i,j}\right), \\ g_{1,i+\frac{1}{2},j} &= \det\left(G_{1,i+\frac{1}{2},j}\right), \\ J_{i+\frac{1}{2},j} &= \frac{1}{4}(x_{i+1,j} - x_{i,j})(y_{i+1,j+1} - y_{i+1,j-1} + y_{i,j+1} - y_{i,j-1}) \\ &\quad - \frac{1}{4}(y_{i+1,j} - y_{i,j})(x_{i+1,j+1} - x_{i+1,j-1} + x_{i,j+1} - x_{i,j-1}), \\ \vec{x}_{\eta,i+\frac{1}{2},j} &= \frac{1}{4}(\vec{x}_{i+1,j+1} - \vec{x}_{i+1,j-1} + \vec{x}_{i,j+1} - \vec{x}_{i,j-1}). \end{aligned} \tag{27}$$

Other terms in the MMPDE (24) are similarly discretized. In the final implementation, the monitor functions are smoothed to obtain smoother meshes. The following

low pass filter is applied four times to each of $\tilde{G}_1$ and $\tilde{G}_2$:

$$\begin{aligned}\tilde{\tilde{G}}_{jk} &= \frac{4}{16}\tilde{G}_{jk} + \frac{2}{16}\left(\tilde{G}_{j+1,k} + \tilde{G}_{j-1,k} + \tilde{G}_{j,k+1} + \tilde{G}_{j,k-1}\right) \\ &\quad + \frac{1}{16}\left(\tilde{G}_{j-1,k-1} + \tilde{G}_{j-1,k+1} + \tilde{G}_{j+1,k-1} + \tilde{G}_{j+1,k+1}\right). \end{aligned} \tag{28}$$

For the discretization of the physical PDE (25), we first transform it into

$$\dot{u} - u_x\dot{x} - u_y\dot{y} = f(t, x, y, u, u_x, u_y, u_{xx}, u_{xy}, u_{yy}), \tag{29}$$

where "˙" denotes the time derivative when ξ and η are fixed, and the spatial derivatives are defined in terms of the computational coordinates by

$$\begin{aligned} u_x &= \frac{1}{J}\left[+(y_\eta u)_\xi - (y_\xi u)_\eta\right], \\ u_y &= \frac{1}{J}\left[-(x_\eta u)_\xi + (x_\xi u)_\eta\right], \\ u_{xx} &= \frac{1}{J}\left[+\left(\frac{y_\eta^2 u_\xi}{J}\right)_\xi - \left(\frac{y_\xi y_\eta u_\eta}{J}\right)_\xi - \left(\frac{y_\xi y_\eta u_\xi}{J}\right)_\eta + \left(\frac{y_\xi^2 u_\eta}{J}\right)_\eta\right], \\ u_{xy} &= \frac{1}{J}\left[-\left(\frac{x_\eta y_\eta u_\xi}{J}\right)_\xi + \left(\frac{x_\xi y_\eta u_\eta}{J}\right)_\xi + \left(\frac{x_\eta y_\xi u_\xi}{J}\right)_\eta - \left(\frac{x_\xi y_\xi u_\eta}{J}\right)_\eta\right], \\ u_{yy} &= \frac{1}{J}\left[+\left(\frac{x_\eta^2 u_\xi}{J}\right)_\xi - \left(\frac{x_\xi x_\eta u_\eta}{J}\right)_\xi - \left(\frac{x_\xi x_\eta u_\xi}{J}\right)_\eta + \left(\frac{x_\xi^2 u_\eta}{J}\right)_\eta\right]. \end{aligned} \tag{30}$$

Central finite differences are then used for the physical PDE in the form (29).

Both the physical and mesh PDEs are integrated in time using the ODE solver VODPK (without restarting on every new time interval). VODPK is an ODE solver which incorporates the preconditioned Krylov subspace iterative method SPIGMR (a scaled preconditioned incomplete version of GMRES) for the linear algebraic systems that arise in the case of stiff systems. It uses backward differentiation formulas (BDF) for stiff ODEs. For efficiency, we choose the option of providing the user-defined preconditioner H (approximation of the Jacobian of the ODE system) and the linear system solver for $Hv = b$. In all computation, the preconditioner H is defined as the 13-point row-sum-equivalent incomplete LU decomposition (e.g., see [27]) of the Jacobian of the ODE system arising from the discretization of the physical PDE or the MMPDE. Other required input data are the initial mesh and the relative and absolute local time stepping error tolerances (in a root-mean-square norm), which are taken as $rtol = atol = 10^{-5}$ when solving both the physical and mesh PDEs.

5. Numerical examples. In this section we present numerical results obtained for the moving mesh method described in the above sections on a mesh generation problem and two time-dependent PDEs. All computations are performed on a Silicon Graphics Onyx2 workstation (R10000) with a single processor and double precision algorithms.

Example 5.1 *(mesh generation for NACA*0012 *airfoil configuration).* Our first problem is to generate a C-type mesh for the NACA0012 airfoil configuration. In principle, many steady mesh generators can be used for this purpose. Key advantages of the moving mesh method for mesh generation are that well-tested ODE solvers

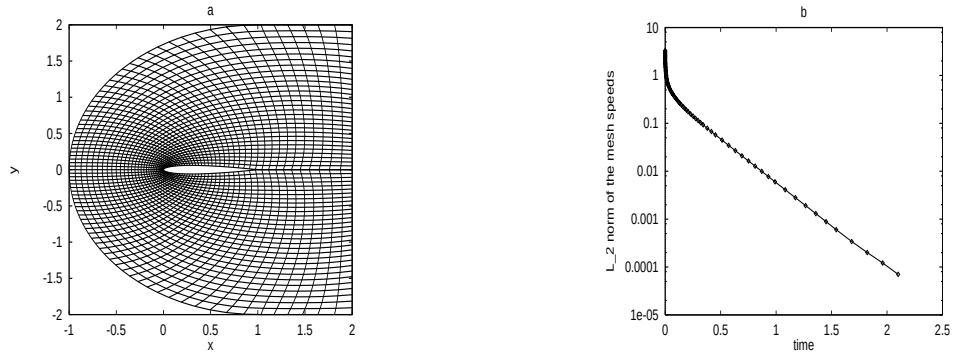

FIG. 1. (a) *A* 30×80 *C-type mesh for NACA*0012 *airfoil.* (b) *The* L_2 *norm of the mesh speeds as a function of time.*

TABLE 1
Example 5.1. *CPU times (in seconds).*

$N_1 \times N_2$	10×40	20×60	30×80	40×100	
MMPDE	4.72	16.2	38.3	69.3	$\sim (N_1 \times N_2)^{1.2}$
GS	14.5	130.6	530.8	1480.5	$\sim (N_1 \times N_2)^2$

are available for numerically solving the MMPDE and that the problem of obtaining convergence in the Newton iteration for the resulting nonlinear equations is avoided.

The MMPDE (24) is used with $u(x, y, t) \equiv 1$, $\gamma_1 = 0$, and $\gamma_2 = 0$. Figure 1a shows typical results with a 30×80 mesh obtained from an algebraic initial mesh, and Figure 1b shows the L_2 norm of the mesh speeds as a function of time. The computation is stopped when the L_2 norm of the mesh speeds (more precisely, of the residuals for the corresponding steady mesh equations) is less than 10^{-4}. As guaranteed by Theorem A.1 (see the appendix), the magnitude of mesh speeds for the continuous problem decreases (in fact, monotonically) and the mesh tends to a (steady state) harmonic mesh. The time integrator VODPK takes small steps at the beginning and much larger steps later on. The CPU times are listed in Table 1 for four runs with different numbers of mesh points. For comparison purposes, corresponding results are given for the pointwise Gauss–Seidel iteration (GS) to solve the steady mesh equation (i.e., the equation obtained by dropping the mesh speed terms in (24)). It is interesting to notice that the cost of GS to solve the steady mesh equation is quadratically increasing and that with MMPDE it is roughly linearly increasing. This agrees with others' observations about the use of preconditioning techniques (e.g., see [11]).

Example 5.2 *(Burgers' equation).* In this example our moving mesh method is applied to the numerical solution of the initial boundary value problem for the two-dimensional Burgers' equation

(31)
$$u_t = R\Delta u - uu_x - uu_y, \quad (x, y) \in (0, 1) \times (0, 1), \quad 0.25 < t \leq 1.25, \quad R = 5 \times 10^{-3},$$

subject to initial and Dirichlet boundary conditions chosen such that the exact solution to the problem is $u(x, y, t) = (1 + e^{\frac{(x+y-t)}{2R}})^{-1}$. With these initial and boundary conditions, the solution is a straight-line wave (u is constant along line $x + y = c$) moving in the direction $\theta = 45°$.

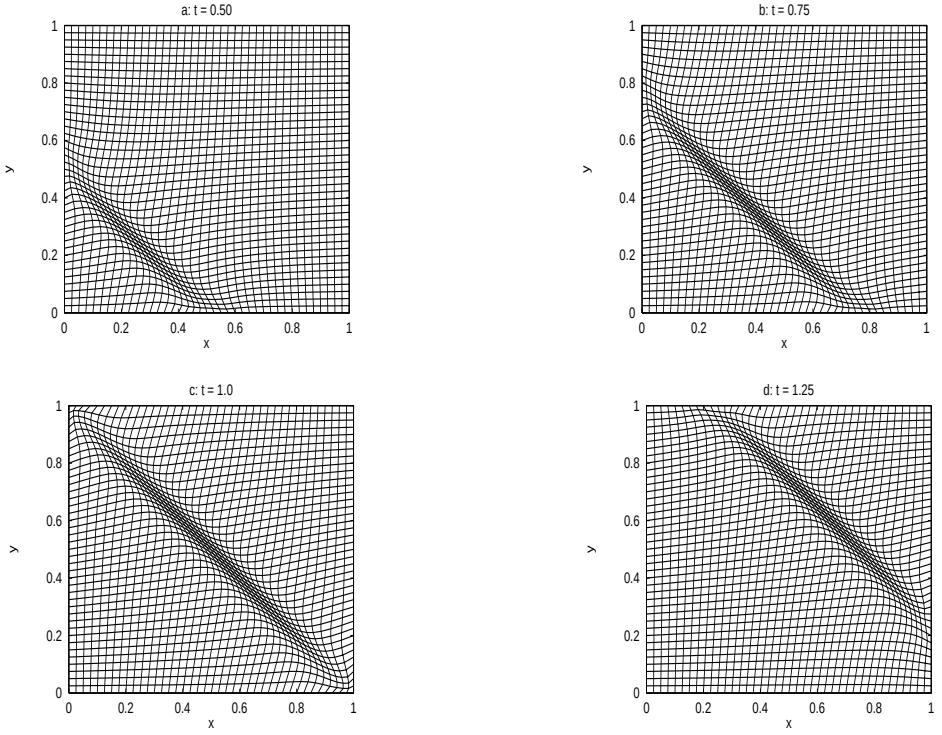

FIG. 2. *Example* 5.2. 40×40 *mesh at different times obtained with the MMPDE with* $\gamma_1 = 0.1$ *and* $\tau = 0.1$.

Figures 2 and 3 show the moving meshes at different times obtained with the moving mesh method with $\tau = 0.1$, $\gamma_1 = 0.1$ and $\tau = 0.1$, $\gamma_1 = 0.5$, respectively. It can be seen that the mesh remains well concentrated in the region of large solution variation. Furthermore, the mesh obtained with orthogonality control $\gamma_1 = 0.5$ (the reference map is $\tilde{\xi} = x$, $\tilde{\eta} = y$) is much less skew than that obtained with $\gamma_1 = 0.1$. This is more apparent from Figure 4, where the solution error and the minimal mesh angle are plotted as functions of time for different values of γ_1. For instance, the minimal mesh angle remains larger than 50° for $\gamma_1 = 0.5$ and about 25° for $\gamma_1 = 0.1$. However, we can see once again from these figures that with the gain of orthogonality the mesh loses some degree of adaptivity. As a result, the solution error may become larger on less adaptive meshes. On the other hand, without orthogonality control (i.e., $\gamma_1 = 0$) the mesh can become very skew. A skew mesh can in turn force the integrators for both the physical and mesh PDE to take very small time steps and even stop the computation. This can easily be seen from Figure 5, where the minimal mesh angle in degrees and the time stepsize taken by VODPK (applied to the physical PDE) are shown as functions of time for the case $\gamma_1 = 0$ and $\tau = 0.1$. When the computation is halted around $t = 0.92$, the minimal mesh angle is about 0.8° and the stepsize is about 10^{-6}. Comparatively, the results appear less sensitive to the time scaling parameter τ. In Figure 6 the solution error and the mesh minimal angle are plotted as functions of time for $\tau = 10$, 1, 0.1, and 0.01. It is worth noting that the smaller τ is, the faster the mesh responds to the change in the physical solution. The results with $\tau = 0.1$ and 0.01 are basically the same. The solution error is also shown in Figure 4a for the fixed uniform mesh (FM) method. It shows that the error for the

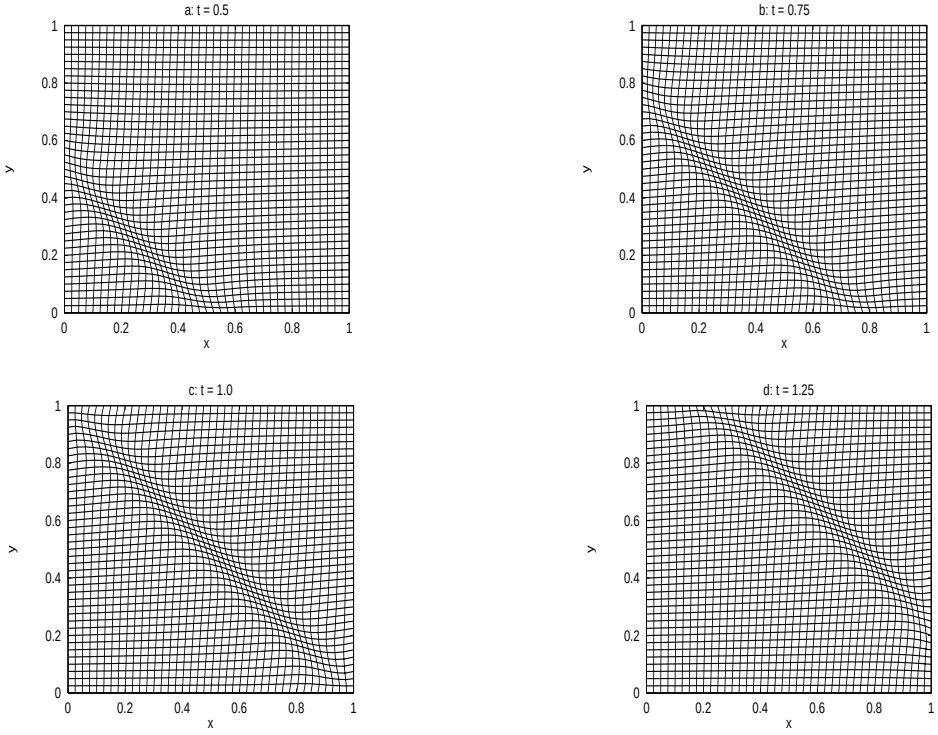

FIG. 3. *Example* 5.2. 40×40 *mesh at different times obtained with the MMPDE with* $\gamma_1 = 0.5$ *and* $\tau = 0.1$.

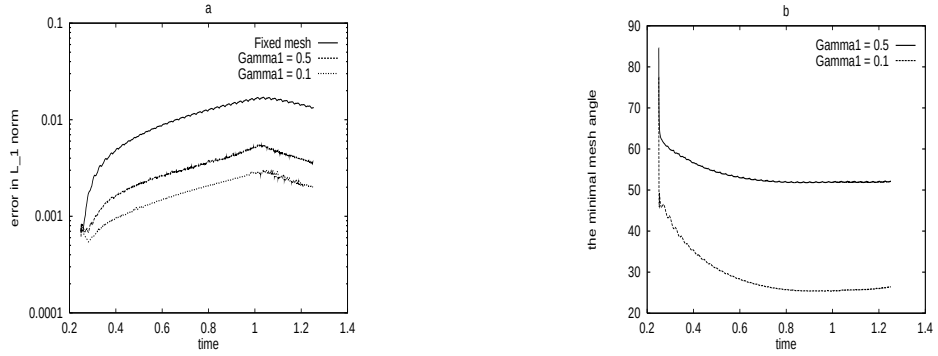

FIG. 4. *Example* 5.2. *The solution error and minimal mesh angle as functions of time for different values of* γ_1 ($\tau = 0.1, N_1 \times N_2 = 40 \times 40$).

moving mesh method with $\gamma_1 = 0$ and $\tau = 0.1$ is about 10 times smaller than that obtained with the fixed uniform mesh. A typical run with $\gamma_1 = 0.1$, $\tau = 0.1$, and $N_1 = N_2 = 40$ takes about 547 seconds CPU time.

Example 5.3 *(model problem in combustion theory).* For our final example, we apply the moving mesh method to the combustion problem

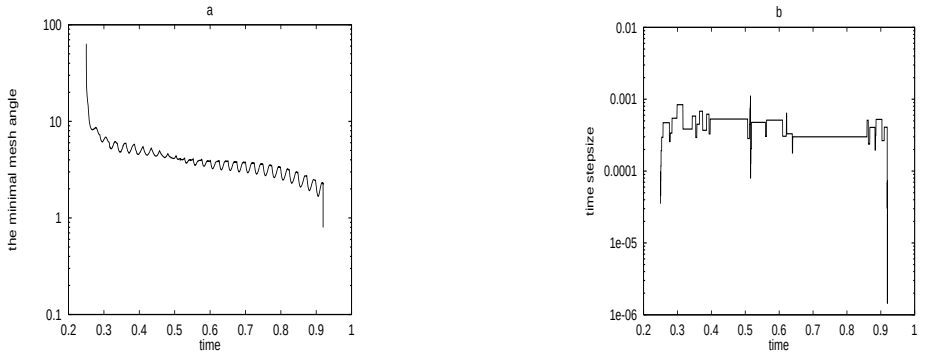

FIG. 5. *Example* 5.2. *The minimal mesh angle and the time stepsize taken by the VODPK (applied to the physical PDE) are shown as functions of time for the case* $\gamma_1 = 0$ *(without orthogonality control) and* $\tau = 0.1, N_1 \times N_2 = 40 \times 40$.

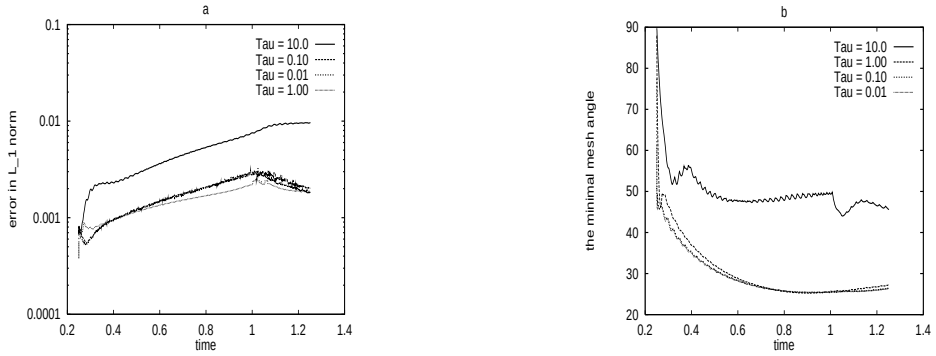

FIG. 6. *Example* 5.2. *The solution error and minimal mesh angle as functions of time for different values of* τ ($\gamma_1 = 0.1, N_1 \times N_2 = 40 \times 40$).

$$
\begin{aligned}
u_t &= \Delta u - \frac{R}{\alpha\delta} u e^{\delta(1-1/T)}, \\
LT_t &= \Delta T + \frac{R}{\delta} u e^{\delta(1-1/T)}, \quad (x,y) \in \Omega_p \equiv (-1,1)\times(-1,1), \quad t > 0, \\
u(x,y,t) &= 1,\ T(x,y,t) = 1, \qquad (x,y) \in \partial\Omega_p, \\
u(x,y,0) &= 1,\ T(x,y,0) = 1, \qquad (x,y) \in \Omega_p,
\end{aligned} \tag{32}
$$

where L, α, δ, and R are physical parameters. The variables u and T denote the concentration and temperature of a chemical which is undergoing a one-step reaction in the domain Ω_p. For small times the temperature gradually increases from unity with a "hot spot" forming at the origin. At a finite time, ignition occurs, causing the temperature at the origin to increase rapidly to $1 + \alpha$. A flame front then forms and propagates toward the boundary of the domain at a high speed. The degree of difficulty of the problem is determined by the value of δ. Following [31], we choose the physical parameters $L = 0.9$, $\alpha = 1$, $\delta = 20$, and $R = 5$.

Figure 7 shows the time evolution of a 40×40 mesh, obtained with the MMPDE with $\tau = 10^{-2}$ and $\gamma_1 = 0.1$. From these figures one can see that the mesh points remain concentrated around the region of the flame front while moving toward the boundary of the domain. Figure 8a shows the minimal angles of the meshes obtained with different degrees of skewness control. It can be seen that without skewness control

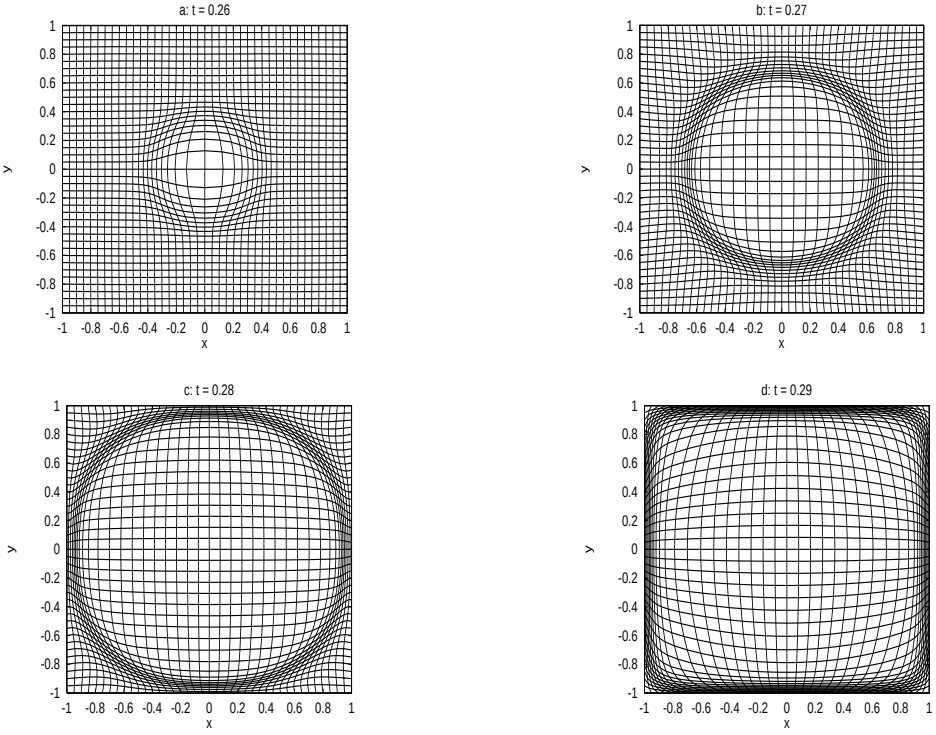

FIG. 7. *Example* 5.3. *Evolution of the mesh.*

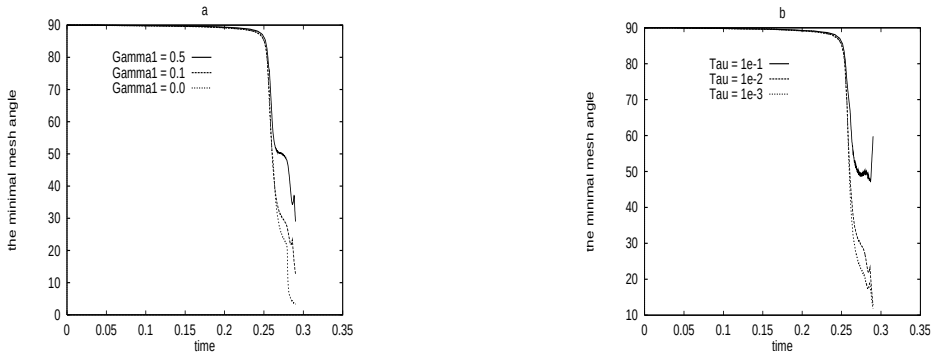

FIG. 8. *Example* 5.3. *The minimal mesh angle as a function of time.* (a) $\tau = 10^{-2}$ *and* $N_1 = N_2 = 40$ *are used.* (b) $\gamma_1 = 0.1$ *and* $N_1 = N_2 = 40$ *are used.*

the mesh becomes very skew (the minimal mesh angle is about 2° for $t = 0.29$) when the flame front reaches the boundary. It is worth noting that we have used a small value of τ ($\tau = 10^{-2}$) in this example because of the high propagation speed of the flame front. Figure 8b shows the results obtained with different values of τ. We can see that when a small value of τ is used, the mesh is more adaptive and responds faster to the change of the physical solution. A typical run with $\gamma_1 = 0.1$, $\tau = 10^{-2}$, and $N_1 = N_2 = 40$ takes about 367 seconds CPU time.

6. Conclusions and comments. An MMPDE mesh movement strategy based on the gradient flow equation has been introduced. This strategy can be regarded as a higher-dimensional generalization of the one-dimensional mesh movement approach presented in [24]. The method requires choosing the monitor function, the reference mesh, and the flow direction, and for efficient and reliable implementation some tuning of the user-defined parameters (the time scaling parameter τ, the mesh orthogonality control parameter γ_1, and the (flow) directional control parameter γ_2) can be necessary.

We have only considered the method with mesh adaptation and orthogonality control (i.e., $\gamma_2 = 0$). No attempt has been made to optimize its implementation. Instead, we have used a straightforward method of lines approach with a low-order finite difference spatial discretization, the monitor function (16), and VODPK with a crude alternate time integration scheme for the extended system consisting of the physical PDE and mesh PDE. The moving mesh method has been applied to three two-dimensional examples, one of them being for mesh generation and the others for the numerical solution of time-dependent PDEs. The results have shown the ability of the method to concentrate the mesh points around the areas of large variation in the physical solution and to control the mesh skewness.

While preliminary results are encouraging and the feasibility of the method is demonstrated, a number of practical issues are yet to be addressed. For instance, a more robust and efficient solution method should be developed for the time integration of the extended system, and other monitor functions besides arclength need investigation. Control of the mesh quality near the boundaries is another important issue. A conservative discretization of the physical PDEs over a moving mesh should be used in order to obtain more accuracy. Investigation of these issues and application of the moving mesh method to more challenging engineering problems are currently underway.

Appendix. Harmonic maps on Riemannian manifolds. Let M and N be compact Riemannian manifolds with metric tensors g_{ij} and $h_{\alpha\beta}$ in local coordinates $\vec{x} = (x^1, x^2, x^3)^T$ and $\vec{\xi} = (\xi^1, \xi^2, \xi^3)^T$, respectively. For a map $\vec{\xi} = \vec{\xi}(\vec{x})$ from M to N, we define its energy or action integral as

$$E[\vec{\xi}] = \int_M d\vec{x}\sqrt{g} \sum_{i,j,\alpha,\beta} g^{ij} h_{\alpha\beta} \frac{\partial \xi^\alpha}{\partial x^i} \frac{\partial \xi^\beta}{\partial x^j}, \tag{33}$$

where $G = (g_{ij})$, $g = \det(G)$, and $(g^{ij}) = G^{-1}$. A map $\vec{\xi}$ is called *harmonic* if it solves the Euler–Lagrange equation $\frac{\delta E}{\delta \vec{\xi}} = 0$ for the energy functional $E[\vec{\xi}]$, viz.,

$$\frac{1}{\sqrt{g}} \sum_{i,j} \frac{\partial}{\partial x^i} \left[\sqrt{g} g^{ij} \frac{\partial \xi^\alpha}{\partial x^j} \right] + \sum_{i,j,\beta,\gamma} g^{ij} \Gamma^\alpha_{\beta\gamma}(N) \frac{\partial \xi^\beta}{\partial x^i} \frac{\partial \xi^\gamma}{\partial x^j} = 0, \tag{34}$$

where $\Gamma^\alpha_{\beta\gamma}(N)$ is the Christoffel symbol of the second kind, defined by

$$\Gamma^\alpha_{\beta\gamma}(N) = \frac{1}{2} \sum_\lambda h^{\alpha\lambda} \left[\frac{\partial h_{\lambda\beta}}{\partial \xi^\gamma} + \frac{\partial h_{\lambda\gamma}}{\partial \xi^\beta} - \frac{\partial h_{\beta\gamma}}{\partial \xi^\lambda} \right]. \tag{35}$$

Existence and uniqueness of the harmonic map are guaranteed when the Riemannian curvature of N is nonpositive and its boundary ∂N is convex (see [19, 34]), and

invertibility can be guaranteed at least when $\dim(M) = \dim(N) = 2$. For the existence of the solution to the corresponding gradient (or heat) equation, we have the following theorem.

THEOREM A.1 (Hamilton [19]). *Let M and N be compact Riemannian manifolds with boundary and with (time-dependent) metric tensors g_{ij} and $h_{\alpha\beta}$. Suppose that N has nonpositive sectional curvature and that ∂N is convex. Let $h : \partial M \to N$ be any smooth map and $\xi_0 : M \to N$ a smooth map with $\xi_0|_{\partial M} = h$ in any given relative homotopy class. There exists a continuous map $\xi : M \times [0, \infty) \to N$ that is smooth except at the corner $\partial M \times 0$ satisfying the nonlinear heat equation*

$$\frac{\partial \xi^\alpha}{\partial t} = \frac{1}{\sqrt{g}} \sum_{i,j} \frac{\partial}{\partial x^i} \left[\sqrt{g} g^{ij} \frac{\partial \xi^\alpha}{\partial x^j} \right] + \sum_{i,j,\beta,\gamma} g^{ij} \Gamma^\alpha_{\beta\gamma}(N) \frac{\partial \xi^\beta}{\partial x^i} \frac{\partial \xi^\gamma}{\partial x^j} \text{ on } M \times [0, \infty), \tag{36}$$

$$\xi = h \text{ on } \partial M \times [0, \infty),$$

$$\xi = \xi_0 \text{ on } M \times 0,$$

where the equation is satisfied in the L^p_2 sense ($p > \dim M + 2$, $L^p_2 =$ space $\{\xi : D^2\xi \in L^p\}$) at the corner $\partial M \times 0$ and in the strong sense everywhere else. Moreover, all derivatives $\left(\frac{\partial}{\partial t}\right)^i \nabla^j \xi$ remain uniformly bounded as $t \to \infty$.

Let $\xi_t(x) = \xi(x, t)$ and regard ξ_t as a map of M into N. Then $\xi_t : M \to N$ converges in $C^\infty(X)$ to a harmonic map $\xi_\infty : M \to N$ with $\nabla^2 \xi_\infty = 0$ in the same relative homotopy class as ξ_0.

In the field of mesh generation and adaptation, M is often chosen as the physical domain Ω_p and N as the logical domain Ω_c. In most practical cases, one can restrict the logical domain Ω_c to having the Euclidean geometry (i.e., $h_{\alpha\beta} = \delta_{\alpha\beta}$) being, say, a square or a cube. When Ω_c is Euclidean and $\partial\Omega_c$ is convex, its curvature is zero, and the existence [19] and invertibility [34] theorems guarantee the suitability of the map for mesh generation and adaptation.

Acknowledgments. We are most grateful to Huaxiong Huang and Michael Lunney for helpful remarks during the preparation of this paper.

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 (1986), pp. 778–796.

[2] S. ADJERID, J. E. FLAHERTY, P. K. MOORE, AND Y. J. WANG, *High-order adaptive methods for parabolic systems*, Phys. D, 60 (1992), pp. 94–111.

[3] S. K. ALIABADI AND T. E. TEZDUYAR, *Space-time finite element computation of compressible flows involving moving boundaries and interfaces*, Comput. Methods Appl. Mech. Engrg., 107 (1993), pp. 209–224.

[4] P. N. BROWN, G. D. BYRNE, AND A. C. HINDMARSH, *VODE: A variable-coefficient ODE solver*, SIAM J. Sci. Stat. Comput., 10 (1989), pp. 1038–1052.

[5] I. BABUSKA AND M. SURI, *The p and h-p versions of the finite element method: An overview*, Comput. Methods Appl. Mech. Engrg., 80 (1990), pp. 5–26.

[6] M. J. BAINES, *Moving Finite Elements*, Clarendon Press, Oxford, UK, 1994.

[7] J. U. BRACKBILL, *An adaptive grid with directional control*, J. Comput. Phys., 108 (1993), pp. 38–50.

[8] J. U. BRACKBILL AND J. S. SALTZMAN, *Adaptive zoning for singular problems in two dimensions*, J. Comput. Phys., 46 (1982), pp. 342–368.

[9] C. J. BUDD, J. CHEN, W. HUANG, AND R. D. RUSSELL, *Moving mesh methods with applications to blow-up problems for PDEs*, in Numerical Analysis 1995: Proc. of 1995 Biennial Conference on Numerical Analysis, Pitman Res. Notes Math. Ser. 344, D. F. Griffiths and G. A. Watson, eds., Longman Scientific and Technical, Harlow, UK, 1996, pp. 1–17.

[10] C. J. Budd, W. Huang, and R. D. Russell, *Moving mesh methods for problems with blow-up*, SIAM J. Sci. Comput., 17 (1996), pp. 305–327.
[11] C. Canuto, M. Y. Hussani, A. Quarteroni, and T. A Zang, *Spectral Methods in Fluid Dynamics*, Springer-Verlag, Berlin, New York, 1988.
[12] 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 (1998), pp. 766–798.
[13] R. Courant and D. Hilbert, *Methods of Mathematical Physics*, Vols. I and II, Interscience Publishers, Inc., New York, 1953.
[14] Y. C. Chao and S. S. Liu, *Streamline adaptive grid method for complex flow computation*, Numer. Heat Transfer, Part B, 20 (1991), pp. 145–168.
[15] E. A. Dorfi and L. O'C. Drury, *Simple adaptive grids for* 1*-D initial value problems*, J. Comput. Phys., 69 (1987), pp. 175–195.
[16] A. S. Dvinsky, *Adaptive grid generation from harmonic maps on Riemannian manifolds*, J. Comput. Phys., 95 (1991), pp. 450–476.
[17] K. Eriksson and C. Johnson, *Adaptive finite element methods for parabolic problems*. IV: *Nonlinear problems*, SIAM J. Numer. Anal., 32 (1995), pp. 1729–1749.
[18] A. E. Guannakopoulos and A. J. Engel, *Directional control in grid generation*, J. Comput. Phys., 74 (1988), 422–439.
[19] R. Hamilton, *Harmonic Maps of Manifolds with Boundary*, Lecture Notes in Math. 471, Springer-Verlag, New York, 1975.
[20] 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 (1991), pp. 254–302.
[21] R. G. Hindman and J. Spencer, *A New Approach to Truly Adaptive Grid Generation*, Paper 83-0450, AIAA Ed. Ser. 1, Washington, DC, 1983.
[22] W. Huang and R. D. Russell, *Analysis of moving mesh partial differential equations with spatial smoothing*, SIAM J. Numer. Anal., 34 (1997), pp. 1106–1126.
[23] W. Huang and R. D. Russell, *A moving collocation method for the numerical solution of time dependent differential equations*, Appl. Numer. Math., 20 (1996), pp. 101–116.
[24] W. Huang, Y. Ren, and R. D. Russell, *Moving mesh partial differential equations (MMPDEs) based on the equidistribution principle*, SIAM J. Numer. Anal., 31 (1994), pp. 709–730.
[25] W. Huang, Y. Ren, and R. D. Russell, *Moving mesh methods based on moving mesh partial differential equations*, J. Comput. Phys., 113 (1994), pp. 279–290.
[26] J. M. Hyman and B. Larrouturou, *Dynamic rezone methods for partial differential equations in one space dimension*, Appl. Numer. Math., 5 (1989), pp. 435–450.
[27] S. A. Jordan and M. L. Spaulding, *A fast algorithm for grid generation*, J. Comput. Phys., 104 (1993), pp. 118–128.
[28] A. A. Johnson and T. E. Tezduyar, *Mesh update strategies in parallel finite element computations of flow problems with moving boundaries and interfaces*, Comput. Methods Appl. Mech. Engrg., 119 (1994), pp. 73–94.
[29] P. Knupp, *Mesh generation using vector-fields*, J. Comput. Phys., 119 (1995), pp. 142–148.
[30] P. Knupp and S. Steinberg, *Fundamentals of Grid Generation*, CRC Press, Boca Raton, FL, 1994.
[31] P. K. Moore and J. E. Flaherty, *Adaptive local overlapping grid methods for parabolic systems in two space dimensions*, J. Comput. Phys., 98 (1992), pp. 54–63.
[32] K. Miller and R. N. Miller, *Moving finite elements*. I, SIAM J. Numer. Anal., 18 (1981), pp. 1019–1032.
[33] L. R. Petzold, *Observations on an adaptive moving grid method for one-dimensional systems for partial differential equations*, Appl. Numer. Math., 3 (1987), pp. 347–360.
[34] R. Schoen and S.-Y. Yau, *On univalent harmonic maps between surfaces*, Invent. Math., 44 (1978), pp. 265–278.
[35] G. R. Shubin and J. B. Bell, *An analysis of the grid orientation effect in numerical simulation of miscible displacement*, Comput. Methods Appl. Mech. Engrg., 47 (1984), pp. 47–71.
[36] J. F. Thompson, Z. U. A. Warsi, and C. W. Mastin, *Numerical Grid Generation*, North–Holland, New York, 1985.
[37] A. Winslow, *Numerical solution of the quasi-linear Poisson equation in a nonuniform triangle mesh*, J. Comput. Phys., 1 (1967), pp. 149–172.