THE LEAST-SQUARES FINITE ELEMENT METHOD FOR GRID DEFORMATION AND MESHFREE APPLICATIONS

by

DIONISIO LAEBER FLEITAS

Presented to the Faculty of the Graduate School of

The University of Texas at Arlington in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT ARLINGTON

August 2005

To my wife Luciana for her love and support.

ACKNOWLEDGEMENTS

I thank my Lord Jesus for his guidance in all aspects of my life and for salvation. I want to thank Dr. Guojun Liao for being my supervising advisor and for his invaluable advice, encouragement and support during my graduate studies. I wish to thank Dr. Hristo Kojouharov, Dr. Chaoqun Liu, Dr. Jianzhong Su and Dr. Hua Shan for taking some of their precious time to serve in my dissertation committee. I want to express my gratitude to the Mathematics Department at The University of Texas at Arlington for the help and also the financial support through teaching assistantships and instructorships during my graduate course. I want to thank Mr. Stan and Mrs. Alice Armentrout, who I consider my American parents, for their care and patience in teaching me the English language and the American culture. Finally, I want to thank my wife Luciana for her patience, love and support in the decisions I make, and my son Matthew who gives me so much joy but is too young to understand what I am doing. I would like to express my gratitude to my parents Dionisio and Ildeléia for their encouragement in pursuing higher degrees. I also thank Pr. Alex Silva and my brothers and sisters of my church for their prayer and support.

June 21, 2005

ABSTRACT

THE LEAST-SQUARES FINITE ELEMENT METHOD FOR GRID DEFORMATION AND MESHFREE APPLICATIONS

Publication No. _____

DIONISIO LAEBER FLEITAS, Ph.D.

The University of Texas at Arlington, 2005

Supervising Professor: Guojun Liao

Grid adaptation is often needed to improve the numerical solution of a Partial Differential Equation (PDE), due to, for example, shock waves and boundary layers. In moving boundary problems, the grid needs to be regenerated or adapted to fit the new domain. In this work, a LSFEM deformation method is developed for grid generation on fixed or moving domains. The LSFEM is a finite-elements method which seeks to minimize the PDE residual equation through the least-squares method. A new class of numerical methods currently being researched is the meshfree methods, in which the main goal is to numerically solve PDEs without the node connectivity. The LSFEM and the meshfree concept can be combined using ideas from current meshfree methods. In the LSFEM, it is important to have enough residual equations from the discretization of the variation equations to obtain an overdetermined system. In some cases, however, this requirement may not be satisfied, or if it is, the system may be extremely overdetermined. Using the meshfree concept, overlapping elements can be created to obtain enough residual equations to meet the right conditions.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

The design and study of physical models can be accomplished by experimental, theoretical and numerical methods. Because of latest advancements of computer hardware and numerical algorithms the numerical approach has increasingly become the preferred method.

Basically, a partial differential equation (PDE) that governs a certain problem can be numerically solved by discretizing the domain of the PDE into nodes forming a computational grid and approximating the solution of the PDE at those nodes. Different methods yield different approximations of the solution. The accuracy of these approximations depends on the method of solving the PDE and the quality of the computational grid. General methods of grid generation can be found in Thompson et al. [17], Knupp and Steinberg [11] and Carey [2].

Most numerical methods of solving PDEs are essentially based on the finite element method (FEM). The domain is discretized into nodes and, by connecting these nodes, small regions are formed yielding a mesh. A variation approach is used to convert the problem into a minimization problem. The minimum will be the numerical solution of the problem. Some of the advantages of the FEM are: flexibility in discretizing domains with complex geometry; well suited for unstructured grids. See [2] for more information in this area.

The Least-Squares Finite Element Method (LSFEM) is a finite-elements method which seeks to minimize the PDE residual equation through the least-squares method. One of the significant features of the LSFEM is the universality of the method: the

LSFEM has a unified numerical formulation for all types of PDEs. More details can be found in [9].

A new class of numerical methods have been currently researched and developed: the meshfree methods, in which the main goal is to numerically solve PDEs on a set of nodes without the need of the connectivity. An overview of meshfree methods can be found in [5]. Some of these methods are not completely meshfree since they can use a background mesh or a temporary mesh. The LSFEM and the meshfree concept can be combined using the ideas from the meshless finite element method [8], the particle finite element [6] and the least-squares finite element collocation method [9]. The integrals from the LSFEM variational equations can be difficult to compute, therefore a numerical integration scheme, such as gaussian quadrature, is utilized. To find a LSFEM solution, it is important to have enough residual equations to obtain an overdetermined system. This can be accomplished according to the number of nodes, elements and gaussian points in the numerical integration. However the requirement of having an overdetermined system may not be satisfied or if it is, the system may be extremely overdetermined. Using the meshfree concept, overlapping elements can be created to obtain enough residual equations to meet the right conditions.

In any case, the construction of the computational grid is of extreme importance. Inaccurate and unstable results can be consequences of a grid that is not well suited to the problem. Grid adaptation is often needed to improve the numerical solution due to, for example, shock waves and boundary layers. In moving boundary problems, the grid needs to be regenerated or adapted after each time iteration to fit the new domain.

The main approaches to grid adaptation are grid refinement and moving grid. Grid refinement is performed by adding new nodes to the current grid. This technique is most used for local refinement where nodes are inserted in regions of the domain where salient features of the PDE occur. These features may propagate and disappear as the solution

evolves. Nodes can be added, removed and re-inserted in other regions of the domain as needed, according to the local accuracy or error measures.

The moving grid approach is performed by moving the nodes to new locations of the domain according to the numerical solution as it is being computed and, for moving boundary problems, to the geometry of the new domain.

In this work, the deformation method for grid generation developed by Liao and Anderson [12] was utilized. It is based on the idea of a deformation method by Moser [15]. In the grid deformation method a monitor function is used to move the nodes. It can be constructed from the residual of the numerical scheme used or gradient of the solution obtained [4]. In the grid deformation method a first order PDE is used to generate a vector field that moves the nodes.

The grid deformation method with the LSFEM has great advantages: (1) Same LSFEM scheme can be used to solve the deformation PDE and the main PDEs (governing the underlying physical phenomenon); (2) For moving boundary problems, the Dirichlet and slippery wall conditions are imposed directly; (3) Unlike other methods where a subsequent numerical differentiation on the potential is required to obtain the vector field, the LSFEM solves the vector field directly; (4) The method can be applied to deformation PDEs with non-zero curl; (5) The LSFEM discretization leads to a symmetric positive-definite matrix with can be efficiently solved.

This dissertation has three main parts. First, chapters 2 and 3 describe the grid deformation method and the LSFEM. Secondly, the implementation of the grid deformation method with the LSFEM is outlined in chapter 4 and numerical examples are given in chapter 5. Lastly, chapter 6 outlines the LSFEM with the meshfree concept through numerical examples.

CHAPTER 2

THE GRID DEFORMATION METHOD

A preliminary version of this method appeared in [1], which is formulated and demonstrated for adaptation towards steady features on fixed domains. The version developed in this work 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 [15]. It was re-formulated for grid generation in [12]. 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 $\Omega(t) \subset D \subset \mathbb{R}^n$ for t in $[t_0, T]$ and that a monitor function

$$f : D \times [t_0, T] \rightarrow \mathbb{R}^+$$

was formed, i.e.,

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

(e.g., $f(\boldsymbol{x}, t) = \frac{C}{\delta(\boldsymbol{x},t)}$ for some $C > 0$ where $\delta(\boldsymbol{x}, t)$ is an error estimator). Moreover, suppose that

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

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

$$\det \nabla \phi(\boldsymbol{x}, t) = f(\phi(\boldsymbol{x}, t), t) \quad \text{for} \quad t_0 \leq t \leq T. \tag{2.3}$$

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

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

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

$$\operatorname{curl} \boldsymbol{u}(\boldsymbol{x},t) = \boldsymbol{r}(\boldsymbol{x},t) \quad , \boldsymbol{x} \in \Omega(t), \tag{2.4b}$$

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

for $t_0 \leq t \leq T$, where $\boldsymbol{n}$ is the outward normal to $\partial\Omega(t)$ and $\boldsymbol{g}$ is a boundary vector field determined by the boundary movement. In the numerical examples throughout this work the vector function $\boldsymbol{r}$ is taken to be zero.

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

$$\frac{\partial}{\partial t}\phi(\boldsymbol{x},t) = \boldsymbol{v}(\phi(\boldsymbol{x},t),t) \quad \text{, for } \; t_0 \leq t \leq T \tag{2.5}$$

where $\boldsymbol{v}(\phi(\boldsymbol{x},t),t) = f(\phi(\boldsymbol{x},t),t)\,\boldsymbol{u}(\phi(\boldsymbol{x},t),t)$, for $t_0 \leq t \leq T$.

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

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

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

The theorem is proved by showing that $\frac{d}{dt}\left(\frac{J(\phi)}{f(\phi,t)}\right) = 0$, and therefore $Jf = 1$ if $\left.\frac{J(\phi)}{f(\phi,t)}\right|_{t=t_0} = 1$. Details of the proof can be found in [16].

This method has been applied to calculations of flows in [13] and [3]. A version of the method is developed in [14] with the use of a level set method.

CHAPTER 3

THE LEAST-SQUARES FINITE ELEMENT METHOD (LSFEM)

3.1 Formulation of the LSFEM

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 $\boldsymbol{u}$ that minimizes the functional

$$I(\boldsymbol{v}) = \| A\boldsymbol{v} - \boldsymbol{f} \|_0^2$$

over an appropriate subspace $\mathcal{V}$ (e.g., $\boldsymbol{H}^1(\Omega)$) of the Hilbert space $\boldsymbol{L}_2(\Omega) = [L_2(\Omega)]^m$ is sought within the constraint of a given boundary condition.

Consider the linear boundary-value problem:

$$A\boldsymbol{u} = \boldsymbol{f} \quad \text{in } \Omega \tag{3.1a}$$

$$B\boldsymbol{u} = \boldsymbol{g} \quad \text{on } \Gamma \tag{3.1b}$$

where

$$A\boldsymbol{u} = \sum_{i=1}^{n_d} A_i \frac{\partial \boldsymbol{u}}{\partial x_i} + A_0 \boldsymbol{u}, \ B \text{ is a boundary operator} \tag{3.2}$$

and $\boldsymbol{x} = (x_1, x_2, ..., x_{n_d})$ for $n_d = 1, 2 \text{ or } 3$,

$$\boldsymbol{u} = \begin{pmatrix} u_1 \\ u_2 \\ \cdots \\ u_m \end{pmatrix} \quad , \boldsymbol{f} = \begin{pmatrix} f_1 \\ f_2 \\ \cdots \\ f_{N_{\text{eq}}} \end{pmatrix} \quad and \ \boldsymbol{g} = \begin{pmatrix} g_1 \\ g_2 \\ \cdots \\ g_{N_{\text{eq}}} \end{pmatrix} \tag{3.3}$$

where m is the number of variables at each node and N_{eq} is number of equations in (3.1a).

Without loss of generality, assume that $\boldsymbol{g} = 0$. Let $\boldsymbol{R} = A\boldsymbol{v} - \boldsymbol{f}$ in Ω for an arbitrary test function $\boldsymbol{v} \in \mathcal{V}$. The distance between $A\boldsymbol{v}$ and $\boldsymbol{f}$ is given by

$$\|\boldsymbol{R}\|_0^2 = \int_\Omega (A\boldsymbol{v} - \boldsymbol{f})^2 \, d\omega \geq 0 \tag{3.4}$$

where

$$\|\boldsymbol{R}\|_0^2 = \sum_{i=1}^{N_{\text{eq}}} \|R_i\|_0^2 = \sum_{i=1}^{N_{\text{eq}}} \int_\Omega R_i^2 d\omega = \int_\Omega \boldsymbol{R} \cdot \boldsymbol{R} \, d\omega = \int_\Omega \boldsymbol{R}^2 \, d\omega. \tag{3.5}$$

Since $\boldsymbol{R}$ is not zero we have $\|\boldsymbol{R}\|_0^2 \geq 0$, and the equality holds only if $\boldsymbol{v}$ is the exact solution of (3.1). The solution $\boldsymbol{u}$ to (3.1) can be viewed as an element of $\mathcal{V}$ that minimizes the L_2 distance between $A\boldsymbol{v}$ and $\boldsymbol{f}$:

$$I(\boldsymbol{v}) = \|A\boldsymbol{v} - \boldsymbol{f}\|_0^2 = \int_\Omega (A\boldsymbol{v} - \boldsymbol{f})^2 d\omega \text{ on } V. \tag{3.6}$$

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

$$\lim_{t \to 0} \frac{\mathrm{d}}{\mathrm{d}t} I(\boldsymbol{u} + t\boldsymbol{v}) = 0 \quad \forall \, \boldsymbol{v} \in \mathcal{V}. \tag{3.7}$$

We have

$$I(\boldsymbol{u}+t\boldsymbol{v}) = \int_\Omega (A\,(\boldsymbol{u}+t\boldsymbol{v}) - \boldsymbol{f}\,)^2 d\omega = \\ = \int_\Omega \left[(A\boldsymbol{u})^2 + t^2\,(A\boldsymbol{v})^2 + \boldsymbol{f}^{\,2} + 2t\,(A\boldsymbol{u})\cdot(A\boldsymbol{v}) - 2\,(A\boldsymbol{u})\cdot\boldsymbol{f} - 2t\,(A\boldsymbol{v})\cdot\boldsymbol{f}\,\right] d\omega, \tag{3.8}$$

then

$$\lim_{t\to 0}\frac{\mathrm{d}}{\mathrm{d}t} I(\boldsymbol{u}+t\boldsymbol{v}) = \int_\Omega \left[2\,(A\boldsymbol{u})\cdot(A\boldsymbol{v}) - 2\,(A\boldsymbol{v})\cdot\boldsymbol{f}\,\right] d\omega = \\ = 2\int_\Omega (A\boldsymbol{v})\cdot(A\boldsymbol{u}-\boldsymbol{f}\,)\,d\omega = 0. \tag{3.9}$$

Hence

$$\int_\Omega (A\boldsymbol{u})\cdot(A\boldsymbol{v})\;d\omega = \int_\Omega \boldsymbol{f}\cdot(A\boldsymbol{v})\;d\omega \quad \forall\,\boldsymbol{v}\in\mathcal{V}. \tag{3.10}$$

Assuming that A is bounded below, a discretization of (3.10) leads to a symmetric positive-definite matrix. Subdividing the domain into a union of finite elements we use the expansion of $\boldsymbol{u}_h$ in each element:

$$\boldsymbol{u}_h^e(\boldsymbol{x}) = \sum_{j=1}^{N_n} \left[\psi_j(\boldsymbol{x}) \begin{pmatrix} u_{1j} \\ u_{2j} \\ \cdots \\ u_{mj} \end{pmatrix} \right], \tag{3.11}$$

where u_{ij} is the nodal value of u_i at the jth node, ψ_j's are the shape functions, and N_n is the number of nodes in an element. Using (3.11) in (3.10), we obtain a linear system of algebraic equations:

$$KU = F \tag{3.12}$$

where K and F are assembled from element matrices

$$K_e = \int_{\Omega_e} (A\psi_1, A\psi_2, ..., A\psi_{N_n})^T (A\psi_1, A\psi_2, ..., A\psi_{N_n})\, d\omega \tag{3.13a}$$

$$F_e = \int_{\Omega_e} (A\psi_1, A\psi_2, ..., A\psi_{N_n})^T \boldsymbol{f}\, d\omega \tag{3.13b}$$

The boundary conditions (3.1b) can also be included into (3.6). The discretization by LSFEM always leads to symmetric positive-definite matrices (see [9]) which can be efficiently solved. It is not necessary to assemble the sparse matrix K if an iterative method is used. In the numerical examples, the Conjugate Gradient Method (CGM) (see [7]), which is an effective iterative method for symmetric positive-definite matrices, was used to obtain the LSFEM solution. In the CGM method, the iterates $U^{(k)}$ are updated by a search direction vector. By choosing an appropriate vector, a certain quadratic form is minimized, where the minimum is the solution of the algebraic system. The search vector is defined in terms of the residual vector $KU^{(k-1)} - F$. The matrix-vector operations are carried out at the element level then accumulated into the global vector $U^{(k)}$. This approach is called the Element-by-Element technique.

3.2 Evaluation of the Element Matrices

The exact evaluation of the integrals in (3.13) can be difficult. Gaussian integration is used for the numerical approximation of these element matrices.

In the LSFEM computation with gaussian quadrature the integral (3.5) is approximated by

$$\int_\Omega \boldsymbol{R}^2 d\omega \approx \sum_{i=1}^{N_{\text{elem}}} \left(\sum_{l=1}^{N_{\text{G}}} w_l \boldsymbol{R}^2(\boldsymbol{\xi}_l)\, |J_{e_i}(\boldsymbol{\xi}_l)| \right) , \tag{3.14}$$

where $N_{\text{elem}}$is the number of elements, w_l is the gaussian weighting factor, N_{G} is the number of Gaussian points, $\boldsymbol{\xi}_l$ is the location of the Gaussian points in the master element

and J_{e_i} is the Jacobian determinant of the coordinate transformation of the element e_i to its master element.

Therefore, the minimization of (3.6):

$$I(v) = \|\boldsymbol{R}\|_0^2 = \int_\Omega \left(A\boldsymbol{v} - \boldsymbol{f}\right)^2 d\omega \tag{3.15}$$

is equivalent to minimizing the sum of weighted residuals

$$I\left(\boldsymbol{v}_h\right) = \sum_{i=1}^{N_{\text{elem}}} \left(\sum_{l=1}^{N_{\text{G}}} w_l \boldsymbol{R}^2(\boldsymbol{\xi}_l) \left| J_{e_i}(\boldsymbol{\xi}_l) \right| \right). \tag{3.16}$$

The minimum of (3.16) is the least-squares solution of the corresponding algebraic system from the LSFEM discretization. In order for the LSFEM solution to be found we need to solve an overdetermined system of residual equations (see Appendix). A necessary condition of this requirement is that

$$N_{\text{elem}} \times N_{\text{G}} \times N_{\text{eq}} > N_{\text{node}} \times m - N_{\text{bc}} \tag{3.17}$$

where N_{node} is the number of nodes, m is the number of components of $\boldsymbol{u}$, N_{eq} is the number of equations in the first order PDE system (3.1) and N_{bc} is the number of boundary condition values. Let $\lambda = N_{\text{elem}} \times N_{\text{G}} \times N_{\text{eq}} - \left(N_{\text{node}} \times m - N_{\text{bc}}\right)$, then, for the existence of the LSFEM solution, it is necessary that

$$\lambda = N_{\text{elem}} \times N_{\text{G}} \times N_{\text{eq}} - \left(N_{\text{node}} \times m - N_{\text{bc}}\right) > 0. \tag{3.18}$$

CHAPTER 4

IMPLEMENTATION OF THE LSFEM GRID DEFORMATION

4.1 LSFEM and the deformation method for grid generation

In the grid deformation method a first-order PDE system required to construct the node velocity used to move the nodes. The PDE system is easily solved by the LSFEM.

For grids with fixed domain, slippery wall condition (i.e, $\boldsymbol{u} \cdot \boldsymbol{n} = \mathbf{0}$ 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.

This work further develops the ideas that first appeared in [1] and gives numerical examples on domains with moving boundaries. The LSFEM grid deformation method consists of the following steps:

1. Define monitor function f and form the right hand side of the div equation.
2. Solve div-curl system (2.4) by LSFEM at each time step.
3. Solve for the new node location from the deformation ODE (2.5).

4.2 The Monitor Function

The monitor function can be constructed from the residual of the LSFEM or other numerical scheme used to solve the governing equations. Details on this approach can be found in [4].

For movement towards an interface (see Figure 4.1) or a boundary I (on a fixed or moving domain), we first construct a function $\overline{f}$ such that

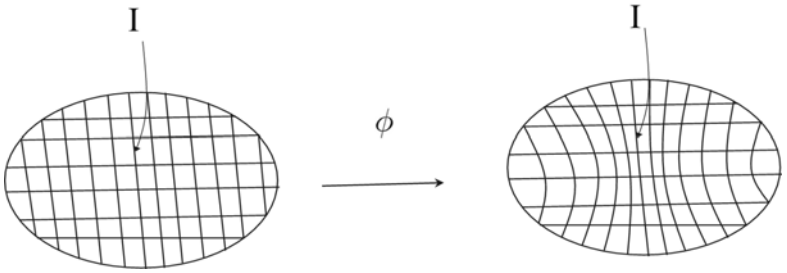

Figure 4.1. Adaptation towards an interface I.

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

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

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

i.e.,

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

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

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

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 4.2), then the monitor function at time $t - dt$ is defined by

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

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

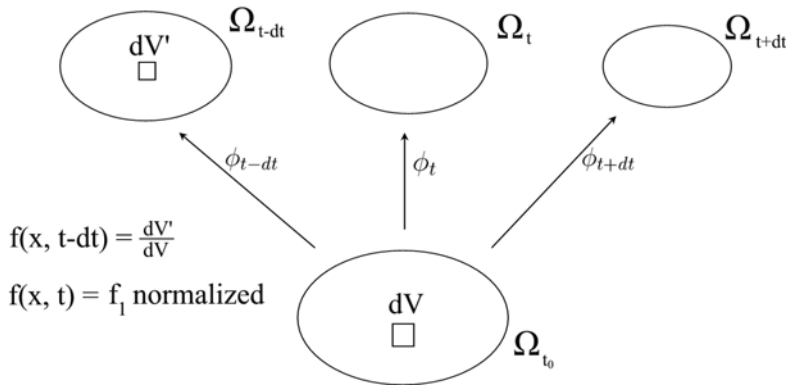

Figure 4.2. Monitor function for moving boundary.

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

For time dependent problems, the governing equations, the deformation PDE and ODE are solved in real time t. After each time step t, the monitor function $\overline{f}$ in (4.1) is updated by being defined in terms of the residual (or other error indicator) from the numerical scheme used to solve the governing equations.

Another approach is to carry out the grid deformation method between two time steps t and $t + dt$. After solving the governing equations at time t, an artificial time s is used to deform the grid from Ω_t to Ω_{t+dt}, see Figure 4.3.

4.3 Solving the div-curl system

A backward-difference approximation can be used for the right hand side of (2.4a):

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

Now, the implementation of the 3D least-squares finite element method to compute the vector field in (2.4) is described. The 2D implementation is easier and straightforward. For 3D we have

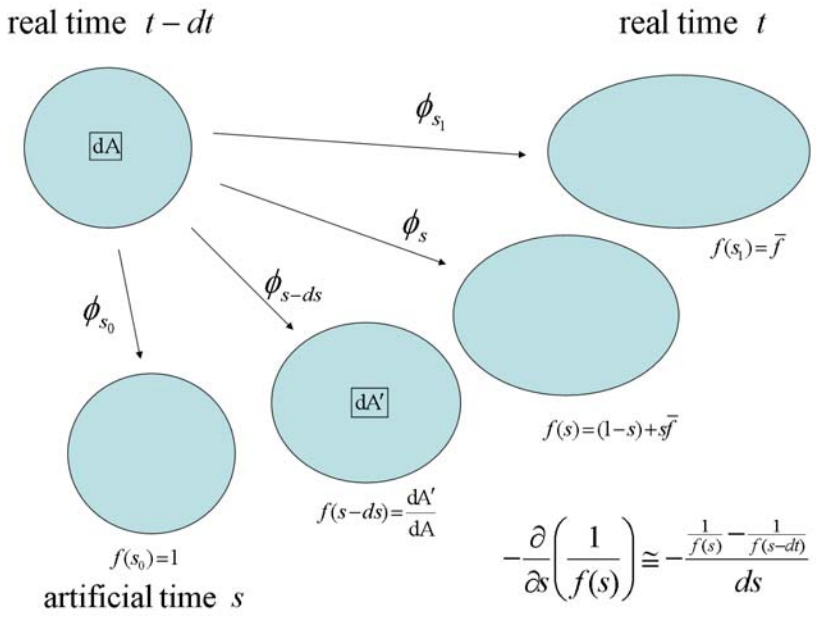

Figure 4.3. An artificial time s used for intermediate deformation.

$$\boldsymbol{u}(\boldsymbol{x},t)=(u_1(x,y,z,t),u_2(x,y,z,t),u_3(x,y,z,t)).$$

The div-curl system is:

$$\frac{\partial u_1}{\partial x}+\frac{\partial u_2}{\partial y}+\frac{\partial u_3}{\partial z}=RHS \tag{4.8a}$$

$$\frac{\partial u_3}{\partial y}-\frac{\partial u_2}{\partial z}=0 \tag{4.8b}$$

$$\frac{\partial u_1}{\partial z}-\frac{\partial u_3}{\partial x}=0 \tag{4.8c}$$

$$\frac{\partial u_2}{\partial x}-\frac{\partial u_1}{\partial y}=0 \tag{4.8d}$$

for $(x,y,z)\in\Omega(t)$, noting that in the system (2.4) curl $\boldsymbol{u}$ is taken to be zero.

Written in the form $A\boldsymbol{u}=\boldsymbol{f}$ the div-curl system (4.8) becomes

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} \frac{\partial u_1}{\partial x} \\ \frac{\partial u_2}{\partial x} \\ \frac{\partial u_3}{\partial x} \end{pmatrix} + \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{pmatrix} \begin{pmatrix} \frac{\partial u_1}{\partial y} \\ \frac{\partial u_2}{\partial y} \\ \frac{\partial u_3}{\partial y} \end{pmatrix} + \begin{pmatrix} 0 & 0 & 1 \\ 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \frac{\partial u_1}{\partial z} \\ \frac{\partial u_2}{\partial z} \\ \frac{\partial u_3}{\partial z} \end{pmatrix} = \begin{pmatrix} RHS \\ 0 \\ 0 \\ 0 \end{pmatrix}. \quad (4.9)$$

In the 3D examples, linear hexahedral elements are used and the finite element expansion at each hexahedral is given by

$$\boldsymbol{u}_h^e(\boldsymbol{x}) = \sum_{j=1}^{8} \left[\psi_j(\boldsymbol{x}) \begin{pmatrix} u_{1j} \\ u_{2j} \\ u_{3j} \end{pmatrix} \right], \quad (4.10)$$

where u_{1j}, u_{2j} and u_{3j} are the nodal values of u_1, u_2 and u_3 at the jth node of the hexahedral element and ψ_j's are the shape functions.

To assemble the algebraic system $KU = F$ we use the element matrices in (3.13)

$$K_e = \int_{\Omega_e} (A\psi_1, A\psi_2, \ldots, A\psi_8)^T (A\psi_1, A\psi_2, \ldots, A\psi_8)\, d\omega \quad (4.11a)$$

$$F_e = \int_{\Omega_e} (A\psi_1, A\psi_2, \ldots, A\psi_8)^T \boldsymbol{f}\, d\omega \quad (4.11b)$$

where

$$\boldsymbol{f} = \begin{pmatrix} RHS \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad (4.12)$$

and

$$A\psi_i = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix} \frac{\partial \psi_i}{\partial x} + \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{pmatrix} \frac{\partial \psi_i}{\partial y} + \begin{pmatrix} 0 & 0 & 1 \\ 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \frac{\partial \psi_i}{\partial z}, \tag{4.13}$$

i.e.,

$$A\psi_i = \begin{pmatrix} \frac{\partial \psi_i}{\partial x} & \frac{\partial \psi_i}{\partial y} & \frac{\partial \psi_i}{\partial z} \\ 0 & -\frac{\partial \psi_i}{\partial z} & \frac{\partial \psi_i}{\partial y} \\ \frac{\partial \psi_i}{\partial z} & 0 & -\frac{\partial \psi_i}{\partial x} \\ -\frac{\partial \psi_i}{\partial y} & \frac{\partial \psi_i}{\partial x} & 0 \end{pmatrix}, \tag{4.14}$$

for $i = 1, 2, \ldots, 8$.

Therefore, the element matrices (4.11) are given by

$$K_e = \int_{\Omega_e} \begin{pmatrix} \left[(A\psi_1)^T A\psi_1 \right] & \cdots & \left[(A\psi_1)^T A\psi_8 \right] \\ \vdots & \ddots & \vdots \\ \left[(A\psi_8)^T A\psi_1 \right] & \cdots & \left[(A\psi_8)^T A\psi_8 \right] \end{pmatrix} d\Omega \tag{4.15a}$$

$$F_e = \int_{\Omega_e} \begin{pmatrix} \left[(A\psi_1)^T \boldsymbol{f} \right] \\ \vdots \\ \left[(A\psi_8)^T \boldsymbol{f} \right] \end{pmatrix} d\Omega. \tag{4.15b}$$

In a typical finite element method the element mesh Ω_e is transformed to its master element $\Omega_0 = [-1, 1]^3$ (Figure 4.4) so that the integrals in (4.15) can be easily approximated by numerical methods such as gaussian integration. In the finite element formulation the integrals are computed via a coordinate transformation, i.e.,

$$\int_{\Omega_e} \left[(A\psi_i)^T A\psi_j \right] (x,y,z) d(x,y,z) =$$

$$= \int_{\Omega_0} \left[(A\psi_i)^T A\psi_j \right] (x(\xi,\eta,\gamma), y(\xi,\eta,\gamma), z(\xi,\eta,\gamma)) \left| \det J(\xi,\eta,\gamma) \right| d(\xi,\eta,\gamma), \qquad (4.16)$$

where $J(\xi,\eta,\gamma)$ is the Jacobian of the coordinate transformation

$$x = \sum_{j=1}^{8} x_j \psi_j(\xi,\eta,\gamma), \quad y = \sum_{j=1}^{8} y_j \psi_j(\xi,\eta,\gamma) \text{ and } z = \sum_{j=1}^{8} z_j \psi_j(\xi,\eta,\gamma) \qquad (4.17)$$

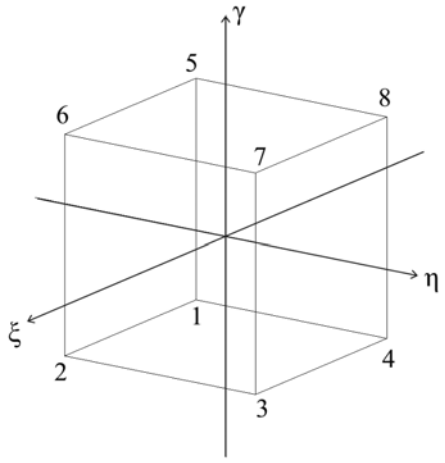

Figure 4.4. The master element Ω_0.

The shape functions $\psi_j(x,y,z)$ in (4.10) are expressed in terms of the coordinates ξ, η and γ through (4.17). Differentiating ψ_j with respect to ξ, η and γ we obtain the system

$$\begin{pmatrix} \frac{\partial \psi_j}{\partial \xi} \\ \frac{\partial \psi_j}{\partial \eta} \\ \frac{\partial \psi_j}{\partial \gamma} \end{pmatrix} = J(\xi,\eta,\gamma) \begin{pmatrix} \frac{\partial \psi_j}{\partial x} \\ \frac{\partial \psi_j}{\partial y} \\ \frac{\partial \psi_j}{\partial z} \end{pmatrix} \qquad (4.18)$$

where

$$J(\xi,\eta,\gamma) = \begin{pmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} & \frac{\partial z}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} & \frac{\partial z}{\partial \eta} \\ \frac{\partial x}{\partial \gamma} & \frac{\partial y}{\partial \gamma} & \frac{\partial z}{\partial \gamma} \end{pmatrix}. \tag{4.19}$$

Assuming that $\det J(\xi,\eta,\gamma)$ is non-zero, the partial derivatives $\frac{\partial \psi_j}{\partial x}$, $\frac{\partial \psi_j}{\partial y}$ and $\frac{\partial \psi_j}{\partial z}$ in the entries of the element matrices (4.15) are computed by

$$\begin{pmatrix} \frac{\partial \psi_j}{\partial x} \\ \frac{\partial \psi_j}{\partial y} \\ \frac{\partial \psi_j}{\partial z} \end{pmatrix} = J^{-1}(\xi,\eta,\gamma) \begin{pmatrix} \frac{\partial \psi_j}{\partial \xi} \\ \frac{\partial \psi_j}{\partial \eta} \\ \frac{\partial \psi_j}{\partial \gamma} \end{pmatrix}. \tag{4.20}$$

The shape functions ψ_j in terms of ξ, η and γ used in the 3D numerical examples are the polynomials associated to the Lagrange linear hexahedral elements, i.e.,

$$\begin{aligned}
\psi_1 &= \frac{1}{8}(1-\xi)(1-\eta)(1-\gamma), \\
\psi_2 &= \frac{1}{8}(1+\xi)(1-\eta)(1-\gamma), \\
\psi_3 &= \frac{1}{8}(1+\xi)(1+\eta)(1-\gamma), \\
\psi_4 &= \frac{1}{8}(1-\xi)(1+\eta)(1-\gamma), \\
\psi_5 &= \frac{1}{8}(1-\xi)(1-\eta)(1+\gamma), \\
\psi_6 &= \frac{1}{8}(1+\xi)(1-\eta)(1+\gamma), \\
\psi_7 &= \frac{1}{8}(1+\xi)(1+\eta)(1+\gamma), \\
\psi_8 &= \frac{1}{8}(1-\xi)(1+\eta)(1+\gamma).
\end{aligned}$$

4.3.1 Boundary Conditions

Previous grid deformation methods were based on solving a potential ω from a Poisson equation. If the Dirichlet boundary condition is used for the Poisson equation, we will still have no control of $\boldsymbol{\nabla}\omega$ on the boundary. This explains why the method based on Poisson equation works on fixed domains only. The method described in this work is based on directly finding a vector field $\boldsymbol{u}$ from a div-curl system by the LSFEM. A main benefit of this method is that it allows us to impose, directly on $\boldsymbol{u}$, various boundary conditions including slippery wall (to make the boundary nodes stay on the boundary) and inflow condition, which allows the method to be used on domains with moving boundaries.

In practical FEM programming all nodes, including the Dirichlet nodes, are included at the outset. The stiffness matrix is assembled by moving terms with known values of $\boldsymbol{u}$ to the right-hand side of the algebraic system and modifying the rows of the matrix accordingly. To enforce a slippery wall condition on the i^{th} node of an element e, we use a coordinate transformation and set u_n, the normal component of $\boldsymbol{u}$, equal to zero and the other components as unknowns. Consider the coordinate transformation M:

$$\begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix}_i = M \begin{pmatrix} u_n \\ u_r \\ u_s \end{pmatrix}_i , \tag{4.21}$$

thus,

$$K_e U_e = \begin{pmatrix} K_{e_{11}} & \cdots & K_{e_{1i}} & \cdots & K_{e_{18}} \\ \vdots & \ddots & \vdots & & \vdots \\ K_{e_{i1}} & \cdots & K_{e_{ii}} & \cdots & K_{e_{i8}} \\ \vdots & & \vdots & \ddots & \vdots \\ K_{e_{81}} & \cdots & K_{e_{8i}} & \cdots & K_{e_{88}} \end{pmatrix} \begin{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix}_1 \\ \vdots \\ M \begin{pmatrix} u_n \\ u_r \\ u_s \end{pmatrix}_i \\ \vdots \\ \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix}_8 \end{pmatrix} = \begin{pmatrix} \boldsymbol{f}_1 \\ \vdots \\ \boldsymbol{f}_i \\ \vdots \\ \boldsymbol{f}_8 \end{pmatrix}_e \tag{4.22}$$

which can be written as

$$\begin{pmatrix} K_{e_{11}} & \cdots & K_{e_{1i}}M & \cdots & K_{e_{18}} \\ \vdots & \ddots & \vdots & & \vdots \\ K_{e_{i1}} & \cdots & K_{e_{ii}}M & \cdots & K_{e_{i8}} \\ \vdots & & \vdots & \ddots & \vdots \\ K_{e_{81}} & \cdots & K_{e_{8i}}M & \cdots & K_{e_{88}} \end{pmatrix} \begin{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix}_1 \\ \vdots \\ \begin{pmatrix} u_n \\ u_r \\ u_s \end{pmatrix}_i \\ \vdots \\ \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix}_8 \end{pmatrix} = \begin{pmatrix} \boldsymbol{f}_1 \\ \vdots \\ \boldsymbol{f}_i \\ \vdots \\ \boldsymbol{f}_8 \end{pmatrix}_e . \tag{4.23}$$

To make K_e symmetric we multiply the i^{th} equation by M^T:

$$\begin{pmatrix} K_{e_{11}} & \cdots & K_{e_{1i}}M & \cdots & K_{e_{18}} \\ \vdots & \ddots & \vdots & & \vdots \\ M^T K_{e_{i1}} & \cdots & M^T K_{e_{ii}} M & \cdots & M^T K_{e_{i8}} \\ \vdots & & \vdots & \ddots & \vdots \\ K_{e_{81}} & \cdots & K_{e_{8i}}M & \cdots & K_{e_{88}} \end{pmatrix} \begin{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix}_1 \\ \vdots \\ \begin{pmatrix} u_n \\ u_r \\ u_s \end{pmatrix}_i \\ \vdots \\ \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix}_8 \end{pmatrix} = \begin{pmatrix} \boldsymbol{f}_1 \\ \vdots \\ M^T \boldsymbol{f}_i \\ \vdots \\ \boldsymbol{f}_8 \end{pmatrix}_e . \tag{4.24}$$

After solving the algebraic system we find $\boldsymbol{u}$ at node i through (4.21).

4.4 Solving the deformation ODE

The deformation ODE in (2.5) is solved by Euler's method. For each $\boldsymbol{x} \in \Omega(t_0)$,

$$\phi(\boldsymbol{x}, t + dt) = \phi(\boldsymbol{x}, t) + \boldsymbol{v}(\phi(\boldsymbol{x}, t), t)\, dt \tag{4.25}$$

where $\boldsymbol{v}(\phi(\boldsymbol{x}, t), t) = f(\phi(\boldsymbol{x}, t), t)\, \boldsymbol{u}(\phi(\boldsymbol{x}, t), t)$.

Therefore, the new grid (mapping $\phi(\boldsymbol{x}, t + dt)$) is calculated from the old grid (mapping $\phi(\boldsymbol{x}, t)$) and the velocity field found through the div-curl system (2.4) at time t:

$$\boldsymbol{x}_{\text{new}} = \boldsymbol{x}_{\text{old}} + f(\boldsymbol{x}_{\text{old}}, t)\boldsymbol{u}(\boldsymbol{x}_{\text{old}}, t)\, dt, \tag{4.26}$$

where

$$\boldsymbol{x}_{\text{new}} \in \Omega(t+dt) \quad \text{and} \quad \boldsymbol{x}_{\text{old}} \in \Omega(t). \tag{4.27}$$

The accuracy of the ODE can be improved via Runge-Kutta method of order 2:

$$\begin{aligned} \phi(\boldsymbol{x}, t+dt) &= \phi(\boldsymbol{x}, t) + \frac{1}{2}\boldsymbol{K}_1 + \frac{1}{2}\boldsymbol{K}_2, \\ \text{where } \boldsymbol{K}_1 &= dt\ \boldsymbol{v}(\phi(\boldsymbol{x}, t), t), \\ \text{and } \boldsymbol{K}_2 &= dt\ \boldsymbol{v}(\phi(\boldsymbol{x}, t) + \boldsymbol{K}_1, t+dt), \end{aligned} \tag{4.28a}$$

i.e.,

$$\phi(\boldsymbol{x}, t+dt) = \phi(\boldsymbol{x}, t) + \frac{dt}{2}\left[\boldsymbol{v}(\phi(\boldsymbol{x}, t), t) + \boldsymbol{v}(\left\{\phi(\boldsymbol{x}, t) + dt\ \boldsymbol{v}(\phi(\boldsymbol{x}, t), t)\right\}, t+dt)\right]. \tag{4.29}$$

This Runga-Kutta discretization scheme makes use of the velocity field $\boldsymbol{v}(\cdot, t+dt)$ which, at time step t, has not yet been found. We use an approximate velocity field $\widetilde{\boldsymbol{v}}(\cdot, t+dt)$ which is calculated on a temporary new grid $\widetilde{\phi}(\boldsymbol{x}, t+dt)$ created via Euler's method described above.

CHAPTER 5

NUMERICAL EXAMPLES OF THE LSFEM GRID DEFORMATION

5.1 2D unstructured grid adaptation

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

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

where $d = 0.1 - \sqrt{(x-0.6)^2 + (y-0.6)^2}$. See Figure 5.1.

5.2 2D moving grid generation and adaptation

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

$$(\frac{x-0.45}{0.15})^2 + (\frac{y-0.55}{0.22})^2 = 1. \tag{5.2}$$

Inflow condition is imposed on the whole boundary. See Figure 5.2.

5.3 3D grid adaptation on moving boundary

A uniform grid on $\Omega(0) = [0,1]^3$ is deformed according to the moving top boundary

$$z = 1 + 0.1 \sin(2\pi x) \cos(2\pi y) \sin(4\pi t) \tag{5.3}$$

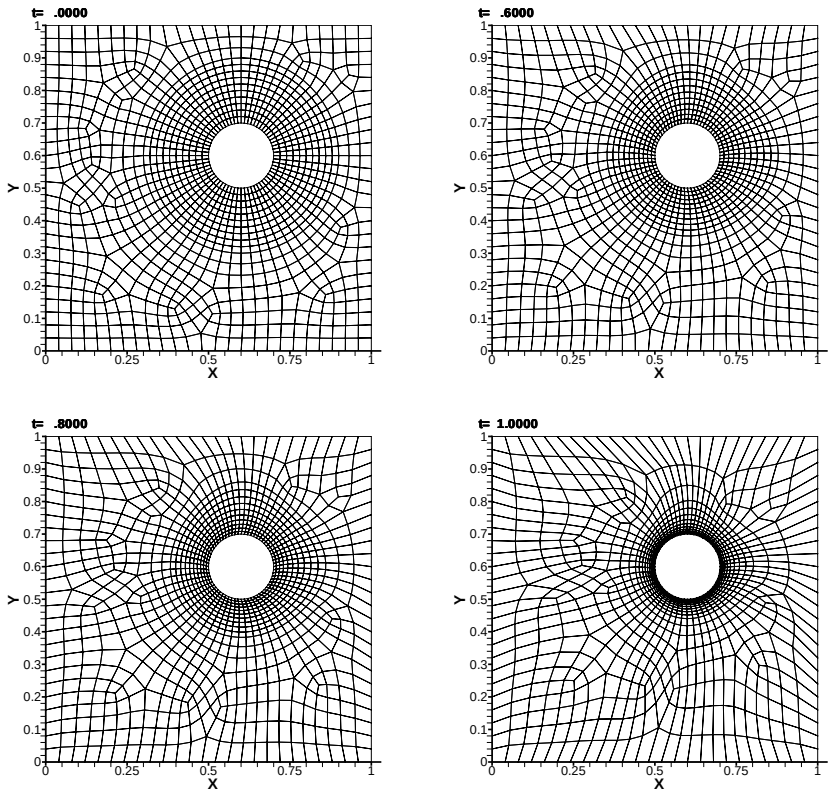

Figure 5.1. Grid adapted to a circle at (0.6,0.6) with radius 0.1.

for $0 \leq t \leq 1$. Slippery wall condition is imposed on the other boundaries. See Figure 5.3 and 5.4.

5.4 3D grid generation and adaptation

The LSFEM grid deformation method can be used to generate the first grid on a physical model from a template grid and at the same time the grid can be adapted to a certain feature. In this example a template grid on $\Omega(0) = \left([0,1]^2 -]0.3, 0.7[^2\right) \times [0,1]$ is deformed to generate a grid on the thick-walled cylinder

$$0.15^2 \leq (x-0.5)^2 + (y-0.5)^2 \leq 0.45^2 \text{ and } 0 \leq z \leq 1.$$

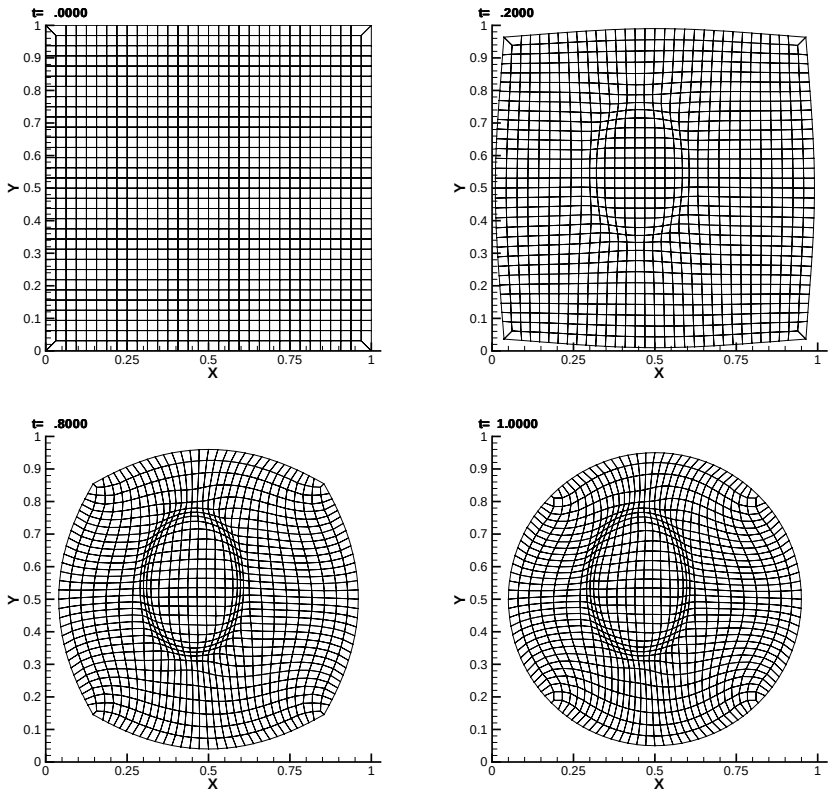

Figure 5.2. Grid on unit square deformed to a circle and adapted to an ellipse.

The boundary moves towards the cylinder boundary while the grid is adapted to the sphere

$$(x-0.8)^2+(y-0.8)^2+(z-0.9)^2=0.4^2 \tag{5.4}$$

On the edges Dirichlet conditions were applied. On the top and bottom boundaries (except the edges) $\boldsymbol{u}\cdot\boldsymbol{n}=\boldsymbol{0}$. On the sides, Dirichlet conditions were enforced only on the x and y components of $\boldsymbol{u}$. See Figures 5.5-5.8. Another approach is to generate the grid on the cylinder from the template before adapting the grid. In this case Dirichlet conditions are used to generate the grid on the cylinder, then for the adaptation, $\boldsymbol{u}\cdot\boldsymbol{n}=0$ on the whole boundary. See Figures 5.9 and 5.10.

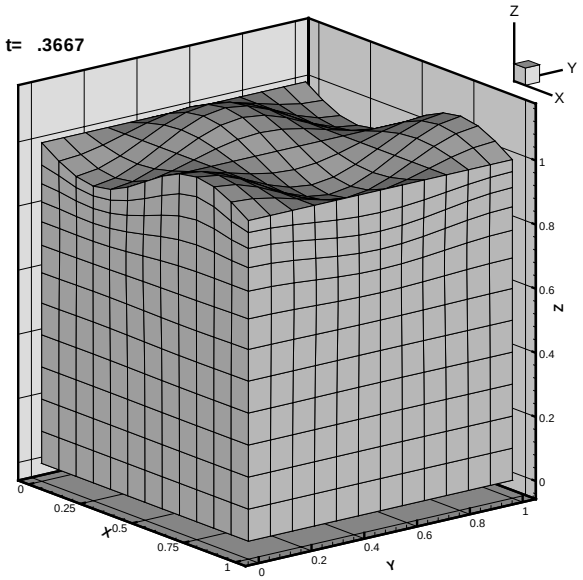

Figure 5.3. Moving top at $t = \frac{11}{30}$.

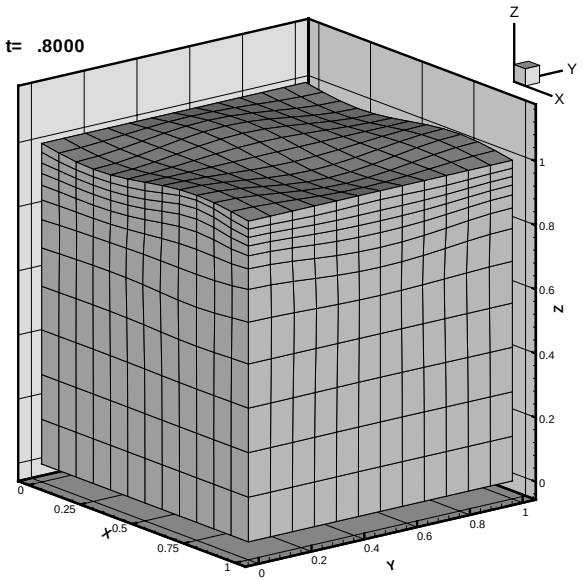

Figure 5.4. Moving top at $t = 0.8$.

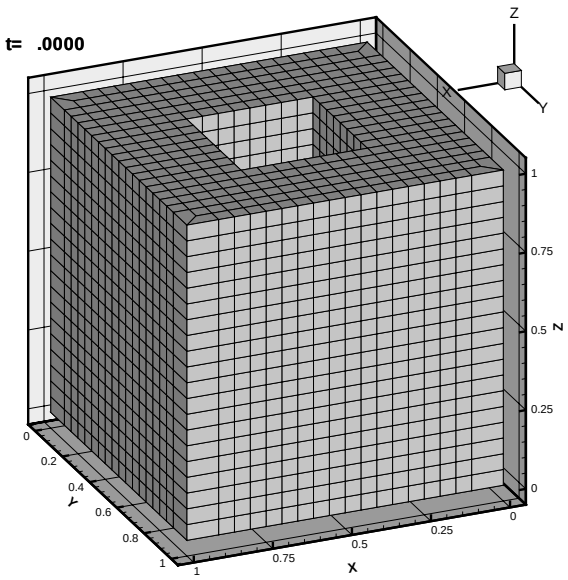

Figure 5.5. Template grid.

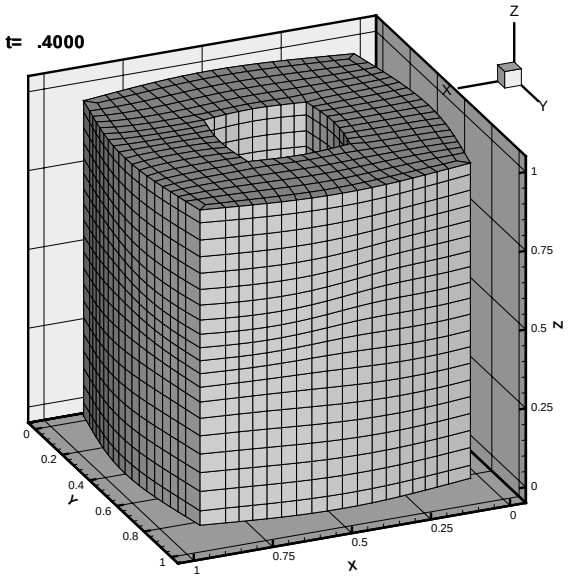

Figure 5.6. Grid at $t = 0.4$.

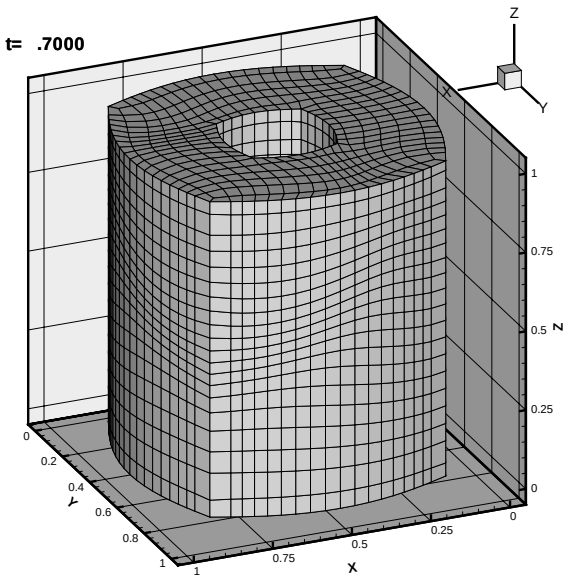

Figure 5.7. Grid at $t = 0.7$.

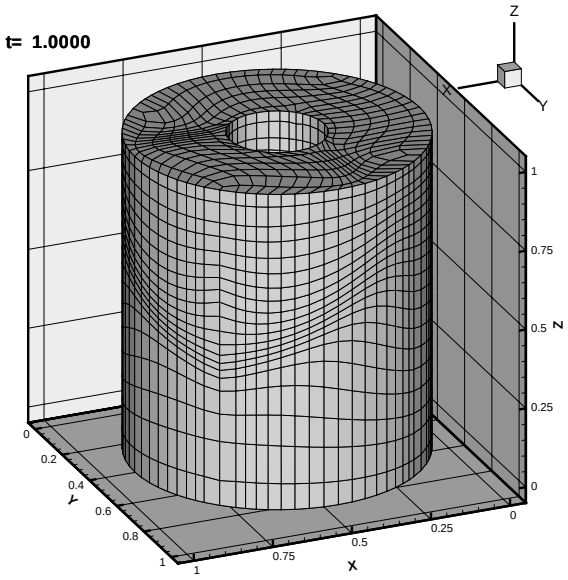

Figure 5.8. Final adapted grid on a thick-walled cylinder.

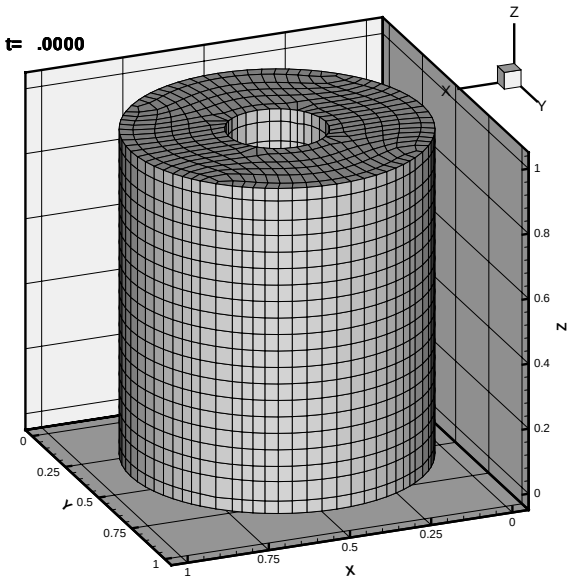

Figure 5.9. First grid from a template.

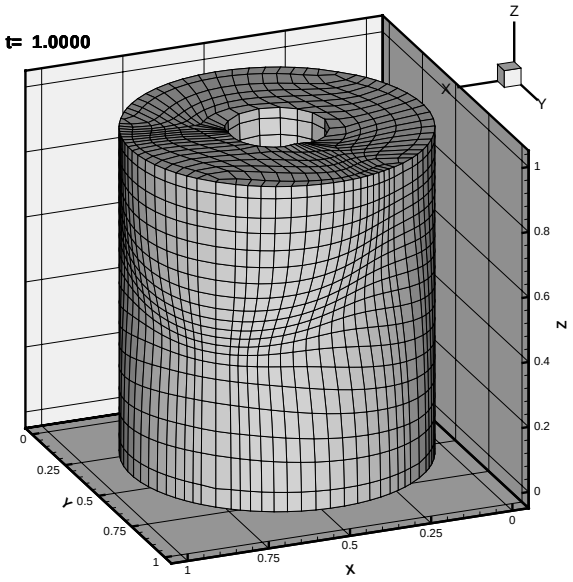

Figure 5.10. Adaptation towards an sphere.

CHAPTER 6

SOLID-STRESS ANALYSIS WITH A MESHFREE LSFEM

In this chapter, the LSFEM combined with the concept of meshfree is used to solve a plane stress problem. The LSFEM formulation of the problem was proposed by Zienkiewicz *et al.* [18]. In [10] the problem was modified to better accommodate the requirement (3.18). An extra variable was introduced and more equations were derived to obtain an elliptic system. However, in this work we apply the LSFEM to the original problem using the ideas of the particle finite element [6], the least-squares finite element collocation method [9].

The first-order PDE system for the plane stress is given by

$$\frac{\partial u}{\partial x} - \frac{1}{E}\sigma_x + \frac{\nu}{E}\sigma_y = 0 \tag{6.1a}$$

$$\frac{\partial v}{\partial y} + \frac{\nu}{E}\sigma_x - \frac{1}{E}\sigma_y = 0 \tag{6.1b}$$

$$\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} - \frac{2(1+\nu)}{E}\tau_{xy} = 0 \tag{6.1c}$$

$$\frac{\partial \sigma_x}{\partial x} + \frac{\partial \tau_{xy}}{\partial y} + f_x = 0 \tag{6.1d}$$

$$\frac{\partial \tau_{xy}}{\partial x} + \frac{\partial \sigma_y}{\partial y} + f_y = 0, \tag{6.1e}$$

which consists of three stress-strain equations and two equilibrium conditions, where σ_x, σ_y and τ_{xy} are stress components, u and v are the displacements in the x and y directions respectively, f_x and f_y are the body force components, E is the Young modulus and v is the Poisson ratio.

The LSFEM solution of (6.1) is a vector variable $\boldsymbol{u} = (u, v, \sigma_x, \sigma_y, \tau_{xy})$ that minimizes the functional

$$I(\boldsymbol{v}) = \int_{\Omega} (A\boldsymbol{v} - \boldsymbol{f})^2 \, d\omega \tag{6.2}$$

where

$$A\boldsymbol{u} = \sum_{i=1}^{n_d} A_i \frac{\partial \boldsymbol{u}}{\partial x_i} + A_0 \boldsymbol{u}, \tag{6.3}$$

i.e.,

$$A\boldsymbol{u} = A_1 \begin{pmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial v}{\partial x} \\ \frac{\partial \sigma_x}{\partial x} \\ \frac{\partial \sigma_y}{\partial x} \\ \frac{\partial \tau_{xy}}{\partial x} \end{pmatrix} + A_2 \begin{pmatrix} \frac{\partial u}{\partial y} \\ \frac{\partial v}{\partial y} \\ \frac{\partial \sigma_x}{\partial y} \\ \frac{\partial \sigma_y}{\partial y} \\ \frac{\partial \tau_{xy}}{\partial y} \end{pmatrix} + A_0 \begin{pmatrix} u \\ v \\ \sigma_x \\ \sigma_y \\ \tau_{xy} \end{pmatrix} \tag{6.4}$$

where

$$A_1 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \tag{6.5}$$

$$A_2 = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}, \tag{6.6}$$

$$A_0 = \begin{pmatrix} 0 & 0 & -\frac{1}{E} & \frac{\nu}{E} & 0 \\ 0 & 0 & \frac{\nu}{E} & -\frac{1}{E} & 0 \\ 0 & 0 & 0 & 0 & -\frac{2(1+\nu)}{E} \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \tag{6.7}$$

and

$$\boldsymbol{f} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ -f_x \\ -f_y \end{pmatrix}. \tag{6.8}$$

We find a LSFEM solution for (6.1) by solving an algebraic system $KU = F$ that is assembled from the element matrices

$$K_e = \int_{\Omega_e} (A\psi_1, A\psi_2, ..., A\psi_{N_n})^T (A\psi_1, A\psi_2, ..., A\psi_{N_n})\, d\omega \tag{6.9a}$$

$$F_e = \int_{\Omega_e} (A\psi_1, A\psi_2, ..., A\psi_{N_n})^T \boldsymbol{f}\, d\omega \tag{6.9b}$$

A plate with a small circular hole is subjected to a uniform stress $\sigma_x = 1$. Because of the symmetry of the model the first order PDE (6.1) is solved on the first quadrant only. The material properties used are $E = 1.0$ and $\nu = 0.3$. For an infinite plate the theoretical solution for σ_x on the line $x = 0$ is given by:

$$\sigma_x = 1 + \frac{1}{2}\left(\frac{1}{y^2} + \frac{3}{y^4}\right) \tag{6.10}$$

In the numerical examples, the grid consists of 750 nodes and $N_{\rm bc} = 213$. Using 697 non-overlapping bilinear quadrilateral elements (Figure 6.1) and one gaussian point, the requirement (3.18) is not satisfied:

$$\lambda = N_{\rm elem} \times N_{\rm G} \times N_{\rm eq} - (N_{\rm node} \times m - N_{\rm bc}) = \tag{6.11}$$

$$= 697 \times 1 \times 5 - (750 \times 5 - 213) = -52. \tag{6.12}$$

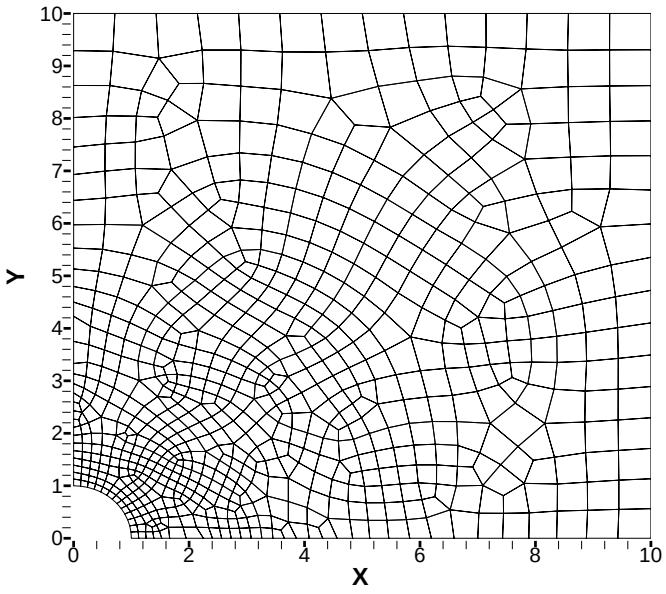

Figure 6.1. Mesh with 697 elements and 750 nodes.

Therefore the LSFEM solution cannot be found. Now, if two Gaussian points are used (therefore $N_{\rm G} = 2 \times 2$, since integration is on each coordinate), we have

$$\lambda = 697 \times 4 \times 5 - (750 \times 5 - 213) = 10403. \tag{6.13}$$

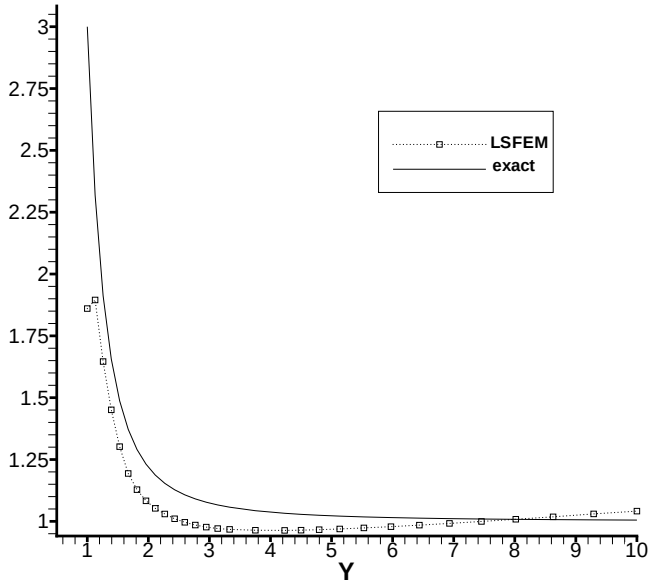

Figure 6.2. Theoretical σ_x and its 2-gaussian-point LSFEM solution at $x = 0$.

However this implies in solving an overdetermined system with too many residual equations and therefore the LSFEM solution is inaccurate (Figure 6.2).

We can use overlapping elements on the domain Ω and use the concept of the particle finite element [6] and the least-squares finite element collocation method [9]. The minimization in the LSFEM with the integrals approximated by gaussian quadrature is equivalent to minimizing the residuals at the gaussian points of each element. Thus, the same LSFEM scheme for non-overlapping elements can still be used.

Figure 6.3 shows an example of how an overlapping element is created. The overlapping elements will add more equations to the system of residual equations on the least-squares minimization. Although the LSFEM solution is found at the nodes, the residuals are minimized at the gaussian points, therefore, the solution is more accurate at the gaussian points.

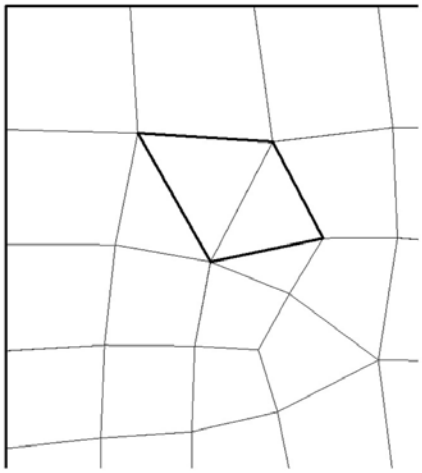

Figure 6.3. Overlapping element.

6.1 Example 1

By adding new elements to the grid in Figure 6.1, the new grid will consist of overlapping quadrilateral elements (Figure 6.4). Using one gaussian point the requirement in (3.18) becomes:

$$\lambda = 736 \times 1 \times 5 - (750 \times 5 - 213) = 143. \tag{6.14}$$

Figure 6.5 shows the contour levels for σ_x and Figure 6.6 shows the comparison with the theoretical σ_x.

6.2 Example 2

In example 1 we have $\sigma_x = 3.005$ at $(0, 1)$. However, away from that region, the solution seems inaccurate. Instead of overlapping elements clustered in a certain region of the domain, we now spread out the overlapping elements (Figure 6.7). The requirement (3.18), using one gaussian point, becomes

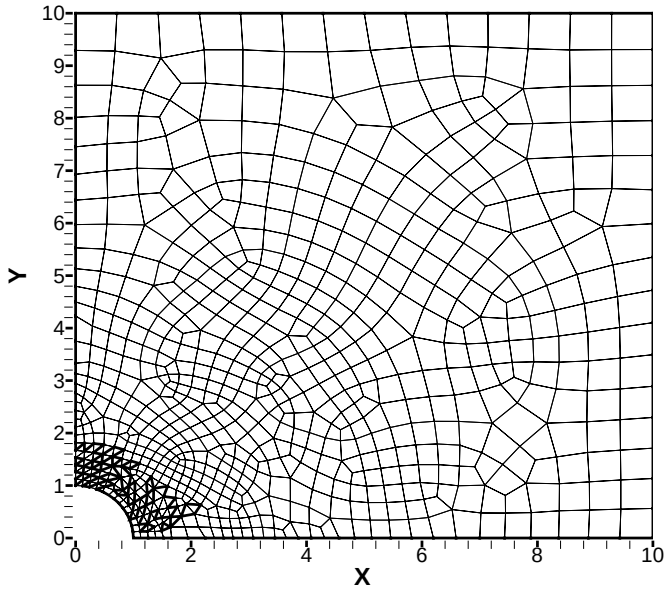

Figure 6.4. Overlapping elements.

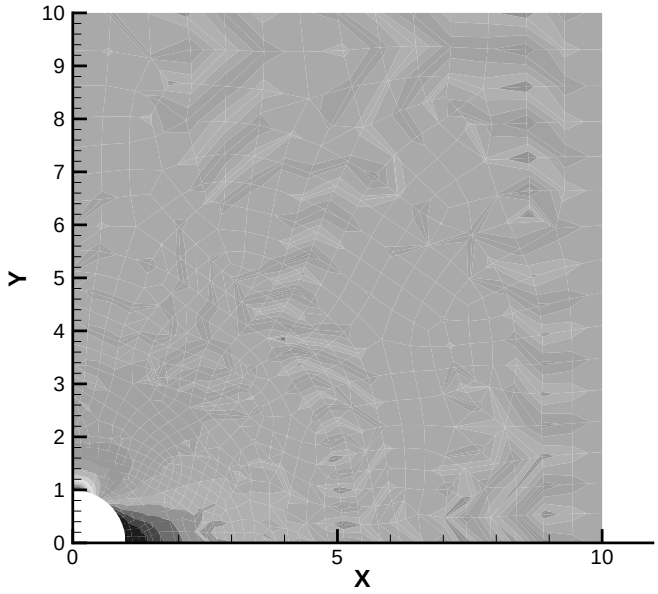

Figure 6.5. Contour levels for σ_x in Example 1.

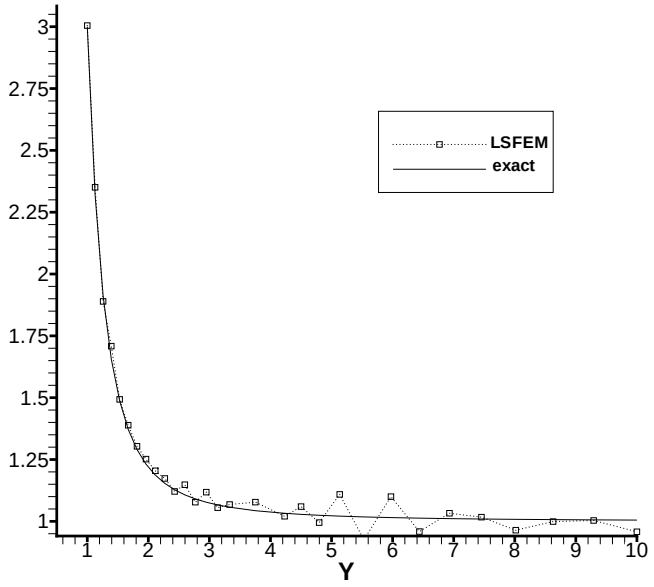

Figure 6.6. Theoretical σ_x at $x = 0$ and its LSFEM solution in Example 1.

$$\lambda = 736 \times 1 \times 5 - (750 \times 5 - 213) = 143. \tag{6.15}$$

Figure 6.8 shows the contour levels for σ_x. At $(0, 1)$ we have $\sigma_x = 3.08$. Figure (6.9) shows the solution for σ_x at $x = 0$, which is in good agreement with the theoretical result. Since the LSFEM minimization is enforced at the gaussian points, the solution is more accurate there, therefore a post-processing procedure may be required to smooth out the solution, interpolating the values at the nodes from the values at the gaussian points.

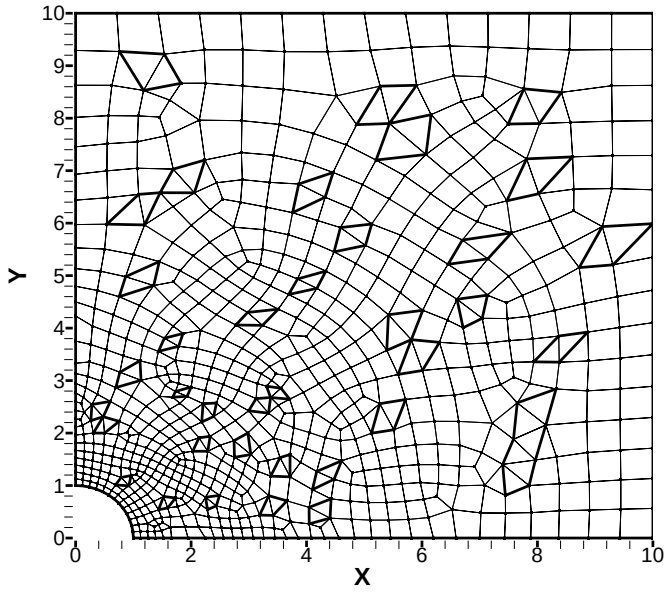

Figure 6.7. Spreading overlapping elements throughout the domain.

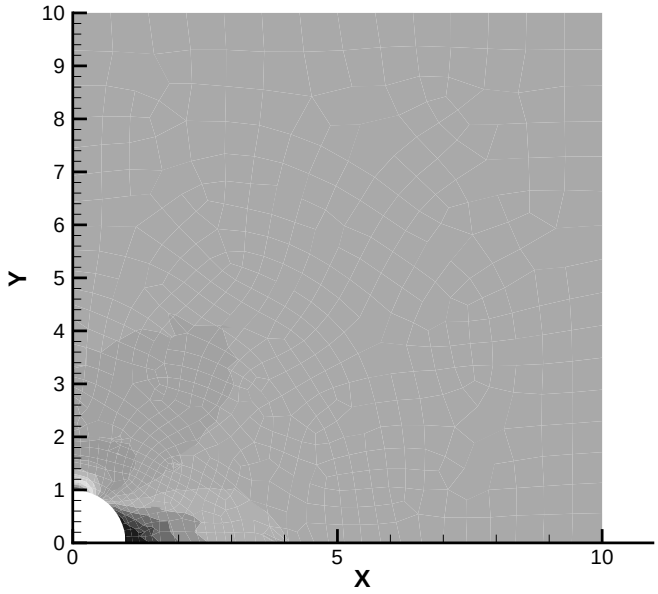

Figure 6.8. Contour levels for σ_x in example 2.

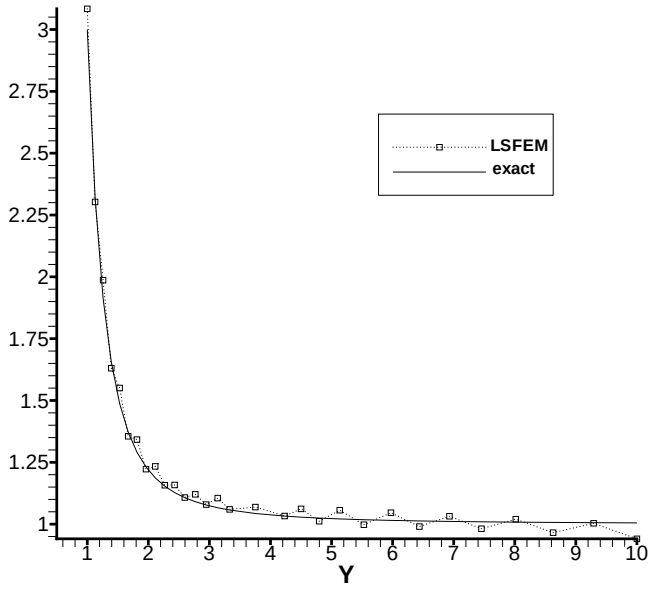

Figure 6.9. Theoretical σ_x at $x = 0$ and its LSFEM solution (in Example 2).

CHAPTER 7

CONCLUSIONS

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 non-folding, even in 3D.

Various boundary conditions can be directly enforced with LSFEM. Non-slippery 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 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.

Previous grid deformation methods were based on solving a potential ψ from a Poisson equation. The node velocity then is chosen to be $f\boldsymbol{\nabla}\psi$. This method can only be used in fixed domains due to the inability of imposing Dirichlet boundary condition on $\boldsymbol{\nabla}\psi$. In fact, the Neumann boundary condition for Poisson equation is used on a fixed domain, and consequently, $\boldsymbol{\nabla}\psi \cdot \boldsymbol{n} = 0$, where $\boldsymbol{n}$ 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

$\boldsymbol{\nabla}\psi$ on the boundary. This explains why the method based on Poisson equation works on fixed domains only. The method described in this work is based on solving directly a vector field $\boldsymbol{v}$ 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.

A LSFEM was applied to a plane stress problem using bilinear quadrilateral elements commonly found in the FEM literature. To find a LSFEM solution, it is important to have enough residual equations to obtain an overdetermined system. It was shown in the examples that this requirement may not be satisfied when using one gaussian point with the bilinear quadrilateral elements. Moreover, with two gaussian points, too many residual equations are obtained and therefore the LSFEM solution is inaccurate. Using the meshfree concept, overlapping elements were created to obtain sufficient residual equations to meet the requirement and not generating an extremely overdetermined system.

APPENDIX A

THE LEAST-SQUARES SOLUTION OF A LINEAR SYSTEM

Consider the linear algebraic system

$$\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nm} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix} \tag{A.1}$$

The least-squares solution of (A.1) is the solution $(x_1, x_2, \ldots, x_m)$ that minimizes the weighted squared residuals

$$I(\boldsymbol{x}) = \sum_{i=1}^{n} w_i \left(a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{im}x_m - b_i\right)^2, \tag{A.2}$$

where $w_i > 0$ are the weighting residuals. Setting the partial derivatives of I with respect to x_j to zero we have

$$\frac{\partial I}{\partial x_j}(\boldsymbol{x}) = 2\sum_{i=1}^{n} w_i \left(a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{im}x_m - b_i\right) a_{ij} = 0, \tag{A.3}$$

hence

$$\sum_{i=1}^{n} w_i \left(a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{im}x_m\right) a_{ij} = \sum_{i=1}^{n} w_i b_i a_{ij} \tag{A.4}$$

for $j = 1, \ldots, m$. Letting W be an $n \times n$ diagonal matrix with w_i in the diagonal, we obtain the $m \times m$ system (normal equations) :

$$A^T W A\boldsymbol{x} = A^T W \boldsymbol{b}. \tag{A.5}$$

The minimum of I is the solution of (A.5).

As mentioned in [9], the least-squares solution of (A.1) is guaranteed if the rank of the augmented matrix

$$A_{\text{aug}} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1m} & b_1 \\ a_{21} & a_{22} & \cdots & a_{2m} & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nm} & b_n \end{pmatrix} \tag{A.6}$$

is equal to $m + 1$, in which case the system (A.1) is overdetermined.

If rank $A_{\text{aug}} < m$ then

$$\text{rank } \left(A^T W A\right) \leq \text{ rank } A \leq \text{ rank } A_{\text{aug}} < m \tag{A.7}$$

therefore $\det \left(A^T W A\right) = 0$ and (A.1) cannot be solved by the least-squares method.

If rank $A_{\text{aug}} = m$ then (A.1) may or may not be solved by the least-squares method. For instance, the rank of the augmented matrix of

$$\begin{cases} x_1 + x_2 = 1 \\ x_1 + x_2 = 2 \\ x_1 + x_2 = 0 \end{cases} \tag{A.8}$$

is 2, and (A.5) becomes

$$A^T W A \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} w_1 + w_2 + w_3 & w_1 + w_2 + w_3 \\ w_1 + w_2 + w_3 & w_1 + w_2 + w_3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} w_1 + 2w_2 \\ w_1 + 2w_2 \end{pmatrix}. \tag{A.9}$$

Hence $A^T W A$ is singular. Now, the rank of the augmented matrix of

$$\begin{cases} x_1 + x_2 = 1 \\ 2x_1 + 2x_2 = 2 \\ x_1 - x_2 = 0 \end{cases} \tag{A.10}$$

is also 2 and (A.5) becomes

$$\begin{pmatrix} w_1 + 4w_2 + w_3 & w_1 + 4w_2 - w_3 \\ w_1 + 4w_2 - w_3 & w_1 + 4w_2 + w_3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} w_1 + 4w_2 \\ w_1 + 4w_2 \end{pmatrix}. \tag{A.11}$$

In this case $\det(A^T W A) \neq 0$ and the solution is $x_1 = \frac{1}{2}$ and $x_2 = \frac{1}{2}$.

REFERENCES

[1] P. Bochev, G. Liao, and G. dela Pena. Analysis and computation of adaptive moving grids by deformation. *Numer. Meth. PDEs*, 12, 1996.

[2] G. Carey. *Computational Grid Generation, Adaptation and Solution Strategies*. Taylor and Francis, Washington, DC, 1997.

[3] G. J. Fix and M. D. Gunzburger. On least squares approximations to indefinite problems of the mixed type. *Int. J. Numer. Meth. Engng.*, 12:453–469, 1978.

[4] D. Fleitas, J. Xue, J. Liu, and G. Liao. Least-squares finite element adaptive grid deformation in a non-linear time dependent problem. In *Advances in Applied Mathematics (2004 SIAM GATORS)*, Gainsville, Florida, 2004.

[5] T. P. Fries and H. G. Matthies. Classification and overview of meshfree methods. *Technical University Braunschweig, Germany*, Mar. 2003.

[6] S. Hao, H. S. Park, and W. K. Liu. Moving particle finite element method. *Int. J. Numer. Meth. Engng.*, 53:1937–1958, 2002.

[7] M. R. Hestenes and E. L. Stiefell. Methods of conjugate gradient for solving linear systems. *Nat. Bur. Std. J. Res.*, 49:409–436, 1952.

[8] S. R. Idelsohn, E. Onate, N. Calvo, and F. D. Pin. The meshless finite element method. *Int. J. Numer. Meth. Engng.*, 58:893–912, 2003.

[9] B.-N. Jiang. *The Least-Squares Finite Element Method: Theory and Applications in Computational Fluid Dynamics and Electromagnetics*. Springer, Berlin, 1998.

[10] B.-N. Jiang and J. Wu. The least-squares finite element method in elasticity - part i: Plane stress or strain with drilling degrees of freedom. *Int. J. Numer. Meth. Engng.*, 53:621–636, 2002.

[11] P. Knupp and S. Steinberg. *The Fundamentals of Grid Generation.* CRC Press, Boca Raton, FL, 1993.

[12] G. Liao and D. A. Anderson. A new approach to grid generation. *Appl. Anal.*, 44, 1992.

[13] G. Liao, Z. Lei, G. dela Pena, and D. Anderson. A moving finite difference method for partial differential equations. 2002.

[14] 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.

[15] J. Moser. Volume elements of a riemann manifold. *Trans AMS*, 120, 1965.

[16] B. Semper and G. Liao. A moving grid finite element method using grid deformation. *Numer. Meth. Part. Diff. Eq.*, 11:603, 1995.

[17] J. F. Thompson, Z. U. A. Warsi, , and C. W. Mastin. *Numerical Grid Generation.* North-Holland, Amsterdam, 1985.

[18] O. Zienkiewicz, D. Owen, and K. Lee. Least-squares finite element for elasto-static problems. use of 'reduced' integration. *Int. J. Numer. Meth. Engng.*, 8:341–358, 1974.

BIOGRAPHICAL STATEMENT

Dionisio Laeber Fleitas was born in Itaguai-RJ, Brazil, in 1973. He received his B.S. degree from Universidade Federal do Rio de Janeiro, Brazil, in 1996, and his Ph.D. degree from The University of Texas at Arlington in 2005, all in Mathematics. During his doctoral studies he taught many undergraduate courses in mathematics at The University of Texas at Arlington, presented research papers in national conferences and received awards such as Outstanding Graduate Student Teaching Award (2002), Outstanding Graduate Student Research Award (2003) and University Scholar Nomination (2003). His current research interest is in the area of Grid Generation and Numerical Analysis.