Available online at www.sciencedirect.com
SCIENCE DIRECT•
ELSEVIER Computers and Mathematics with Applications 48 (2004) 1077–1085
An International Journal computers & mathematics with applications
www.elsevier.com/locate/camwa

Adaptive Grid Generation Based on the Least-Squares Finite-Element Method

XIAN-XIN CAI
Nanhua Power Research Institute
Zhuzhou, Hunan 412002, P.R. China

D. FLEITAS
Department of Mathematics, University of Texas
Arlington, TX 76019, U.S.A.

BONAN JIANG
Department of Mathematics and Statistics
Oakland University
Rochester, MI 48309, U.S.A.

GUOJUN LIAO*
Department of Mathematics, University of Texas
Arlington, TX 76019, U.S.A.
liao@uta.edu

Abstract—Approximate solutions of a partial differential equation become inaccurate if they are computed on a fixed grid that is not sufficiently fine in regions of the domain where the variables change rapidly. For time dependent problems, special features of a partial differential equation and their location could change in time as well. Thus, adaptive grid methods are necessary.

In this paper, we develop an adaptive deformation method based on the least-squares finite-element method (LSFEM). A main advantage of this method as compared to the existing deformation method is its ability to generate adaptive grids on domains with moving boundary. It computes the node velocity from a div-curl system according to an error indicator (monitor function), and then moves the nodes to new locations so that the size of the new grid cells can be directly controlled. In this method, the connectivity of the nodes is unchanged if the grid quality is acceptable. Otherwise, various optimization procedures can be applied after node movements to improve grid quality. The grid formed becomes refined in regions where the solution error is large. © 2004 Elsevier Ltd. All rights reserved.

Keywords—Least-squares finite elements, Grid deformation.

1. INTRODUCTION

Grid generation methods can be classified into structured and unstructured methods. A very popular approach in structured methods is to use an algebraic method to generate an initial grid and then use an elliptic differential system to improve the grid quality.

Variational methods are also very successful in generating grids by optimizing a combination of the functionals representing grid smoothness, cell skewness, and cell size. See [1–3].

*Author to whom all correspondence should be addressed.

0898-1221/04/$ - see front matter © 2004 Elsevier Ltd. All rights reserved. Typeset by $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX
doi:10.1016/j.camwa.2004.10.006

The above methods lead to nonlinear differential systems that are solved on a uniform grid of a simple computational domain (such as a cube in 3D). For domains of complex geometry, multiblock methods are developed.

For finite-element methods, unstructured grids are used due to their flexibility in discretizing domains of complex geometry. We refer to [4] for more information in this area.

The method described in this paper can be used to generate adaptive (structured or unstructured) grids on the physical domain by solving a linear, first-order differential system (div-curl). Since the div-curl system is solved by the LSFEM, the method can be used to deform an unstructured grid directly on the physical domain. In this paper, the method is formulated in Sections 2–4. Numerical examples are given in Section 5.

2. THE DEFORMATION METHOD

A preliminary version of this method appeared in [5], which is formulated and demonstrated for adaptation towards steady features on fixed domains. The version developed in the current paper is capable of adapting the grids according to time dependent features on domains with moving boundaries.

The deformation method has its origin in differential geometry [6]. It was reformulated for grid generation in [7]. The method generates a time-dependent nodal mapping from a domain $\Omega(t_0)$ to another domain $\Omega(T)$. A monitor function is used to obtain a vector field that moves the grid nodes to desired locations.

Assuming that we formed a monitor function

$$f(\mathbf{x},t) > 0, \qquad \text{for } \mathbf{x} \in \Omega(t) \text{ and } t \text{ in } [t_0,T] \tag{2.1}$$

(e.g., $f(\mathbf{x},t) = C/\delta(\mathbf{x},t)$ for some $C > 0$ where $\delta(\mathbf{x},t)$ is an error estimator), such that

$$\int_{\Omega(t)} \frac{1}{f(\omega,t)}\, d\omega = |\Omega(t_0)|. \tag{2.2}$$

We look for a time-dependent mapping $\phi(\cdot,t) : \Omega(t_0) \to \Omega(t)$ such that

$$\det \nabla\phi(\mathbf{x},t) = f(\phi(\mathbf{x},t),t), \qquad \text{for } t_0 \le t \le T. \tag{2.3}$$

Also we require that $\phi(\mathbf{x},t) \in \partial\Omega(t)$ for all $\mathbf{x} \in \partial\Omega(t_0)$.

The mapping can be calculated in two steps. First, find a vector field $\mathbf{u}(\mathbf{x},t)$ that satisfies

$$\operatorname{div} \mathbf{u}(\mathbf{x},t) = -\frac{\partial}{\partial t}\left(\frac{1}{f(\mathbf{x},t)}\right), \qquad \mathbf{x} \in \Omega(t), \tag{2.4a}$$

$$\operatorname{curl} \mathbf{u}(\mathbf{x},t) = 0, \qquad \mathbf{x} \in \Omega(t), \tag{2.4b}$$

$$\mathbf{u}(\mathbf{x},t) \cdot \mathbf{n} = 0 \text{ or } \mathbf{u}(\mathbf{x},t) = \mathbf{g}(\mathbf{x},t), \qquad \mathbf{x} \in \partial\Omega(t), \tag{2.4c}$$

for $t_0 \le t \le T$, where $\mathbf{n}$ is the outward normal to $\partial\Omega(t)$ and $\mathbf{g}$ is a boundary vector field determined by the boundary movement.

Second, find ϕ by solving the transport equation (the deformation ODE) for each fixed $\mathbf{x} \in \Omega(t_0)$,

$$\frac{\partial}{\partial t}\phi(\mathbf{x},t) = f(\phi(\mathbf{x},t),t)\mathbf{u}(\phi(\mathbf{x},t),t), \qquad \text{for } t_0 \le t \le T. \tag{2.5}$$

These steps imply (2.3) which in turn ensures that the grid becomes more refined on regions of large error. The mathematical foundation of this method is established from the following.

THEOREM. *The mapping ϕ obtained from (2.4) and (2.5) satisfies*

$$(J(\phi) :=) \det \nabla\phi(\mathbf{x},t) = f(\phi(\mathbf{x},t),t) \tag{2.6}$$

for each $\mathbf{x} \in \Omega(t_0)$ and each t in $[t_0,T]$.

The theorem is proved by showing that $\frac{d}{dt}(J(\phi)/f(\phi,t)) = 0$, and therefore $Jf = 1$ if $J(\phi)/f(\phi,t)|_{t=t_0} = 1$. Details of the proof can be found in [8].

This method has been applied to calculations of flows in [9] and [10]. A version of the method is developed in [11] with the use of a level set method.

3. LEAST-SQUARES FINITE-ELEMENT METHOD

The least-squares finite-element method (LSFEM) is based on the minimization of the residual in a least-squares sense. In this method a vector field $\mathbf{u}$ that minimizes the functional

$$I(\mathbf{v}) = \int_\Omega |A\mathbf{v} - \mathbf{f}|^2 \, d\omega \tag{3.1}$$

is sought within the constraint of a given boundary condition. For more information on LSFEM, we refer to [12–17].

Consider the linear boundary-value problem

$$A\mathbf{u} = \mathbf{f}, \qquad \text{in } \Omega, \tag{3.2a}$$

$$B\mathbf{u} = \mathbf{g}, \qquad \text{on } \Gamma, \tag{3.2b}$$

where

$$A\mathbf{u} = \sum_{i=1}^{n_d} A_i \frac{\partial \mathbf{u}}{\partial x_i} + A_0\mathbf{u}, \qquad B \text{ is a boundary operator}, \tag{3.3}$$

and $\mathbf{x} = (x_1, x_2, \ldots, x_{n_d})$ for $n_d = 1, 2$ or 3,

$$\mathbf{u} = \begin{pmatrix} u_1 \\ u_2 \\ \ldots \\ u_m \end{pmatrix}, \qquad \mathbf{f} = \begin{pmatrix} f_1 \\ f_2 \\ \ldots \\ f_m \end{pmatrix}, \qquad \text{and} \qquad \mathbf{g} = \begin{pmatrix} g_1 \\ g_2 \\ \ldots \\ g_m \end{pmatrix},$$

for m = number of variables at each node.

Without loss of generality, we assume $\mathbf{g} = 0$. Let $\mathbf{R} = A\mathbf{v} - \mathbf{f}$ in Ω for an arbitrary test function $\mathbf{v} \in \mathcal{V} \subset \mathbf{L}_2(\Omega) = [L_2(\Omega)]^m$. The distance between $A\mathbf{v}$ and $\mathbf{f}$ is given by

$$\|\mathbf{R}\|_0^2 = \int_\Omega |A\mathbf{v} - \mathbf{f}|^2 \, d\omega, \tag{3.4}$$

where

$$\|\mathbf{v}\|_0^2 = \sum_{i=1}^{m} \|v_i\|_0^2 = \sum_{i=1}^{m} \int_\Omega v_i^2 \, d\omega = \int_\Omega \mathbf{v} \cdot \mathbf{v} \, d\omega. \tag{3.5}$$

Since $\mathbf{R}$ is not zero we have $\|\mathbf{R}\|_0^2 \geq 0$, and the equality holds only if $\mathbf{v}$ is the exact solution in (3.2). The solution $\mathbf{u}$ to (3.2) can be viewed as an element of $\mathcal{V}$ that minimizes the L_2 distance between $A\mathbf{v}$ and $\mathbf{f}$,

$$I(\mathbf{v}) = \|A\mathbf{v} - \mathbf{f}\|_0^2 = (A\mathbf{v} - \mathbf{f}, A\mathbf{v} - \mathbf{f}), \qquad \text{on } V. \tag{3.6}$$

A necessary condition for $\mathbf{u} \in \mathcal{V}$ to minimize $I(\mathbf{v})$ is the vanishing of its first variation, that is,

$$\lim_{t \to 0} \frac{\mathrm{d}}{\mathrm{d}t} I(\mathbf{u} + t\mathbf{v}) = 2 \int_\Omega (A\mathbf{v}) \cdot (A\mathbf{u} - \mathbf{f}) \, d\omega = 0, \qquad \forall \mathbf{v} \in \mathcal{V}, \tag{3.7}$$

and hence,

$$\int_\Omega A\mathbf{u} \cdot A\mathbf{v} \, d\omega = \int_\Omega \mathbf{f} \cdot A\mathbf{v} \, d\omega, \qquad \forall \mathbf{v} \in \mathcal{V}. \tag{3.8}$$

Assuming that A is bounded below, a discretization of (3.8) leads to a symmetric positive-definite matrix. Subdividing the domain into a union of finite elements we use the expansion of $\mathbf{u}_h$ in each element

$$\mathbf{u}_h^e(\mathbf{x}) = \sum_{j=1}^{N_n} \psi_j(\mathbf{x}) = \begin{pmatrix} u_{1j} \\ u_{2j} \\ \ldots \\ u_{mj} \end{pmatrix}, \tag{3.9}$$

where u_{ij} is the nodal value of u_i at the j^{th} node, ψ_js are the shape functions, and N_n is the number of nodes in an element. Using (3.9) in (3.8), we obtain a linear system of algebraic equations

$$KU = F, \tag{3.10}$$

where K and F are assembled from element matrices

$$K_e = \int_{\Omega_e} (A\psi_1, A\psi_2, \dots, A\psi_{N_n})^{\mathsf{T}} (A\psi_1, A\psi_2, \dots, A\psi_{N_n})\, d\omega, \tag{3.11}$$

$$F_e = \int_{\Omega_e} (A\psi_1, A\psi_2, \dots, A\psi_{N_n})^{\mathsf{T}} \mathbf{f}\, d\omega. \tag{3.12}$$

The boundary conditions (3.2b) can also be included into (3.6). The discretization by LSFEM always leads to symmetric positive-definite matrices which can be efficiently solved.

In the deformation method for fixed domains the Neumann boundary condition (i.e., $\nabla \cdot \mathbf{u} = 0$ on $\partial\Omega$) is used, which ensures that boundary nodes move along the boundary. For free surface or moving boundary, inflow conditions should be enforced. The LSFEM scheme can be used to solve the main PDEs (governing the underlying physical phenomenon) and the div-curl system used to move the grid nodes.

4. MOVING GRID BASED ON LEAST-SQUARES FINITE-ELEMENT METHOD

In this section, we formulate a deformation method which is based on the LSFEM. This work further develops the ideas that first appeared in [5] and gives numerical examples on domains with moving boundaries. The method consists of the following steps.

STEP 1. Define monitor function f and form the right-hand side of the div equation.

STEP 2. Solve div-curl system (2.4) by LSFEM at each time step.

STEP 3. Solve for the new node location from the deformation ODE (2.5).

The Monitor Function

For refinement based on the solution errors by LSFEM, a monitor function can be constructed from the residual. This construction will be demonstrated in another paper on adaptive LSFEM for PDEs.

For movement towards an interface or a boundary I (on a fixed or moving domain), we first construct a function $\bar{f}$ such that

$$\bar{f} = \begin{cases} \text{small}, & \text{near interface,} \\ 1, & \text{far from it.} \end{cases} \tag{4.1}$$

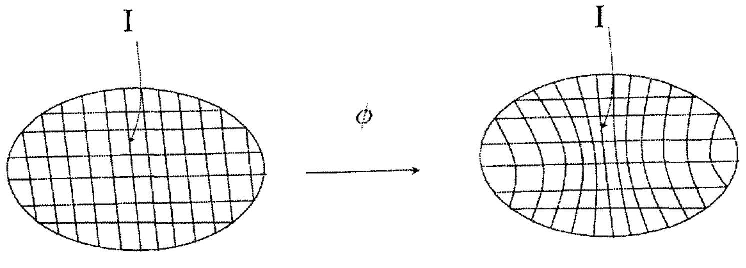


Figure 1. Refinement towards an interface.

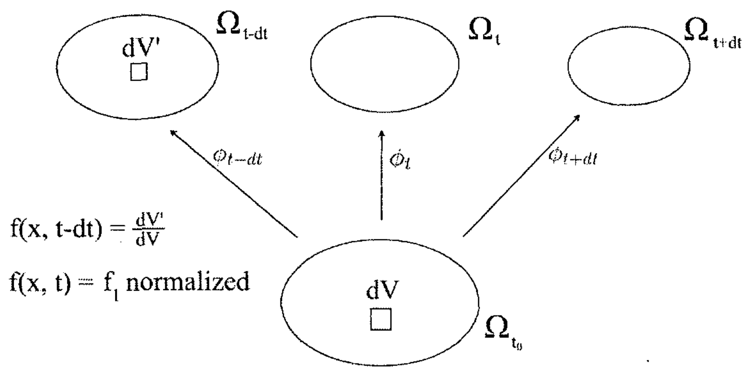


Figure 2. Monitor function for moving domains.

Then we let a time-dependent function f_1 be defined by

$$f_1 := 1 - t + t\bar{f}, \qquad \text{for } t_0 = 0 \le t \le 1 = T, \tag{4.2}$$

i.e.,

$$f_1 := (1-t)\left(1-\bar{f}\right) + \bar{f}, \qquad \text{for } t_0 = 0 \le t \le 1 = T, \tag{4.3}$$

or, we may take, for faster adjustment towards $\bar{f}$,

$$f_1 := (1-t)^2\left(1-\bar{f}\right) + \bar{f}, \qquad \text{for } t_0 = 0 \le t \le 1 = T. \tag{4.4}$$

If Ω is a fixed domain, then the monitor function f is defined to be equal to f_1 (i.e., $f = f_1$). If Ω has a moving boundary (see Figure 2), then the monitor function at time $t - dt$ is defined by

$$f(\mathbf{x}, t - dt) = \frac{\mathrm{dV}'}{\mathrm{dV}}, \tag{4.5}$$

where dV, dV$'$ are the element volumes at t_0 and $t - dt$, respectively (assuming that they have been calculated already). At time t, it is defined by

$$f(\mathbf{x}, t) = f_1 \text{ normalized according to (2.2).} \tag{4.6}$$

The RHS of the Div Equation

Now the right-hand side of (2.4a) can be calculated

$$\text{RHS} = -\frac{\partial}{\partial t}\left(\frac{1}{f(\mathbf{x},t)}\right) \approx -\frac{(1/f(\mathbf{x},t) - 1/f(\mathbf{x},t-dt))}{dt}. \tag{4.7}$$

Solving the Transport ODE

The deformation ODE in (2.5) is solved by Euler's method.

For each $\mathbf{x} \in \Omega(t_0)$,

$$\phi(\mathbf{x}, t + dt) = \phi(\mathbf{x}, t) + f(\phi(\mathbf{x}, t), t)\mathbf{u}(\phi(\mathbf{x}, t), t)\, dt, \tag{4.8}$$

i.e.,

$$\mathbf{x}_{\text{new}} = \mathbf{x}_{\text{old}} + f(\mathbf{x}_{\text{old}}, t)\mathbf{u}(\mathbf{x}_{\text{old}}, t)\, dt, \tag{4.9}$$

where

$$\mathbf{x}_{\text{new}} \in \Omega(t + dt) \qquad \text{and} \qquad \mathbf{x}_{\text{old}} \in \Omega(t). \tag{4.10}$$

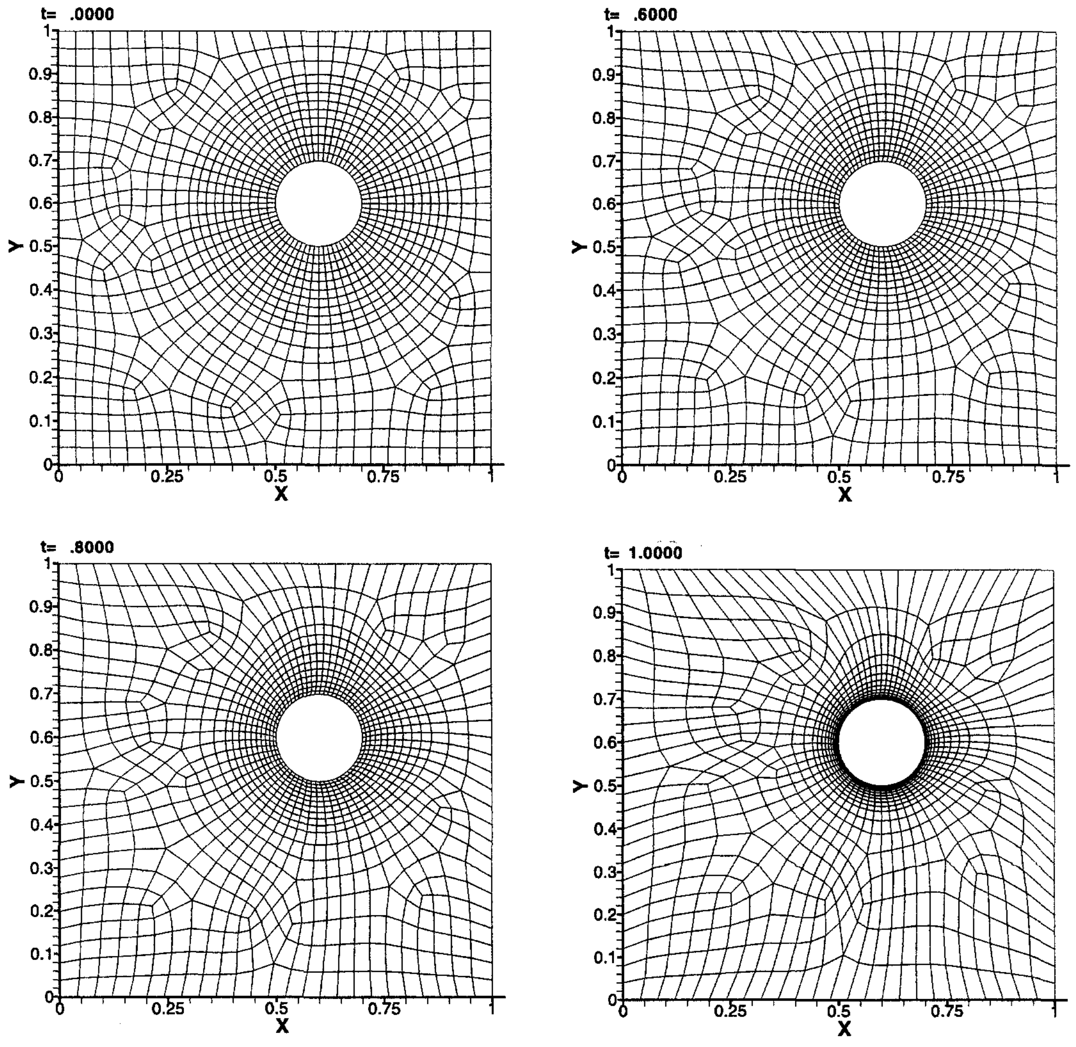


Figure 3. Grid adapted to a circle at (0.6, 0.6) with radius 0.1.

5. NUMERICAL EXAMPLES

EXAMPLE 5.1. Let Ω be the domain inside the unit square and outside the circle centered at $(0.6, 0.6)$ with radius 0.1. An unstructured quadrilateral grid on Ω is deformed according to

$$\bar{f} = \begin{cases} 0.05 - \dfrac{0.95d}{0.4}, & -0.4 \le d < 0, \\ 0.05 + \dfrac{0.95d}{0.4}, & 0 \le d \le 0.4, \\ 1, & |d| > 0.4, \end{cases} \tag{5.1}$$

where $d = 5(x - 0.6)^2(y - 0.6)^2$. See Figure 3.

EXAMPLE 5.2. Let $\Omega(t)$ be the image of the unit square $[0,1] \times [0,1]$ as the top boundary oscillates according to

$$y(x,t) = 1 + 0.1 \sin(2\pi x) \sin\left(\frac{1}{2}\pi t\right). \tag{5.2}$$

A hybrid grid on $\Omega(0)$ is deformed into a grid on $\Omega(t)$. An inflow condition is imposed on the top boundary. A slippery wall condition is imposed on the rest of the boundary. See Figure 4.

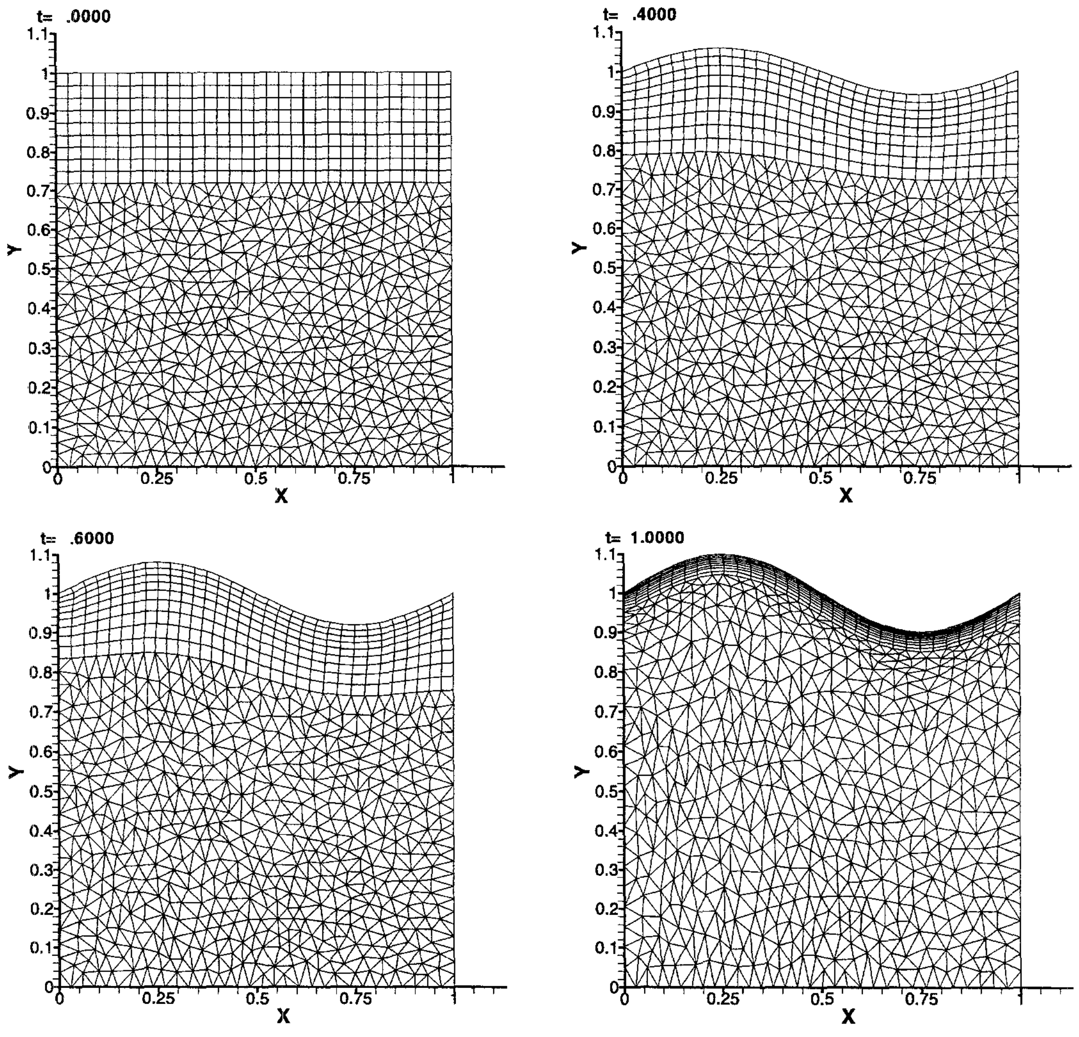


Figure 4. Grid adapted to a moving boundary.

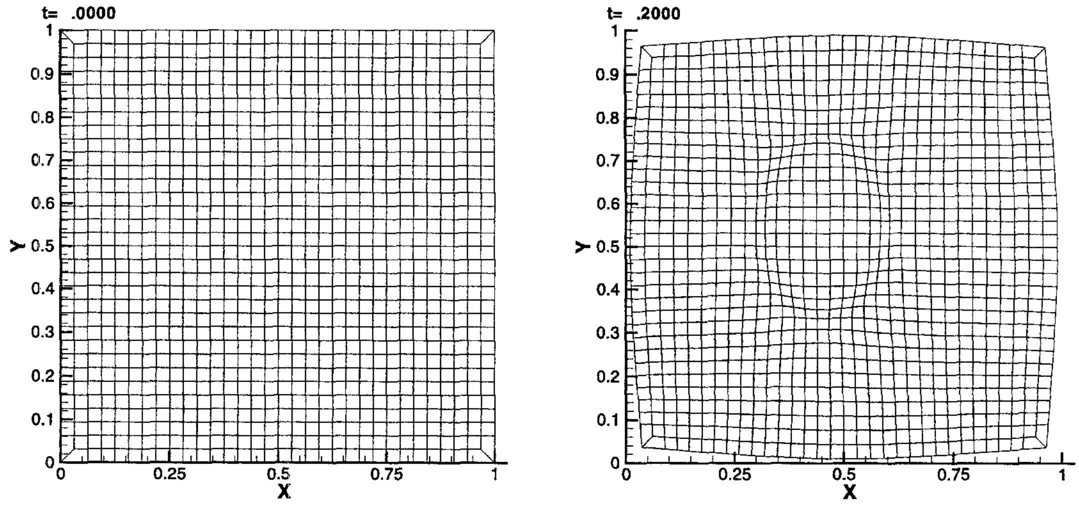
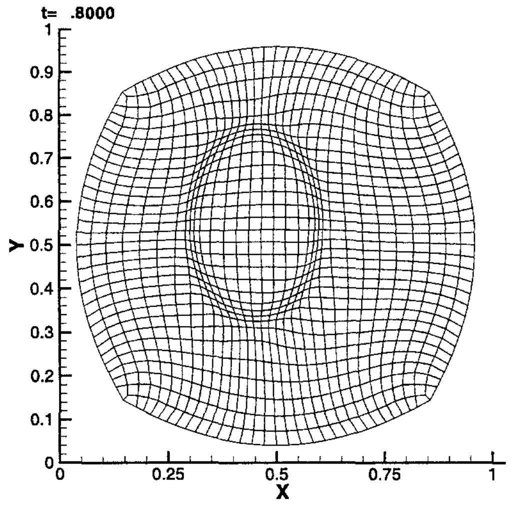


Figure 5. Grid on unit square deformed to a circle and adapted to an ellipse.



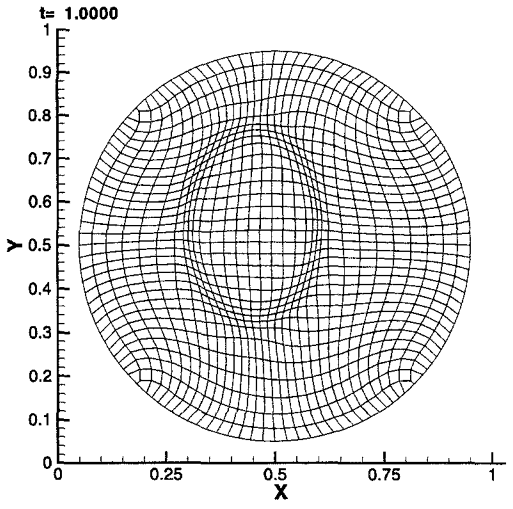


Figure 5. (cont.)

EXAMPLE 5.3. A uniform grid (except at the corners where modifications were made) on $\Omega(0) = [0,1] \times [0,1]$ is deformed to a grid of uniform size on a circle and adapted to the ellipse

$$\left(\frac{x-0.45}{0.15}\right)^2 + \left(\frac{y-0.55}{0.22}\right)^2 = 1.$$

Inflow condition is imposed on the whole boundary. See Figure 5.

6. CONCLUSION AND REMARKS

For a chosen monitor function f, the deformation method controls the cell size by making the Jacobian determinant $J(\phi) = f$. Thus, it generates grids of desirable cell sizes. Moreover, since $f > 0$, the deformed grid is nonfolding, even in 3D.

Various boundary conditions can be directly enforced with LSFEM. A nonslippery wall condition is used for fixed boundaries, which ensures that boundary nodes move along the boundary. Other boundary conditions such as inflow conditions may be enforced for free surface and moving boundary problems.

The deformation method can be used in 3D problems as well. Results in 3D will be presented in a forthcoming paper.

The LSFEM always leads to symmetric positive-definite matrices which can be efficiently solved. Parallelization of LSFEM is straightforward. Thus, large scale grids on 3D domains of complex boundaries can be efficiently adapted according to an error indicator. In fact, LSFEM can be used to solve many different types of PDEs and the residuals can be used to construct the monitor function. Thus, it is ideal to use LSFEM to solve both the host PDEs (governing the underlying physical phenomenon) and the div-curl system for deformation of the grid.

The existing deformation method is based on solving a potential ψ from a Poisson equation. The node velocity then is chosen to be $f\nabla\psi$. This method can only be used in fixed domains due to the inability of imposing Dirichlet boundary condition on $\nabla\psi$. In fact, the Neumann boundary condition for Poisson equation is used on a fixed domain, and consequently, $\nabla\psi \cdot \mathbf{n} = 0$, where $\mathbf{n}$ is the outward unit normal vector to the boundary. Thus, boundary nodes will remain on the boundary. If the Dirichlet boundary condition is used for the Poisson equation, we will still have no control of $\nabla\psi$ on the boundary. This explains why the method based on the Poisson equation works on fixed domains only. The method described in this paper is based on solving directly a vector field $\mathbf{v}$ from a div-curl system by LSFEM. A main benefit of this method is that it allows us to impose various boundary conditions including slippery wall condition (which is equivalent to the existing Poisson equation method with the Neumann boundary condition) and inflow condition, which allows the method to be used on domains with moving boundaries. Indeed, an initial grid on the initial domain will be deformed into a moving grid on the subsequent domain at any time t, once the new location of the boundary nodes are known. Another advantage is that the velocity components are computed directly from the div-curl system, instead of obtaining them by a numerical differentiation of the potential from the Poisson equation, which may decrease the order of accuracy.

REFERENCES

1. J.F. Thompson, U.A. Warsi and C.W. Mastin, *Numerical Grid Generation*, North Holland, (1985).
2. P. Knupp and S. Steinberg, *The Fundamentals of Grid Generation*, CRC Press, Boca Raton, FL, (1993).
3. V.D. Liseikin, *Grid Generation Methods*, Springer, (1999).
4. G. Carey, *Computational Grid Generation, Adaptation and Solution Strategies*, Taylor and Francis, Washington, DC, (1997).
5. P. Bochev, G. Liao and G. dela Pena, Analysis and computation of adaptive moving grids by deformation, *Numer. Meth. PDEs* **12**, (1996).
6. J. Moser, Volume elements of a Riemann manifold, *Trans AMS* **120**, (1965).
7. G. Liao and D.A. Anderson, A new approach to grid generation, *Appl. Anal.* **44**, (1992).
8. B. Semper and G. Liao, A moving grid finite element method using grid deformation, *Numer. Meth. Part. Diff. Eq.* **11**, 603, (1995).
9. F. Liu, S. Ji and G. Liao, An adaptive grid method with cell-volume control and its applications to Euler flow calculations, *SIAM J. Sci. Comput.* **20**, 811–825, (1998).
10. G. Liao, Z. Lei, G.C. de la Pena and D. Anderson, A moving finite difference method for partial differential equations, (2002).
11. G. Liao, F. Liu, G. dela Pena, D. Peng and S. Osher, Level-set based deformation methods for adaptive grids, *J. Compt. Phys.* **159**, 103–122, (2000).
12. G.J. Fix and M.D. Gunzburger, On least squares approximations to indefinite problems of the mixed type, *Int. J. Numer. Meth. Engrg.* **12**, 453–469, (1978).
13. G.J. Fix, M.D. Gunzburger and R.A. Nicolaides, On finite element methods of the least squares type, *Computers Math. Applic.* **5**, 87–98, (1979).
14. T.F. Chen and G.J. Fix, Least-squares finite element simulation of transonic flows, *Appl. Numer. Math.* **2**, 399–408, (1986).
15. B.-N. Jiang and G.F. Carey, Least-squares finite element method for compressible Euler equations, *Int. J. Numer. Meth. Fluids* **10**, 557–568, (1990).
16. B.-N. Jiang and C.L. Chang, Least-squares finite elements for Stokes problem, *Comput. Meth. Appl. Mech. Engrg.* **78**, 297–311, (1990).
17. B.-N. Jiang, *The Least-Squares Finite Element Method: Theory and Applications in Computational Fluid Dynamics and Electromagnetics*, Springer, (1998).
18. G. Liao, T. Pan and J. Su, A numerical grid generator based on Moser's deformation method, *Numer. Meth. PDEs* **10**, (1994).