An Anisotropic Adaptive Method for the Solution of 3-D Inviscid and Viscous Compressible Flows

Anna Tam

A Thesis

in

The Department

of

Mechanical Engineering

Presented in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy at
Concordia University
Montreal, Quebec, Canada

April 1998

0-612-40309-2

ABSTRACT

An Anisotropic Adaptive Method for the Solution of 3-D Inviscid and Viscous Compressible Flows

Anna Tam, Ph.D.
Concordia University, 1998

The solution of complex three-dimensional computational fluid dynamics (CFD) problems in general necessitates the use of a large number of mesh points to approximate directional flow features such as shocks, boundary layers, vortices and wakes. Such large grid sizes have motivated researchers to investigate methods of introducing very high aspect ratio elements to capture these features. In this Thesis, an anisotropic adaptive grid method has been developed for the solution of three-dimensional inviscid and viscous flows by the finite element method. An edge-based error estimate drives a mesh movement strategy that allows directional stretching and re-orientation of the grid with more mesh points introduced along those directions with rapidly changing gradients. The error estimate is built from a modified positive-definite form of the Hessian tensor of a selected solution variable or combination of variables. The resulting metric tensor controls the magnitude as well as the direction of the grid stretching. The desired directionally adapted anisotropic mesh is constructed in physical space by a coordinate transformation based on this tensor. This research thus seeks a near-isotropic mesh in the transformed metric space and an equidistribution of the error over the mesh edges. The adaptive strategy can be considered to be the first 3-

D implementation of an improved spring analogy-based algorithm originally applied on quadrilateral meshes.

The adaptive methodology has been validated on various benchmark cases on both hexahedral and tetrahedral meshes. The numerical results obtained span inviscid and viscous flows, as well as internal and external aerodynamics. The effectiveness of the adaptive scheme to equidistribute the interpolation error over the edges of tetrahedral and hexahedral meshes has been gauged on analytical test cases where near-Gaussian distributions of the error were obtained. It was further demonstrated that the error estimate closely follows the true solution error.

In analyzing the solution error of different sized non-adapted and adapted grids, one could not only achieve the same level of solution error by adapting and solving on a much coarser grid, but a significant reduction in solution time as well. All test cases revealed that the flow solver required lower amounts of artificial dissipation for solution on the final adapted grids.

The current work should convincingly pave the way for its logical extension to unstructured grids, taking further advantage of refinement, coarsening and edge-swapping operations. It is strongly anticipated that this approach will shortly result in "optimal" grids.

Acknowledgments

I would first like to thank Dr. Wagdi Habashi for serving as my Thesis Supervisor. His openness to ideas (and my opinions!) and sense of humor have made my years of studies both enjoyable and interesting. I would also like to thank Dr. Michel Fortin of Université Laval for serving as Thesis co-supervisor.

The early stages of my Thesis were carried out at the Concordia University CFD Laboratory and the later stages at Pratt & Whitney Canada (P&WC). I would thus like to express my appreciation to all members of P&WC's CFD group, both in Montreal and Mississauga, for their many helpful suggestions and technical expertise. In particular, I would like to say a very special "thank you" to Dr. Michel Robichaud, Dr. Grant Guèvremont, Dr. Djaffar Ait Ali Yahia, Martin Peeters, Mohamad Sleiman and Mary Moore.

Thanks are also due to Dr. Laura Dutto of the Concordia University CFD Lab, Julien Dompierre of the CFD Lab and now at CERCA (Centre de Recherche en Calcul Appliqué) and his wife, Marie-Gabrielle Vallet, formerly of CERCA and now at IMI (Industrial Materials Institute) for their valued technical advice in the initial phases of this work.

Particular thanks are due to Pascal Tremblay of the Concordia University CFD Lab who is currently taking this work to the next level of application, namely unstructured grids.

I would like to take this opportunity to thank my family for their support during my studies and, especially my parents, Yuey Sun and Lan Fong Tam, for also serving as wonderful role models for me throughout life.

Last but certainly not the least, I would like to express my deep appreciation to my husband, Dr. Nicholas Krouglicof, for his support, patience and subtle words of encouragement ("Aren't you finished with the damn thesis yet?") during my lengthy studies.

This Thesis is dedicated to my son, Thomas, who was born during my doctoral studies, for bringing immense joy to my life.

Table of Contents

List of Figures

Nomenclature

Cp	coefficient of pressure
C_p, C_v	specific heat coefficients
Ec	Eckert number
e	approximation error
el	element
H_o	stagnation enthalpy
$H^m(\Omega)$	Sobolev space of functions with m square integrable generalized derivatives
k	thermal conductivity
L	reference length
$L_2(\Omega)$	Hilbert space of square integrable functions
M	Mach number
N	finite element shape function
p	pressure
Pr	Prandtl number
R	gas constant
$\Re$	real numbers
$\Re^n$	Euclidean n-dimensional space
Re	Reynolds number
Res	residual of a differential equation
T	static temperature
T_o	stagnation temperature
u, v, w	Cartesian velocity components
U	Solution vector
$\vec{V}$	velocity vector

V	finite dimensional space
W	Galerkin weight function
$\vec{x}$	position vector of global coordinates
x, y, z	Cartesian coordinates

Greek Symbols

$\vec{\xi}$	position vector of local coordinates
ξ, η, ζ	non-dimensional local coordinates of element
∇	gradient operator
Δ	change in a variable between iterations
$\in$	is a member of
γ	ratio of specific heats
Γ	boundary of Ω
λ	pressure dissipation coefficient
μ	viscosity
μ_{art}	artificial dissipation coefficient for momentum equations
μ_{cw}	crosswind artificial dissipation coefficient
μ_{sw}	streamwise artificial dissipation coefficient
Ω	domain in $\Re^n$
ρ	density
ϕ	unknown finite element variable
τ_{ij}	viscous stress tensor

Subscripts

∞	free stream conditions

h	mesh size parameter
i, j	row, column indices

Superscripts

e	elemental quantity
n	Newton iteration number

1. Introduction

The appeal of adaptive methods in computational fluid dynamics (CFD) lies in their potential for optimizing the computational process. Their objective is to provide optimal or near-optimal numerical approximations to complex boundary-value problems, satisfying user-specified requirements of accuracy and cost in terms of memory and CPU time.

The solution of CFD problems involves the use of a large number of mesh points to approximate directional flow features such as shocks, boundary layers and wakes. The grids necessary for analyses of such large-scale problems are generated either through proprietary or commercial codes and often only reflect the user's engineering sense of where to place and concentrate mesh points. Furthermore, these grids may also contain undesirable geometrical properties such as skewness that introduce grid-dependent errors in the flow solutions. This may result in the unwelcome situation of different flow solutions being obtained by different users of the same code, on the same flow problem, using the same overall number of grid points. Hence, emphasis should be placed on automatically optimizing the grid through an adaptive solution procedure.

An adaptive grid solution procedure would begin with the computation of an initial solution on the original grid. An assessment of the quality of this numerical solution is then obtained by deriving some form of estimate of the local error in the solution. This estimate would be subsequently used to modify the grid, whether it be a redistribution of the mesh points (r-method), a refinement or coarsening of the mesh size (h-method), an increase in the order of the interpolation polynomials (p-method), edge-swapping or any combination of the above adaptive strategies, in a systematic manner to improve the quality of the solution. The initial solution is then interpolated on the newly created adapted grid and the iteration resumes.

A complete adaptive solution procedure must therefore include modules for the solution of the governing flow equations, construction of the error estimates, and adaptation strategies all of which represents a significant amount of code development for three-dimensional applications. The grid adaptation method itself may be viewed as comprising two key components:

(1) error estimation

(2) adaptive strategy

1.1 Error Estimation

Grid adaptation requires selective local enrichment or refinement of meshes. It presupposes some method of post-processing of the solution to determine the local solution quality. The most common way is to estimate upper bounds on the discretization errors in the finite element solution. This represents only a measure of the error introduced by the approximate numerical solution and no claim is made on the error inherent in modeling a physical system by a mathematical model.

The quality of the numerical solution is assessed *a posteriori*, by the local approximation error in some appropriate norm, that is, after an initial calculation has been obtained. The error, $\|e\| = \|u - u_h\|$, is given by the difference between the exact solution u and its approximation u_h on a given mesh. A reliable error estimate drives the adaptive process by serving as the criterion for improving the quality of the solution, with its reliability, ε, characterized by an effectivity index η:

$$\eta = \frac{\varepsilon}{\|e\|} \tag{1.1}$$

which ideally should approach one as the mesh size, h, approaches zero or the order of the piecewise polynomial approximation, p, approaches ∞ [1]. Thus, an error estimate with an index close to unity becomes more accurate as the mesh is refined. Such sharp estimates tend to be difficult to construct and expensive to compute, with the danger that the error estimations and implementation time could become of the same order as the solution itself. It is for this reason that the tendency has been to use less computationally expensive error estimates which are easier to implement into existing finite element codes.

In practice, however, the construction of accurate error estimates for solutions of CFD problems still presents quite a challenge. An exact characterization of the error requires a knowledge of the solution itself, which is obviously not possible. While finite-element error estimates have been developed for simple elliptic problems, the difficulty is compounded for fluid dynamics problems by the fact that the governing equations represent a coupled system of non-linear partial differential equations (PDEs). Furthermore, most error estimates described in the literature are derived from one selected flow variable, whereas a true characterization of the error would necessarily require information from all flow variables.

The *a posteriori* error estimates used in finite element calculations may be classified into three main groups: interpolation, post-processing and residual methods.

1.1.1 Interpolation Error Estimates

Economical estimates of the local error over the individual elements are developed using existing bounds on the interpolation error, which are available from finite element interpolation theory. These estimates generally have poor effectivity indices and provide a crude estimate of the local error. This is related to the fact that the interpolation error is problem-independent and approximate solutions are needed to estimate higher order derivatives of the solution over each element. However, the advantage of interpolation estimates lies in their ease of construction and portability into existing codes [1].

1.1.2 Post-processing Error Estimates

Post-processing error estimates require the recovery of higher order finite element approximations from lower order solutions. As an example, differentiation of a C^0 continuous Galerkin approximation u_h will result in low order discontinuous derivatives, with inferior accuracy at inter-element nodes. For many years, nodal averaging and global or local L_2 projection have been widely used as derivative (stress) recovery techniques [2,3]. Zienkiewicz and Zhu [4] have employed a smoothing procedure, due to Oden and Brauchli, for extracting C^0 continuous derivatives by employing the same interpolation as that for u_h. The norm of the error in the derivative is estimated by calculating the norm of the difference between the approximate derivative and the recovered derivative. More recently, a recovery process based on a local least-squares projection for superconvergent points have been presented by the same authors [5]. Several applications of this error estimate can be found in the literature [6, 7].

1.1.3 Residual Error Estimates

As their name implies, these estimate techniques depend on the computation of local residuals. These methods estimate the error by solving local boundary-value problems satisfied by the error function over an element or a patch of elements of the finite element mesh. The formulation of local problems for linear elliptic problems has been studied by Bank and Weiser [8]. A comparison of several residual error estimates for the Stokes problem based on the formulation of local problems has been presented in [9].

Since these schemes depend on the governing equations of the problem, they are often more accurate than the previously described methods and are considered among the most robust of error estimate types. However, their cost and implementation complexity have inhibited their widespread use.

1.2 Adaptive Strategies

Once the error estimate of the solution has been calculated, it is coupled to an adaptive strategy to improve the accuracy of the initial numerical solution. Adaptive strategies are classified according to the actual mechanics used to modify the finite element approximation.

1.2.1 Mesh Movement Methods

These schemes, also called node redistribution or *r*-methods, adapt the grid by moving a fixed number of nodes (fig. 1.1). The grid points are relocated so that the grid density is large in regions of high error and small in regions of low error. Both the number of degrees of freedom and the order of the polynomial approximation inside the elements are kept constant. Its ease of implementation may be attributed to the method's fixed nodal connectivities, resulting in little change in an existing code's data structure. On the down side, however, the degree of desired grid adaptation based solely on such a strategy depends on the initial grid point distribution. An extensive review of various mesh movement strategies is provided in [10].

1.2.2 Mesh Enrichment Methods

Mesh enrichment methods may be further categorized into h-methods and p-methods. h-Methods control the local mesh size in the finite element grid by subdividing the elements when the local error estimate exceeds a user-specified tolerance (fig. 1.2). Such methods involve a renumbering of nodal points, elements, and element connectivities as the mesh is refined. The h-method may also simultaneously incorporate a capability to coarsen or de-refine a mesh, thereby producing larger mesh cells and reducing the number of unknowns. p-Methods adapt the approximation by increasing or decreasing the local order of polynomial basis in elements where the error in the element exceeds or falls below a user-specified error criterion (fig. 1.3). The order of polynomials can then range from low order ones all the way to spectral ones.

1.2.3 Combined Methods

A combination of the above adaptive strategies leads to some very effective techniques, the most common being h-r and h-p methods. Examples of h-r methods are described in the works of Kallinderis [11] and Palmerio [12].

Kallinderis introduces an adaptive algorithm for turbulent flows, which combines both grid embedding and nodal redistribution techniques, in an effort to efficiently resolve the small scales involved in viscous flows. In this case, the grid point redistribution is used to better align the grid with the flow features. Palmerio describes an adaptive method for shock capturing whereby several cycles of mesh refinement are initially applied. Since the enrichment steps can not take into account the shock direction, a mesh movement step was added to stretch the mesh along that direction.

An *h-r* method, in combination with an edge swapping strategy, has resulted in an edge-based anisotropic mesh optimization methodology applicable to 2-D inviscid and viscous unstructured meshes [13]. By applying a judicious sequence of these three adaptive strategies, a considerable savings in CPU time can be gained by continuously operating on the original grid, without any recourse to remeshing.

The *h-p* method combines a variation in local mesh size h with a local variation in the spectral order p of the approximation. These methods have been shown to have the best theoretical convergence rates [14] and thus, for a given computational cost they can deliver the most accurate solution. However, before such results can be obtained, several formidable problems in their effective implementation must be resolved, including new data structures, equation solvers, and criteria for choosing a distribution of mesh size and approximation orders. Pioneering work on adaptive *h-p* finite element methods has been carried out by Babuska [14] and Oden [15, 16].

1.2.4 Mesh Regeneration Method

Adaptive mesh regeneration or remeshing uses the existing grid and solution, tightly coupled to a proper error estimate, to generate a mesh that is more suited to the problem at hand. The new mesh will concentrate nodes in regions where relatively large discretization errors occur in order to produce a more suitable discretization. The current mesh becomes the background mesh for the new adapted grid. A single cycle of adaptive remeshing would consist of computing the error estimate for each grid point of the current grid, using this estimate to determine the element size, element stretching and stretching direction for the new grid, remeshing the computational domain by using the old grid as the background grid and, finally, interpolating the solution from the old to the new grid [17].

Stretched or anisotropic adaptive mesh generation methods may be viewed as a special case of adaptive remeshing. The need to solve the full Navier-Stokes equations and resolve the boundary layer and wake region characteristics of viscous flows has spurred on the development of these techniques [18, 19]. Directional flow features require elements with high aspect ratios where the mesh spacing is several orders of magnitude smaller in the direction normal to the boundaries than in the streamwise direction. Mavriplis [20] and Vallet [21] were among the first to generate highly stretched triangular meshes by using a locally mapped Delaunay triangulation method. The remeshing step is driven by local stretching vectors or metric tensors defined at each vertex of the grid. The triangulation is performed in a transformed space, which is obtained through a local mapping of the physical space, according to the values of the stretching vector. This creates the desired stretched Delaunay triangulation in physical space.

A less expensive alternative to global remeshing methods is to locally regenerate the grid. The current mesh is removed in those regions of the domain where the discrepancy between the current and desired mesh resolution is large. New local meshes are generated in these void regions using an advancing front-type mesh generation method where the initial front is defined by the boundary between meshed and void regions [22].

1.3 Literature Review

Over the past fifteen years the development of adaptive methodologies for CFD problems have become an active area of research, particularly for use on unstructured grids. An unstructured grid is defined as having no restriction placed on the number of elements which can meet at any vertex of the mesh. The most common unstructured grids are those consisting of triangular (in 2-D) and tetrahedral (in 3-D) elements. Much more rarely, however, hexahedral, prismatic and other grids are also used [23-26]. This is opposed to structured grids where the pattern and number of elements meeting at the vertices are specified. Structured grids usually refer to those comprising quadrilateral (in 2-D) or hexahedral (in 3-D) elements.

One reason for the increased popularity of unstructured grids in recent years is the relative ease with which flows over arbitrary complex geometries may be discretized. Even efficient conventional structured grid generation for a complicated geometry can be very laborious and time-consuming since these grids must meet certain orthogonality constraints. On the other hand, the generation of an unstructured tetrahedral mesh to fit a complex geometry often proceeds with significantly lower turnaround times due to the geometric flexibility of tetrahedra in "hugging" such a geometry.

Aside from the easier treatment of geometric complexities, another advantage of unstructured meshes is the straightforward manner in which adaptive strategies may be implemented. Due to its lack of inherent structure, mesh refinement, coarsening and edge swapping, which are all adaptive operations that can alter the nodal connectivities, may be implemented arbitrarily in any part of the grid. This is not possible with structured meshes where such operations would lead to a destruction of existing grid structure and create a need for the manipulation of constrained or hanging nodes [27, 28].

Despite these advantages of unstructured grids, much successful work in mesh adaptation continues to be pursued on structured grids [15, 29-33]. One also observes that grid generation packages and flow solvers for structured grids remain the norm in industry. Reasons for this may be attributed to the decreased CPU time for numerical integration of PDEs on structured grids over unstructured counterparts and the need for orthogonality of the wall layer elements for turbulence modeling.

Implementation of adaptive strategies in earlier work on inviscid flow simulations sought to improve solution accuracy by "classical" refinement, in which the elements were subdivided isotropically. Two-dimensional unstructured meshes are described as isotropic if the two length scales of all the triangular elements are essentially the same. A triangle marked for refinement would be subdivided into two or four elements, depending on the number of desired mid-side nodes to be added [34, 35]. The refined element would have a local length h_e as determined by the error estimate and its shape would approach that of an equilateral triangle. Similarly, in the three-dimensional case, Löhner presented a mesh enrichment strategy in which a tetrahedron is refined isotropically into two, four or eight elements [36].

However, these classical enrichment techniques do not consider the directionality of the flow features to be captured. In typical aerodynamic computations, regions with rapid changes in flow variables are embedded in regions where the flow variables vary more smoothly. Examples of directional flow phenomena possessing large gradients include shocks, boundary layers, vortices and contact discontinuities. Since the classical isotropic adaptive methods are optimal only for those situations in which large gradients of the flow variables occur in all spatial directions, directional flow characteristics are not necessarily resolved in a cost-effective manner and the number of elements required rapidly increases with each isotropic refinement.

Grid generation techniques, based upon the advancing front or Delaunay triangulation, are most suited to creating elements with aspect ratios of the order of unity. Although this type of element is optimal for inviscid flow simulation, the approach is deemed not to be valid for viscous flows, particularly in three-dimensional applications. In order to resolve boundary layers with unit aspect ratio elements, the number of grid points for a complex configuration would be impractically high. These potentially large grid sizes have motivated researchers to investigate techniques of introducing or generating very high aspect ratio elements for viscous flow problems.

Simpson's error studies on optimal triangular mesh generation [37-39] and on whose pioneering work this thesis is motivated, have led him to observe that:

> *"A connection between mesh anisotropy and error behavior of piecewise polynomial approximations based on the mesh is certainly to be expected. Consider, for example, a mesh to be used for piecewise linear approximation of a bivariate function which changes rapidly, i.e. has a large second derivative, in one direction but changes slowly, (a small second derivative) in the perpendicular direction. It is intuitively apparent that, for a fixed number of vertices, the approximation will be better if the vertices are spaced more closely in the first direction than in the second, i.e. if the mesh is anisotropic."*

Peraire et al. [40] were among the first researchers to seek solutions on stretched or anisotropic meshes. They described an adaptive remeshing procedure that incorporated directional stretching and refinement for the steady solution of the 2-D Euler equations. The error estimate, which was a function of stretching and directional parameters obtained from a computed flow solution on a previous grid, was coupled to a triangular mesh generator. This work was later extended by the same group to solve the 3-D inviscid flow equations, again by mesh regeneration [41]. Palmerio presented a 2-D adaptive node movement algorithm for the steady Euler calculations. The high degree of stretching obtained by the node movement scheme was combined with local isotropic refinement to achieve shock capturing [12].

While the directional adaptive strategies described in [40, 41] have proven valuable for inviscid flow simulations, the degree of grid stretching achieved is still several orders of magnitude smaller than that needed to resolve viscous flows. To produce the very high aspect ratio elements required for such flows, two different approaches have been adopted. One solution would be to define the types of triangular or tetrahedral elements which are desirable for anisotropic mesh generation and to modify existing methods or devise new techniques for generating meshes which contain these elements. The second approach would be to use hybrid grids whereby alternative element types are introduced only in those parts of the domain requiring a high degree of stretching.

In applying the first approach, some researchers have chosen to compute the Navier-Stokes equations on a fully unstructured mesh in which the viscous region is characterized by strongly anisotropic triangular elements [20, 21, 42-44]. Mavriplis has shown that a Delaunay triangulation performed in a locally stretched space yields very high aspect ratio triangles suitable for computing high Reynolds number flows [20]. In a similar vein, Vallet has described in her thesis how the desired anisotropy in the mesh can be specified by a symmetric error metric tensor [21]. Kornhuber and Roitzsch have devised an anisotropic refinement strategy to efficiently resolve boundary layers based on directed refinement of pairs of triangles [45]. More recently, other authors have attempted to generate unstructured viscous meshes by adopting an *a priori* adaptation strategy in which the stretched elements generated assume a particular dominant flow direction [46, 47].

Since the generation of tetrahedral elements for boundary layers is quite difficult and structured grids are superior in capturing the directionality of the flow field, some authors have pursued a hybrid structured-unstructured grid approach. This involves fitting a thin local mesh of structured (quadrilateral) or semi-structured (prismatic) high aspect ratio elements in the viscous regions and generating an unstructured (triangular/tetrahedral) mesh to fill the remainder of the domain [25, 48-50]. Prismatic grids serve as a compromise between unstructured and structured grids. They possess the geometric flexibility of unstructured meshes, as well as the orthogonality and high aspect ratio features of structured meshes.

Anisotropic meshes for three-dimensional inviscid and viscous flows can also be created by using an improved edge-based mesh movement or nodal redistribution scheme which will be described in Chapters Three and Four. It was Gnoffo who first introduced an adaptive method for grid point redistribution, based on equidistribution of local flow gradients [51,52]. The method is analogous to determining the equilibrium position of a system of springs that connect each node with neighboring nodes and whose spring constant is proportional to the local error. Nakahashi and Deiwert extended this method to two and three dimensions using the concept of tension and torsion springs to control the grid spacing and skewness, respectively [32]. Other applications of mesh movement schemes include work by Palmerio, Davies, Löhner, Catherall, Diaz and Oden [12, 33, 53-56].

1.4 Objectives and Overview of Thesis

In this thesis an anisotropic adaptive grid method has been developed for the solution of three-dimensional inviscid and viscous flows by the finite element method. An edge-based error estimate drives an improved mesh movement strategy to allow directional stretching and re-orientation of the grid such that, effectively, more mesh points are introduced along those directions with rapidly changing gradients of the selected solution variable. This method is applicable on both structured and unstructured grids.

The error estimate is evaluated using a bound available from finite element interpolation theory [55, 57] and is constructed from a matrix of second derivatives of the selected solution variable [21, 39]. This matrix or symmetric tensor controls the magnitude as well as the direction of the grid stretching. The tensor is a function of the space coordinates and is used to define a measure of error, namely a Riemannian metric. The desired directionally adapted anisotropic mesh in physical space is constructed by a coordinate transformation based on this error metric tensor. This results in an isotropic mesh in the transformed metric space but an anisotropic mesh in physical space. The resulting error estimate is thus considered edge-based since it is precisely the length of the edges of the elements with respect to this metric space. The length of an edge in this metric may be interpreted as the 1-D interpolation error along this edge. The approach taken in this research seeks a near-isotropic mesh in the transformed metric space and aims to equidistribute the error over the edges of the elements of the computational domain.

This adaptive methodology has been embedded in *NS3D*, an efficient, three-dimensional finite element code for the analysis of inviscid and viscous compressible flows, which has been jointly developed by Pratt & Whitney Canada with Concordia University.

The second chapter of the thesis describes the numerical discretization of the governing flow equations: the Euler and Navier-Stokes equations. The finite element equations, which are based on a weak Galerkin formulation, are derived.

The third chapter begins with a discussion of anisotropic grid transformations which derive from the work of Simpson and D'Azevedo [36-38]. This is then followed by a detailed description of the construction of the edge-based interpolation error estimate implemented in this thesis.

The fourth chapter on adaptive strategies focuses primarily on the mesh movement scheme. The treatment of boundary nodes, control of mesh quality, interpolation of the error metric and solution on the adapted grid and optimization of the mesh movement algorithm are all described. The chapter ends with a brief treatment of other adaptive strategies needed for an unstructured grid version of this work, such as mesh refinement/coarsening and edge-swapping.

The fifth chapter on solution procedure provides details of the flow solver algorithm as well as the coupling of the solver with the grid adaptation method.

The sixth chapter presents validation test cases for inviscid and viscous flows on tetrahedral, as well as hexahedral, meshes. For all cases, the non-adapted and adapted solutions are analyzed and compared to experimental and/or other numerical results.

The seventh and final chapter states the major conclusions and proposes the direction of near-future research work in mesh adaptation.

FIGURES

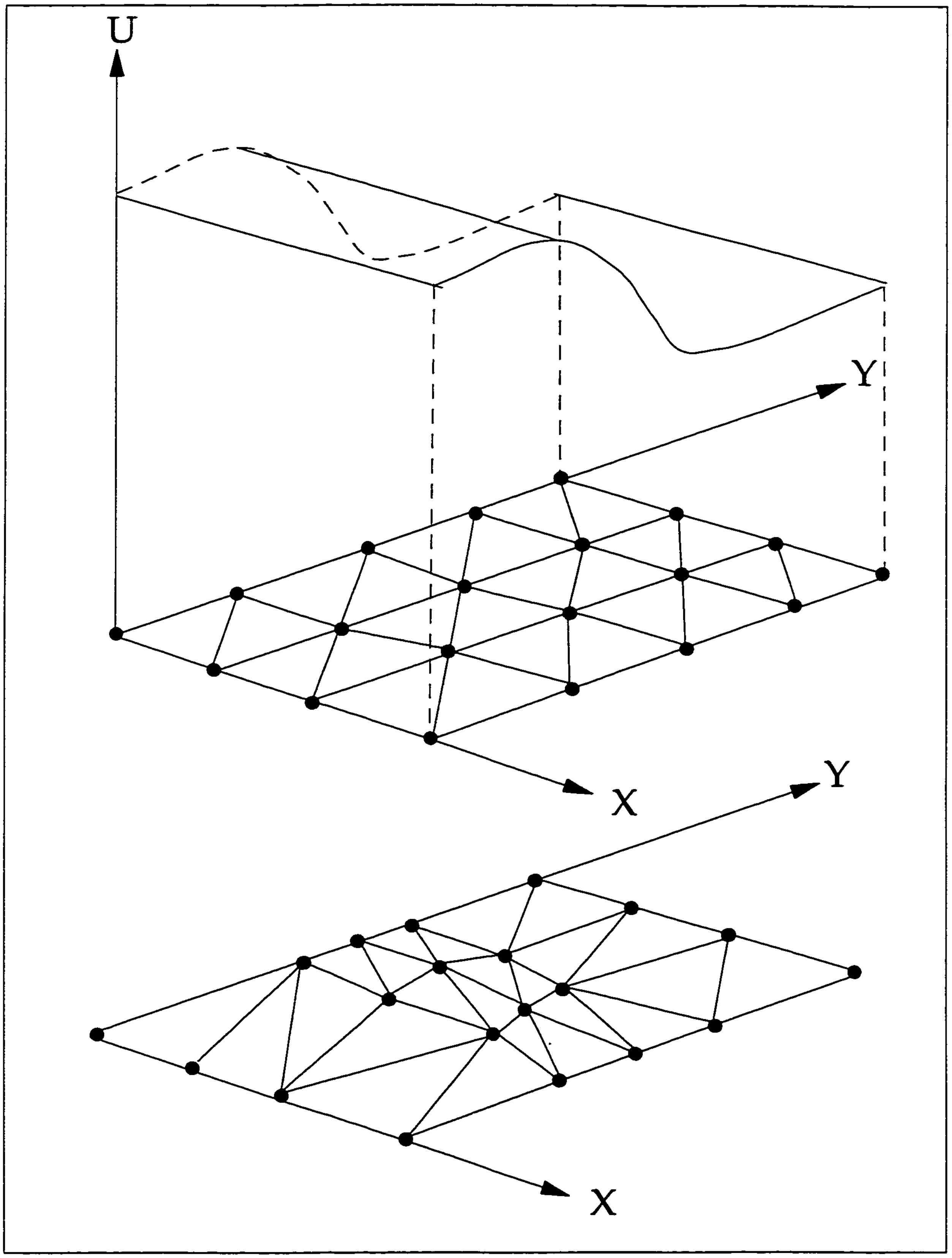


Figure 1.1 r-Method

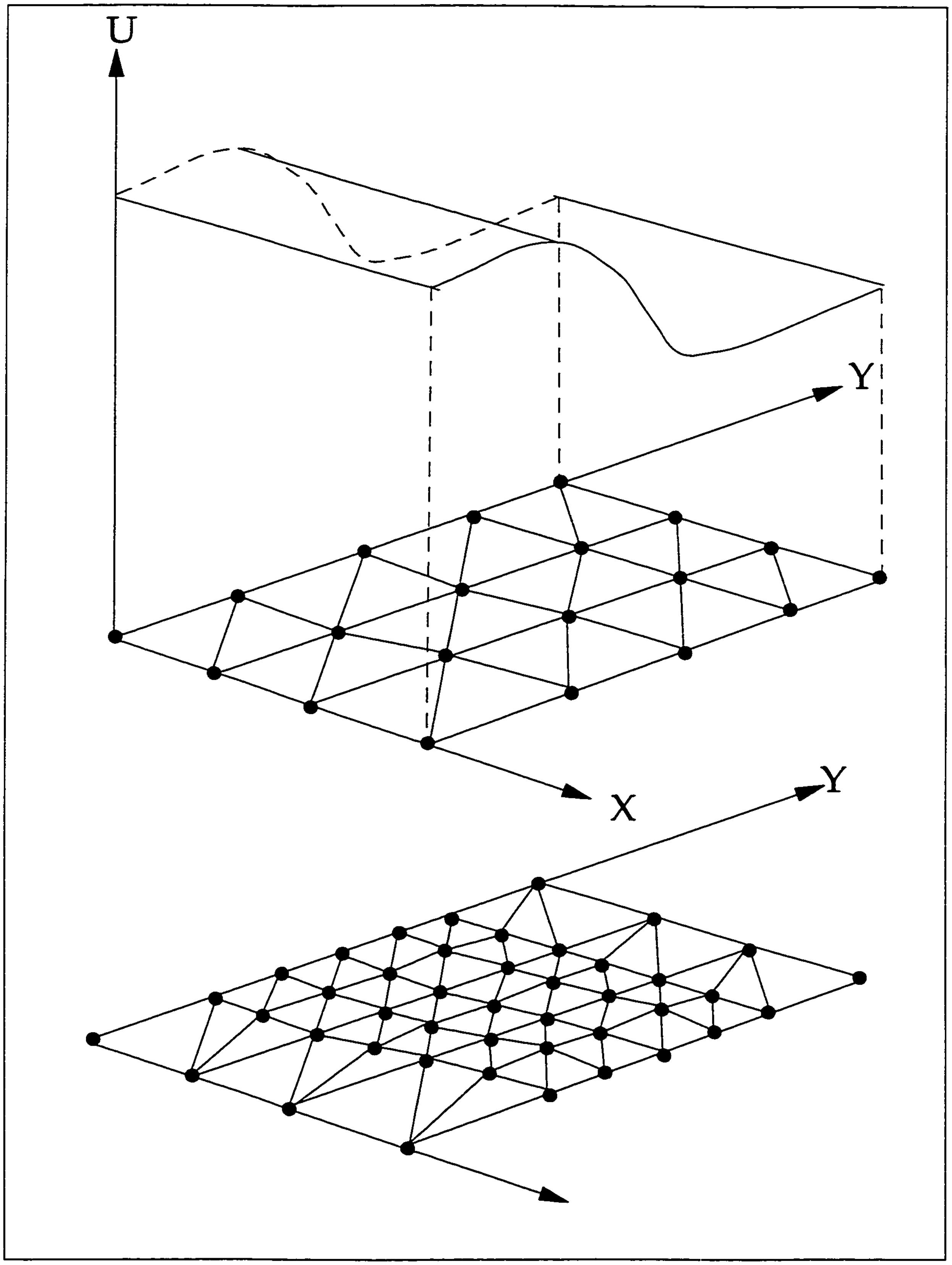


Figure 1.2 h-Method

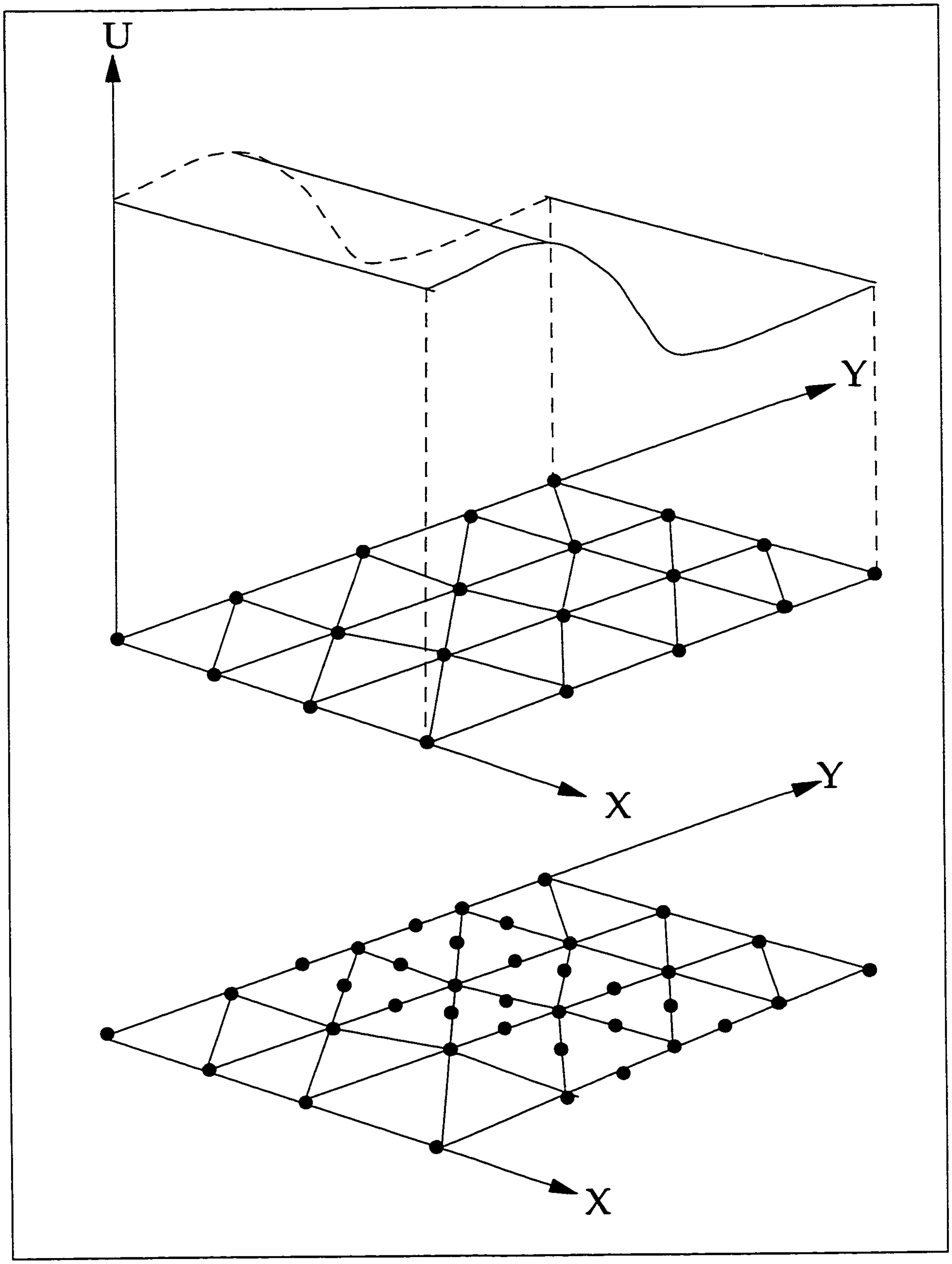


Figure 1.3 p-Method

2. Governing Flow Equations and Finite Element Discretization

2.1 Introduction

This chapter describes the governing equations for inviscid and viscous flows as well as their numerical discretization by the finite element method. The flow solver used in the present work is *NS3D*, which is a 3-D laminar/turbulent, steady/unsteady, compressible Navier-Stokes/Euler code, developed jointly by Concordia University (CFD Laboratory) and Pratt & Whitney Canada [58]. In this thesis, steady inviscid and viscous compressible flow of Newtonian fluids will be considered in the context of 3-D grid adaptation. *NS3D* solves for the mass flux components variables $\rho u, \rho v, \rho w$, pressure p and temperature T.

2.2 Governing Equations

The fundamental equations of fluid dynamics are developed from the application of the universal laws of conservation of mass, momentum and energy to a fluid flow. These governing equations are usually written in non-dimensional form. The advantage in doing this is that the non-dimensional flow variables are usually of the order of magnitude of unity - a convenient means in computational work to minimize numerical roundoff errors resulting from the different flow variable scales. In *NS3D* the following non-dimensionalizing procedure is performed,

$$x^* = \frac{x}{L} \quad u^* = \frac{u}{V_\infty} \quad \rho^* = \frac{\rho}{\rho_\infty} \quad p^* = \frac{p}{\rho V_\infty^{\,2}} \quad T^* = \frac{T}{T_\infty} \quad C_p^{\,*} = \frac{C_p}{C_{p_\infty}} \qquad (2.1)$$

where the non-dimensional variables are denoted by an asterisk, free stream conditions are represented by ∞ and L is the reference length used in the Reynolds number.

2.2.1 Navier-Stokes Equations

The full system of Reynolds-Averaged Navier-Stokes equations provides the most commonly used description of viscous fluid flow. This system of equations includes:

Continuity Equation

$$\frac{\partial \rho}{\partial t} + \nabla \bullet \left(\rho \vec{V}\right) = 0 \tag{2.2}$$

Momentum Equations

The body forces are neglected in the following vector form of the viscous momentum equations:

$$\begin{aligned} \rho \frac{D\vec{V}}{Dt} = -\nabla p + \frac{1}{Re}\left[\frac{4}{3}\nabla\left(\mu \nabla \bullet \vec{V}\right) + \nabla\left(\vec{V} \bullet \mu \nabla\right) - \vec{V}\nabla^2 \mu\right] \\ + \frac{1}{Re}\left[\nabla \mu \times \left(\nabla \times \vec{V}\right) - \left(\nabla \bullet \vec{V}\right)\nabla \mu - \nabla \times \left(\nabla \times \mu \vec{V}\right)\right] \end{aligned} \tag{2.3}$$

Energy Equation

The energy equation, under the assumption of a variable property perfect gas and in the absence of heat sources and radiation heat transfer, can be written as

$$\rho C_p \frac{DT_0}{Dt} - \frac{1}{PrRe}\nabla \bullet \left[\kappa \nabla \left(T_0 - \frac{\vec{V}^2}{2C_p}\right)\right] = Ec\frac{\partial p}{\partial t} + \frac{Ec}{Re}\nabla \bullet \left(\vec{V}\tau_{ij}\right) \tag{2.4}$$

where τ_{ij} denotes the viscous stress tensor.

It has become common practice to include the continuity equation (2.2) and the energy equation (2.4) in the set of equations called the Navier-Stokes equations although, strictly speaking, this term refers only to the components of the viscous momentum equation (2.3). To close the system of equations, additional relations are required, namely, the equation of state

$$p = \rho R T \tag{2.5}$$

and empirical equations expressing viscosity μ and thermal conductivity k as functions of temperature [59].

2.2.2 Euler Equations

The Euler equations are a reduced set of equations strictly valid only in the nearly inviscid portion of a high-speed flow field. They are derived from the Navier-Stokes equations by neglecting all shear stresses terms. The system of Euler equations includes the continuity equation (2.2) as well as the following:

Inviscid Momentum Equations

$$\rho \frac{D\vec{V}}{Dt} = -\nabla p \tag{2.6}$$

Inviscid Energy Equation

$$\rho C_p \frac{DT_0}{Dt} = Ec \frac{\partial p}{\partial t} \tag{2.7}$$

Under the assumptions of steady flow, negligible heat transfer and constant total enthalpy along the inflow boundaries, the energy equation can further simplify to the constant total enthalpy condition:

$$H_0 = \frac{\gamma}{\gamma - 1}\frac{p}{\rho} + \frac{1}{2}\vec{V} \bullet \vec{V} \tag{2.8}$$

2.3 Galerkin Finite Element Method

2.3.1 Overview of the Finite Element Approximation

In the finite element method, a geometrically complex domain of a given flow problem is discretized as a collection of simple nonoverlapping subdomains called elements [60-62]. Within each element a certain number of points or nodes are defined, which can be located on the edges, faces or inside the element. The numerical value of the solution unknowns is to be determined at these nodes. These solution variables are approximated by polynomials. If ϕ_h is an approximate solution of $\phi(\bar{\mathbf{x}})$, a series expansion of the form

$$\phi_h(\bar{\mathbf{x}}) = \sum_J \hat{\phi}_J N_J(\bar{\mathbf{x}}) \tag{2.9}$$

may be written where the summation extends over all nodes J and N_J are the shape or interpolation functions. The functions N_J are chosen to be locally defined polynomial interpolations within each element, taking on a zero value outside the particular element. The coefficients $\hat{\phi}_J$ in equation (2.9) are the unknown nodal values of the solution variable. These local interpolation functions possess the following properties on each element e, where node J belongs to e:

(i) at $\phi_h(x_J, y_J, z_J) = \hat{\phi}_J$, the function $N_J^{(e)}$ satisfies

$$N_I(x_J, y_J, z_J) = \begin{cases} 0 & if \quad I \neq J \\ 1 & if \quad I = J \end{cases} \tag{2.10}$$

(ii) at any point (x, y, z) within an element,

$$\sum_J N_J^{(e)}(x, y, z) = 1.0 \tag{2.11}$$

The global function N_J is obtained by assembling the contributions $N_J^{(e)}$ of all the elements to which node J belongs.

The method of weighted residuals offers a means by which to formulate the finite element equations. In this work the discretized form of the governing equations is obtained by minimizing, in a weighted average sense, the residuals of the system of equations (Navier-Stokes equations (2.2)-(2.4) or Euler equations (2.2) and (2.6)) over the solution domain. This is carried out by multiplying each equation by a weight function, which in the Galerkin finite element method is identical to the shape function, and integrating over the domain. The weak form is then obtained by integrating by parts the weighted residual form of the system of equations. Details of the weak Galerkin formulation for the full Navier-Stokes system of equations are provided in Appendices A and C.

2.3.2 Finite Element Discretization

NS3D accommodates isoparametric linear tetrahedral and trilinear hexahedral elements. Isoparametric elements use the same shape functions to define both geometry and solution variables. To simplify analytical expressions for elements of complex shapes, a reference element is defined in a local non-dimensional space with a simple geometrical shape, as shown for linear tetrahedral and trilinear hexahedral elements (fig. 2.1-2.2). The transformation from $\bar{\mathbf{x}}^e$-space to $\bar{\xi}^e$-space makes use of the shape functions in local coordinates through the geometrical discretization:

$$x=\sum_{J}^{ndperl}N_J(\xi,\eta,\zeta)\hat{x}_J \quad y=\sum_{J}^{ndperl}N_J(\xi,\eta,\zeta)\hat{y}_J \quad z=\sum_{J}^{ndperl}N_J(\xi,\eta,\zeta)\hat{z}_J \tag{2.12}$$

where the variable, *ndperl,* refers to the number of nodes per element. Similarly, the solution vector $\mathbf{U}=(\rho u,\rho v,\rho w,p)$ is approximated as

$$\mathbf{U}=\sum_{J}^{ndperl}N_J(\xi,\eta,\zeta)\hat{\mathbf{U}}_J \tag{2.13}$$

In *NS3D* all the solution variables are interpolated with shape functions of equal order, be they linear tetrahedral shape functions,

$$N_1=1-\xi-\eta-\zeta \qquad N_2=\xi \qquad N_3=\eta \qquad N_4=\zeta \tag{2.14}$$

or trilinear hexahedral shape functions,

$$\begin{aligned}
N_1 &= \frac{1}{8}(1-\xi)(1-\eta)(1-\zeta) & N_5 &= \frac{1}{8}(1-\xi)(1-\eta)(1+\zeta) \\
N_2 &= \frac{1}{8}(1+\xi)(1-\eta)(1-\zeta) & N_6 &= \frac{1}{8}(1+\xi)(1-\eta)(1+\zeta) \\
N_3 &= \frac{1}{8}(1+\xi)(1+\eta)(1-\zeta) & N_7 &= \frac{1}{8}(1+\xi)(1+\eta)(1+\zeta) \\
N_4 &= \frac{1}{8}(1-\xi)(1+\eta)(1-\zeta) & N_8 &= \frac{1}{8}(1-\xi)(1+\eta)(1+\zeta)
\end{aligned} \tag{2.15}$$

Such a transformation to local coordinates facilitates the numerical integration of the stiffness matrix and residual vector equations.

In the Galerkin weighted residual method, the weight functions are chosen to be identical to their corresponding shape functions, that is,

$$W_I(\xi,\eta,\zeta) = N_I(\xi,\eta,\zeta) \tag{2.16}$$

2.4 Artificial Dissipation

In *NS3D* the discretized form of the governing equations is obtained by applying the Galerkin finite element method which is equivalent to approximating the derivative terms by a central finite difference scheme. In such a centered scheme, the first order derivatives are decoupled leading to an odd-even point decoupling or checkerboarding effect. The Ladyzenskhaya, Babuska and Brezzi (LBB) stability condition established that checkerboarding would occur unless the interpolation functions for velocities are one order higher than those for pressure [62]. *NS3D* uses equal order interpolation for both velocities and pressure variables and to suppress this decoupling, artificial dissipation terms are added to the governing equations. These terms are needed to eliminate unphysical numerical oscillations and should be kept as small as possible so as not to degrade the approximate solution.

A pressure dissipation term, $\lambda\nabla^2 p$, is added to the right-hand side of the continuity equation, yielding

$$\frac{\partial\rho}{\partial t}+\nabla\bullet\left(\rho\vec{V}\right)=\lambda\nabla^2 p \tag{2.17}$$

This dissipation term produces an error in mass conservation proportional to its magnitude. To refine it to second order accuracy, the dissipation term is reformulated as follows,

$$\frac{\partial\rho}{\partial t}+\nabla\bullet\left(\rho\vec{V}\right)=\lambda\nabla\bullet\left(\nabla p-\overline{\nabla p}\right) \tag{2.18}$$

where the balancing term, $\overline{\nabla p}$, represents the nodal values of the averaged gradients of pressure [58, 63]. The user-specified coefficient λ must be sufficiently small to minimize the error in mass conservation but it must also be large enough to prevent spurious pressure oscillations.

Artificial dissipation terms, denoted as μ_{art} (see Chapter Five), are also added to the right-hand side of the momentum equations and consist of both "crosswind" and "streamwise" components. For example, in the x-momentum equation, the first order crosswind term takes the form

$$\mu_{cw}\left[\left(\Delta x u_x\right)_x+\left(\Delta y u_y\right)_y+\left(\Delta z u_z\right)_z\right] \tag{2.19}$$

where Δx, Δy, and Δz may be viewed as weighting factors for the Laplacian of the x-component of velocity and μ_{cw} is the user-specified crosswind artificial dissipation coefficient. The streamwise artificial dissipation component for the x-momentum equation is formulated as

$$\frac{\partial}{\partial s}\left[\mu_{sw}\left(\frac{|u|\Delta x+|v|\Delta y+|w|\Delta z}{|\vec{\mathbf{V}}|}\right)\left(uu_x+vu_y+wu_z+P_x\right)\right] \tag{2.20}$$

where s refers to the streamwise direction and μ_{sw} represents the user-specified streamwise artificial dissipation coefficient. As with the pressure dissipation coefficient λ, the values of μ_{cw} and μ_{sw} must be minimized so as not to degrade the solution quality but must also be large enough to prevent unphysical oscillations. Typical values of μ_{cw} and μ_{sw} are such that the crosswind artificial dissipation coefficient is one order of magnitude smaller than the streamwise one. In choosing such values, one minimizes the amount of artificial dissipation introduced into the numerical scheme by adding dissipation predominantly in the streamwise direction. It is preferable to keep the crosswind term as small as possible to avoid artificially dissipating the solution in all directions. Based on experience, the addition of only streamwise artificial dissipation is not sufficient and a minimum amount of crosswind dissipation is required for non-oscillatory solutions.

2.5 Time Discretization

A finite difference time integration scheme based on Gear's method has been implemented in *NS3D* for the time-accurate solution of the unsteady compressible Navier-Stokes and Euler equations [64]. Although unsteady flows are not considered here, the time integration scheme is necessary in the present work since the steady-state solution of the governing equations is obtained by an implicit time-marching technique.

Gear's method consists of a series of implicit time integration schemes which are characterized by large stability limits. The method is of the backward differentiation type and possesses a variable order of accuracy in time which can be controlled by the number of time levels used.

In general, after space discretization one arrives at a set of continuous ordinary differential equations in time,

$$\mathbf{M}\dot{\mathbf{U}} + \mathbf{K}\mathbf{U} = \mathbf{F} \qquad 0 \le t \le T \tag{2.21}$$

where $\mathbf{U}$ represents the global vector of nodal values of the solution variable $\mathrm{U}(x,y,z,t)$, $\mathbf{M}$ is the mass matrix, $\mathbf{K}$ is the influence matrix, $\mathbf{F}$ is the source vector and T denotes the time span over which $\mathbf{U}$ is computed. Applying the *k*th order Gear scheme to the time term in equation (2.21) yields:

$$\left(\frac{\partial \mathbf{U}}{\partial t}\right)^{t} = \frac{1}{\Delta t}\left(\alpha_0 \mathbf{U}^{t} + \sum_{i=1}^{k} \alpha_i \mathbf{U}^{t-i\Delta t}\right) \tag{2.22}$$

where α_i (for $i=0,1,\ldots,k$) and k are the coefficients and required order of time accuracy of the Gear scheme, respectively. In *NS3D*'s time-marching approach to a steady-state solution, the first order Gear scheme ($k=1, \alpha_0=1$ and $\alpha_1=-1$), which is identical to the implicit Euler backward scheme, is applied to the time-dependent terms of the system of equations, giving

$$\mathbf{U}_t = \mathbf{U}_{t-1} + \Delta t \mathbf{M}^{-1}[\mathbf{F} - \mathbf{K}\mathbf{U}]_t \tag{2.23}$$

The Euler backward scheme is commonly used as a matrix preconditioner to augment the diagonal dominance and hence the stability of time-marching approaches to steady-state problems. Details of the temporal discretization of the full Navier-Stokes system of equations are given in Appendices B and C.

2.6 Newton Linearization

For reasons of stability and robustness of the numerical scheme, the continuity and momentum equations (2.2-2.3) will be solved simultaneously (for details, see Chapter Five). To keep the coefficient matrix to a practical size, the energy equation (2.4) is solved in a segregated manner. The nonlinear governing equations are linearized by a Newton method whereby each variable of the solution vector, $\mathbf{U} = (\rho u, \rho v, \rho w, p)$, is expressed in delta form, $\Delta \mathbf{U} = \mathbf{U}^{n+1} - \mathbf{U}^{n}$. After neglecting second order terms and substituting equations (2.13) and (2.16) into the Newton linearized system of equations, the delta form of the continuity, momentum and energy equations is assembled, over the elements of the domain, in terms of the nodal unknowns, $\Delta(\rho u)$, $\Delta(\rho v)$, $\Delta(\rho w)$, Δp and ΔT_0. The discretized equations may be written compactly as:

$$\sum_{e=1}^{nelem}\left[\sum_{j=1}^{ndperl}\left\{\left[k^{\rho u}{}_{ij}\right]_p \Delta(\rho u)_j + \left[k^{\rho v}{}_{ij}\right]_p \Delta(\rho v)_j + \left[k^{\rho w}{}_{ij}\right]_p \Delta(\rho w)_j + \left[k^{p}{}_{ij}\right]_p \Delta(p)_j\right\}\right] = -Res_p^n \quad (2.24)$$

$$\sum_{e=1}^{nelem}\left[\sum_{j=1}^{ndperl}\left\{\left[k^{\rho u}{}_{ij}\right]_{\rho u} \Delta(\rho u)_j + \left[k^{\rho v}{}_{ij}\right]_{\rho u} \Delta(\rho v)_j + \left[k^{\rho w}{}_{ij}\right]_{\rho u} \Delta(\rho w)_j + \left[k^{p}{}_{ij}\right]_{\rho u} \Delta(p)_j\right\}\right] = -Res_{\rho u}^n \quad (2.25)$$

$$\sum_{e=1}^{nelem}\left[\sum_{j=1}^{ndperl}\left\{\left[k^{\rho u}{}_{ij}\right]_{\rho v} \Delta(\rho u)_j + \left[k^{\rho v}{}_{ij}\right]_{\rho v} \Delta(\rho v)_j + \left[k^{\rho w}{}_{ij}\right]_{\rho v} \Delta(\rho w)_j + \left[k^{p}{}_{ij}\right]_{\rho v} \Delta(p)_j\right\}\right] = -Res_{\rho v}^n \quad (2.26)$$

$$\sum_{e=1}^{nelem}\left[\sum_{j=1}^{ndperl}\left\{\left[k^{\rho u}{}_{ij}\right]_{\rho w} \Delta(\rho u)_j + \left[k^{\rho v}{}_{ij}\right]_{\rho w} \Delta(\rho v)_j + \left[k^{\rho w}{}_{ij}\right]_{\rho w} \Delta(\rho w)_j + \left[k^{p}{}_{ij}\right]_{\rho w} \Delta(p)_j\right\}\right] = -Res_{\rho w}^n \quad (2.27)$$

$$\sum_{e=1}^{nelem}\left[\sum_{j=1}^{ndperl}\left[k^{T_0}{}_{ij}\right]_{T_0} \Delta(T_0)_j\right] = -Res_{T_0}^n \quad (2.28)$$

where *nelem* denotes the total number of elements, Res^n represents the residual of the given governing equation at Newton step n, and (i,j) are the row and column indices. Details of equations (2.24)-(2.28) may be found in Appendices B and C.

2.7 Boundary Conditions

2.7.1 Navier-Stokes Equations

Inlet Boundary Condition: $\rho u, \rho v$ and ρw are specified.

The surface integrals of the continuity equation (A.6) are computed using the specified inlet mass flux, $\rho\vec{V}$. However, the contribution of the pressure dissipation term to the continuity contour integral is neglected due to the small value of λ. The momentum equations are not evaluated at the inlet and are replaced by the imposition of the specified inlet mass flux, $\rho\vec{V}$.

Wall Boundary Condition: $u, v, w = 0$ is imposed as the no-slip condition. The surface integral of the continuity equation is not evaluated since $\vec{V} = 0$ and the contribution of the pressure dissipation term to this integral is neglected due to the small value of λ. The momentum equations are replaced at the wall nodes by the imposition of $\vec{V} = 0$.

Exit Boundary Condition: p is specified.

The static pressure is specified at all exit points and replaces the continuity equation. The normal derivatives of $\vec{V}$ are neglected in the momentum contour integrals. All artificial dissipation terms in the momentum surface integrals are also neglected due to the small value of the artificial dissipation coefficients.

Symmetry Boundary Condition:

For the continuity equation zero normal mass flux is imposed by neglecting the surface integrals. In the computation of the momentum surface integrals the convective terms are assumed small and neglected. The shear stress is also set to zero as one of the boundary conditions. All artificial dissipation terms in the momentum surface integrals are neglected due to the small value of the artificial dissipation coefficients.

2.7.2 Euler Equations

Inlet Boundary Condition: $\rho u, \rho v$ and ρw are specified.

The inlet boundary conditions are identical to those of the Navier-Stokes equations.

Wall Boundary Condition: $\vec{V}\cdot\vec{n}=0$ is specified.

This boundary condition is imposed by not evaluating $\vec{V}\cdot\vec{n}$ in the surface integral of the continuity equation. The contribution of the pressure dissipation term to this integral is neglected due to the small λ value. The convective terms in the surface integrals of the momentum equations are assumed small and neglected. All artificial dissipation terms in the momentum surface integrals are also neglected due to the small value of the artificial dissipation coefficients.

Exit Boundary Condition: p is specified.

The static pressure is specified at all exit points and replaces the continuity equation. All artificial dissipation terms in the momentum surface integrals are neglected due to the small value of the artificial dissipation coefficients.

Symmetry Boundary Condition:

The symmetry and wall boundary conditions are identical for the Euler equations.

FIGURES

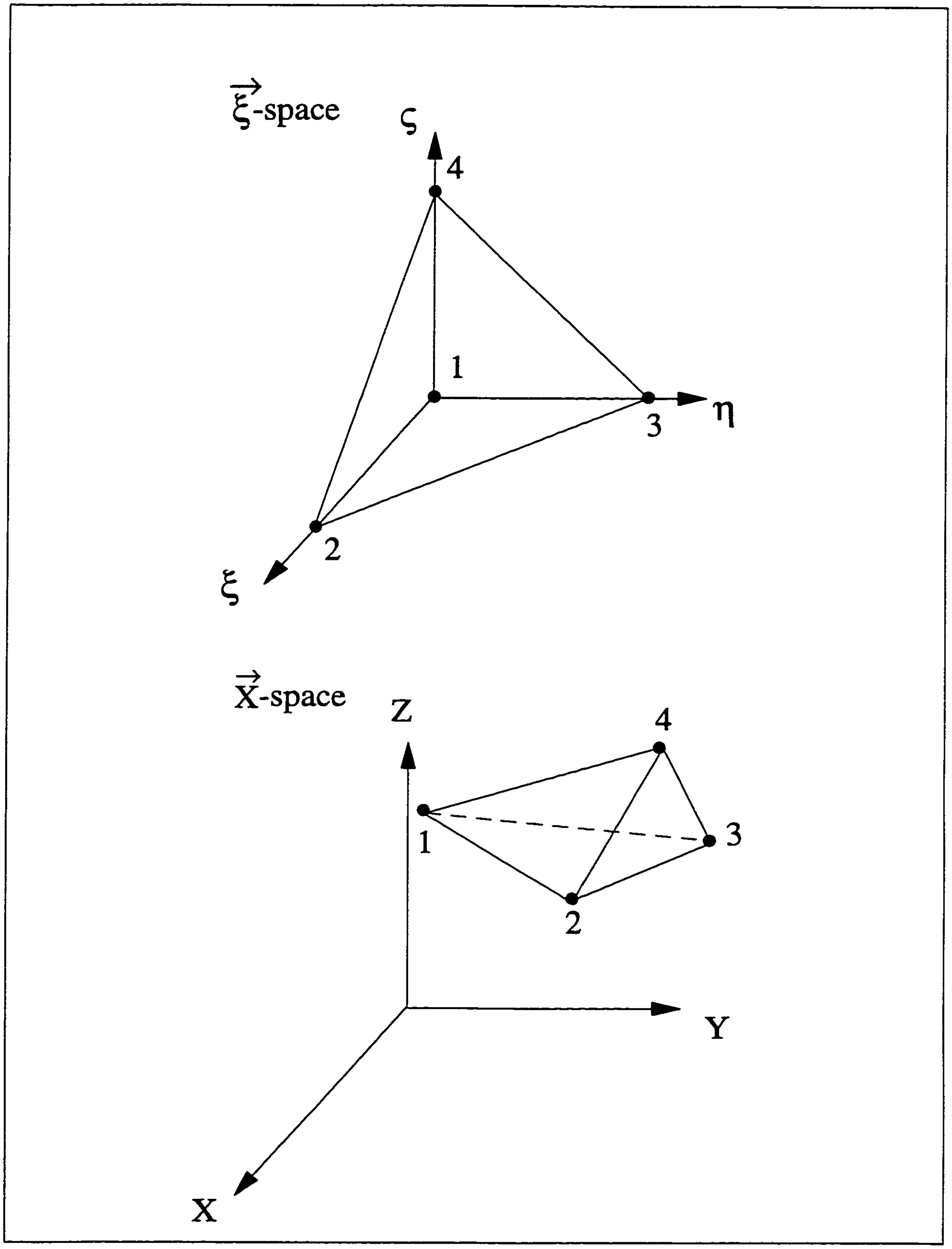


Figure 2.1 Linear Tetrahedral Finite Element

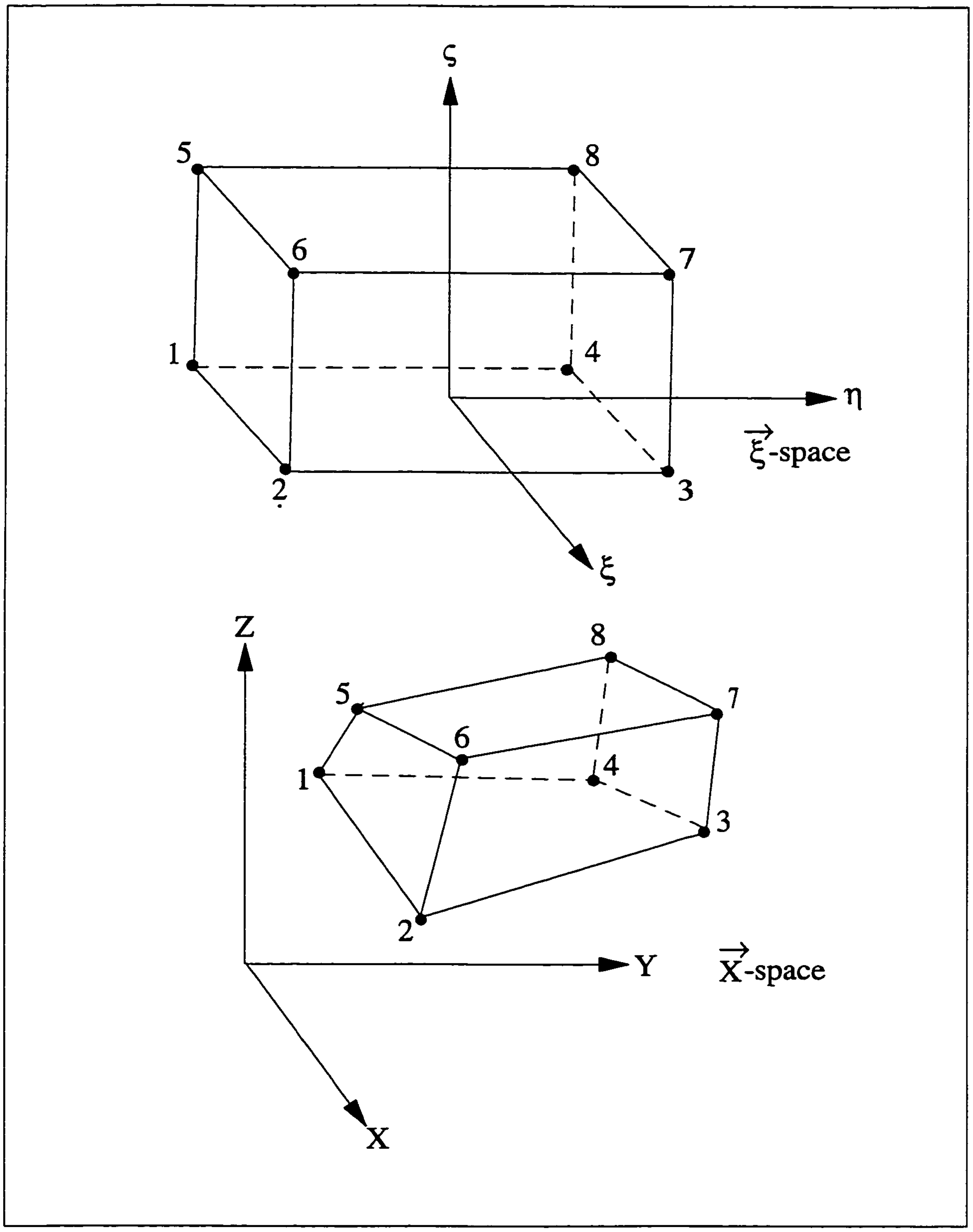


Figure 2.2 Trilinear Hexahedral Finite Element

3. Error Estimation

3.1 Definition of an Optimal Mesh

A good definition of an optimal mesh, as this relates to the numerical properties of the solution scheme, is still lacking in the literature. However, most practitioners of grid adaptation have chosen to define an optimal mesh criterion depending on whether or not the number of nodes used to mesh the computational domain remains fixed. If the number of vertices is fixed, as is the case with an adaptive mesh movement strategy, the optimal mesh would be one in which the error is minimized. If, however, the number of grid points are allowed to vary as in the application of mesh enrichment schemes, the optimal mesh is defined as one which meets a user-specified error tolerance while employing a minimum number of nodes. Such cases have been referred in the literature as *minimum error* and *maximum efficiency mesh problems*, respectively [37].

In the present work, a mesh movement strategy, which redistributes the grid points so the nodal density is large in regions of high error and small in regions of low error, will be implemented. To extend the above definition of an optimal mesh criterion, based on a fixed number of nodes, to this work, the objective of the adaptive strategy will be to equidistribute, as much as possible, the error over the edges of the elements.

3.2 Accuracy of the Finite Element Approximation

Once a finite element solution to a given flow problem has been computed, the question of the accuracy of the approximation naturally arises. The quality of the finite element approximation and its behaviour as the mesh is refined are fundamental issues addressed by users of numerical methods.

The quality of the finite element solution is assessed by the local approximation error e_h, which is defined as the difference between the exact solution u and its finite element approximation u_h,

$$e_h = u - u_h \tag{3.1}$$

where the subscript h represents a mesh length parameter. Since the actual error can never be calculated unless the exact solution is known, there may seem to be little reason for computing e_h. However, even when u is unknown, it is possible to build estimates of the approximation error and to determine if the error decreases as h decreases and the number of elements grows larger. Such bounds on the approximation error, which are measured in an appropriate norm and expressed in terms of h, provide useful information regarding the behaviour of the solution error as the mesh is modified.

The magnitude of the approximation error and its bound are measured using the norms and semi-norms associated with the Sobolev spaces [62]. Sobolev spaces serve as one of the most important examples of normed spaces in the theory of partial differential equations. In this context, the question of the accuracy of finite element approximations may be stated as follows: given an m^{th} Sobolev space H^m and a finite element space V_h, how well can a given solution $u \in H^m$ be approximated for any member of V_h? The answer is detailed in the numerical analysis work of Ciarlet [57]. A result from finite element interpolation theory, Céa's Lemma, states that the approximation error is always bounded by the interpolation error

$$\|u-u_h\| \leq C\|u-\Pi_h u\| \tag{3.2}$$

where $\Pi_h u$ is the interpolate of the solution u in the finite element space and C is some positive constant. Therefore, the approximation error must exhibit the same rate of convergence as the interpolation error and determining the bounds for the interpolation error provides a bound on the approximation error. Ciarlet has demonstrated that for any $u \in H^{k+1}(\Omega)$ there exists a positive constant C' such that

$$\|u-\Pi_h u\|_{H_m(\Omega)} \leq C' \frac{h^{k+1}}{\rho^m} |u|_{H_{k+1}(\Omega)} \tag{3.3}$$

where k is a fixed integer,

m is an integer such that $0 \leq m \leq k+1$,

$\|\cdot\|_{H^m(\Omega)}$ is the H^mnorm,

$|\cdot|_{H^{k+1}(\Omega)}$ is the H^{k+1} semi-norm,

ρ^e is the diameter of the largest sphere contained in element e,

$\rho = \min\limits_{1 \leq e \leq nelem} (\rho^e)$,

and $h = \max\limits_{1 \leq e \leq nelem} (h^e)$.

As an example, consider a collection of finite element spaces consisting of isoparametric elements on a domain $\Omega \subset R^n$, $n \geq 1$ [62]. One finds for every $u \in H^2$,

$$\|u - \Pi_h u\|_{H^0} \leq C'' h^2 |u|_{H^2} \tag{3.4}$$

where

$$\|u\|_{H^0(\Omega)} = \|u\|_{L^2(\Omega)} = \left[\int_\Omega u^2 d\Omega \right]^{1/2} \tag{3.5}$$

$$|u|_{H^2} = \left[\int_\Omega \sum_{|\alpha|=2} |D^\alpha u|^2 dx \right]^{1/2} \tag{3.6}$$

The symbol α denotes $\alpha = (\alpha_1, .., \alpha_n)$ and $D^\alpha u = \partial^{|\alpha|} u / \partial^{\alpha_1} x_1 \partial^{\alpha_2} x_2$ represents the generalized derivatives of order ≤ 2. According to equation (3.4), the approximation error in the L_2-norm is bounded, to within a multiplicative constant C'', by the product of the second derivatives of the solution variable u and the square of the mesh parameter h.

Since such estimates (3.3-3.4) require no information from the actual finite element solution and are known prior to the determination of the solution, they are termed *a priori* estimates. Other more elaborate estimates of accuracy, which derive information from the finite element solution itself and may only be computed after the solution has been obtained, are called *a posteriori* estimates and will be used in the work of this thesis.

3.3 Anisotropic Grid Transformations

As described in the previous section, the finite element interpolation error serves as the upper bound for the approximation error, thus implying that a control of the interpolation error will result in a control of the solution error itself. Simpson and D'Azevedo [37, 39] have investigated the use of coordinate transformations that are derived from error properties in approximation to generate optimal meshes for linear interpolation. The distribution of nodes in the optimal mesh is specified by an error estimate in the form of a symmetric metric tensor, namely, the Hessian or matrix of second derivatives of the solution being sought. The desired anisotropic mesh in physical space is constructed by a coordinate transformation based on this error metric tensor. This results in an isotropic mesh in the transformed metric space but an anisotropic mesh in physical space. Such anisotropic grid transformations serve as the motivation for the construction of the edge-based interpolation error estimate to be described in Section 3.5. A summary of the results of Simpson's optimal error control studies on triangular meshes follows. It should be noted that no such studies have been undertaken for higher order interpolants, but experience has shown that Simpson's conclusion works very well even for quadratic interpolations.

Simpson sought the "best" triangle shape that minimized the maximum interpolation error and derived the linear transformation that maps a regular triangular mesh into an optimal mesh for the interpolation of a quadratic function, $q(\vec{x})$. This transformation is suggested by the simplification of the exact error function, $\varepsilon(\vec{x})$, obtained in the linear interpolation of $q(\vec{x})$. As described in [39], the error function takes the form

$$\varepsilon(\vec{\mathbf{x}}) = (\Delta\vec{\mathbf{x}})\mathbf{H}(\Delta\vec{\mathbf{x}})^T \tag{3.7}$$

where $\mathbf{H}$ represents the 2x2 Hessian matrix of $q(\vec{\mathbf{x}})$. The isolines of $\varepsilon(\vec{\mathbf{x}})$ consist of concentric ellipses if the determinant of the Hessian matrix is greater than zero. The symmetric matrix $\mathbf{H}$ may be diagonalized as

$$\mathbf{H} = \mathbf{Q}^T \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \mathbf{Q} = \mathbf{R}^T \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \mathbf{R} \tag{3.8}$$

where $\mathbf{Q}$ is the matrix of eigenvectors, λ_1 and λ_2 are the eigenvalues of $\mathbf{H}$, and

$$\mathbf{R} = \begin{bmatrix} \sqrt{|\lambda_1|} & 0 \\ 0 & \sqrt{|\lambda_2|} \end{bmatrix} \mathbf{Q} \tag{3.9}$$

represents the anisotropic grid transformation matrix. By applying a change of variables, $\vec{\mathbf{x}}'^T = \mathbf{R}\vec{\mathbf{x}}^T$, the error function can be rewritten as

$$\varepsilon'(\vec{\mathbf{x}}') = (\Delta x')^2 + (\Delta y')^2 = |\Delta\vec{\mathbf{x}}'|^2 \tag{3.10}$$

where $\varepsilon'(\vec{\mathbf{x}}')$ denotes the corresponding function under the transformation $\mathbf{R}$ with its isolines now transformed to concentric circles. If T' is the transformed image of the triangle T with vertices $[(x_1', y_1'), (x_2', y_2'), (x_3', y_3')]$, the circle circumscribing T' will be the isoline of zero error. The radius of this circle is thus directly proportional to the maximum error attainable. An equilateral triangle is the optimal shape being sought since it would cover the maximal area for a given radius or error level. Simpson concluded that an isotropic mesh comprising equilateral triangles over the transformed plane corresponds to an "optimally efficient" mesh, defined as one that achieves a specified error tolerance with the fewest elements. The inverse transformation $\mathbf{R}^T$ maps this "optimal" mesh of equilateral triangles into the desired anisotropic grid in the original plane.

3.4 Euclidean and Riemannian Metrics

The generation of optimal meshes through coordinate transformations has been demonstrated primarily on triangular meshes for quadratic functions where the Hessian $\mathbf{H}$ is constant [37]. Quadratic functions may be viewed as local approximations of general functions with varying $\mathbf{H}(\vec{\mathbf{x}})$. The local coordinate transformations required to construct nearly optimal meshes therefore become nonlinear. Applying transformations based on variable $\mathbf{H}(\vec{\mathbf{x}})$ to the original space yields, in general, a Riemannian space with its metric defined by $\mathbf{H}(\vec{\mathbf{x}})$. These transformations raise the issue of whether the original space can be mapped to a "flat" space. The transformed space is characterized as "flat" if its metric tensor can be reduced to the Euclidean metric.

Two important structures of Euclidean and Riemannian spaces include the *scalar product* and *metric*, which are defined below [65].

1. A *scalar product* $\langle\cdot,\cdot\rangle$ on a vector space A is a map $\langle\cdot,\cdot\rangle : A \times A \rightarrow \Re$. In other words, $\langle\cdot,\cdot\rangle$ assigns to an ordered pair $\vec{\mathbf{x}}, \vec{\mathbf{y}} \in A$ a real number denoted $\langle \vec{\mathbf{x}}, \vec{\mathbf{y}} \rangle$ with the following properties. Let $\vec{\mathbf{x}}, \vec{\mathbf{y}}, \vec{\mathbf{z}} \in A$ and $\alpha \in \Re$. Then

$$\begin{aligned}
&a. \quad \langle \vec{\mathbf{x}},\vec{\mathbf{y}}\rangle = \langle \vec{\mathbf{y}},\vec{\mathbf{x}}\rangle \\
&b. \quad \langle \alpha\vec{\mathbf{x}},\vec{\mathbf{y}}\rangle = \alpha\langle \vec{\mathbf{x}},\vec{\mathbf{y}}\rangle \\
&c. \quad \langle \vec{\mathbf{x}}+\vec{\mathbf{y}},\vec{\mathbf{z}}\rangle = \langle \vec{\mathbf{x}},\vec{\mathbf{z}}\rangle + \langle \vec{\mathbf{y}},\vec{\mathbf{z}}\rangle \\
&d. \quad \langle \vec{\mathbf{x}},\vec{\mathbf{x}}\rangle \geq 0
\end{aligned} \tag{3.11}$$

2. A *Euclidean metric* d on A is a map $d{:}A \times A \rightarrow \Re^{+}$ with the following properties. Let $\vec{\mathbf{x}},\vec{\mathbf{y}},\vec{\mathbf{z}} \in A$. Then

$$\begin{aligned}
&a. \quad d(\vec{\mathbf{x}},\vec{\mathbf{y}}) = d(\vec{\mathbf{y}},\vec{\mathbf{x}}) \\
&b. \quad d(\vec{\mathbf{x}},\vec{\mathbf{y}}) > 0 \qquad \textit{for} \quad \vec{\mathbf{x}} \neq \vec{\mathbf{y}} \\
&c. \quad d(\vec{\mathbf{x}},\vec{\mathbf{y}}) = 0 \qquad \textit{for} \quad \vec{\mathbf{x}} = \vec{\mathbf{y}} \\
&d. \quad d(\vec{\mathbf{x}},\vec{\mathbf{z}}) \leq d(\vec{\mathbf{x}},\vec{\mathbf{y}}) + d(\vec{\mathbf{y}},\vec{\mathbf{z}})
\end{aligned} \tag{3.12}$$

3. If $\langle \cdot,\cdot \rangle$ is a scalar product on A and $\vec{\mathbf{x}},\vec{\mathbf{y}} \in A$, then

$$d(\vec{\mathbf{x}},\vec{\mathbf{y}}) = \langle \vec{\mathbf{x}}-\vec{\mathbf{y}},\vec{\mathbf{x}}-\vec{\mathbf{y}}\rangle^{1/2} \tag{3.13}$$

defines a *metric* on A.

4. In $\Re^n$, a scalar product may take the form $\langle \vec{\mathbf{x}},\vec{\mathbf{y}}\rangle = \vec{\mathbf{x}}^T\mathbf{M}\vec{\mathbf{y}}$ where $\mathbf{M}$ is an $n \times n$ symmetric positive-definite matrix. As an example,

$$d(\vec{\mathbf{x}},\vec{\mathbf{y}}) = \sqrt{(\vec{\mathbf{x}}-\vec{\mathbf{y}})^T\mathbf{M}(\vec{\mathbf{x}}-\vec{\mathbf{y}})} \tag{3.14}$$

for all $\vec{\mathbf{x}},\vec{\mathbf{y}} \in \Re^n$ defines a Euclidean metric on $\Re^n$.

If one considers only metrics which are induced by a scalar product, then a metric defined on $\Re^n$ may be merely identified by the matrix $\mathbf{M}$ which defines the associated scalar product. For example, a Euclidean metric (3.14) on $\Re^2$ will be described by its 2×2 symmetric matrix $\mathbf{M} = \{m_{ij}\}$, for $1 \le i \le j \le 2$. If $\mathbf{M}$ is the identity matrix, then equation (3.14) represents the usual Euclidean measure of length. For example, the length of an edge between nodes I and J in the Euclidean metric, denoted as $d_E(\bar{\mathbf{x}}_I, \bar{\mathbf{x}}_J)$, is defined as

$$d_E(\bar{\mathbf{x}}_I, \bar{\mathbf{x}}_J) = \sqrt{(\bar{\mathbf{x}}_J - \bar{\mathbf{x}}_I)^T (\bar{\mathbf{x}}_J - \bar{\mathbf{x}}_I)} \tag{3.15}$$

In the general case, $\mathbf{M}$ may be diagonalized as

$$\mathbf{M} = [\mathbf{P}(\alpha)]^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} [\mathbf{P}(\alpha)] \tag{3.16}$$

where $\lambda_i > 0$ for $i=1, 2$ are the eigenvalues of $\mathbf{M}$ and $\mathbf{P}(\alpha)$ is a matrix which rotates the x- and y-axes through an angle α. A unit circle in this metric $\mathbf{M}$ would be an ellipse whose principal axes have lengths inversely proportional to the square roots of the eigenvalues, $\sqrt{\lambda_1}$ and $\sqrt{\lambda_2}$ (fig. 3.1). Similarly, equilateral triangles with edges of unit length in this metric would have a characteristic length $\sqrt{\lambda_1}^{-1}$ along the direction of angle α and $\sqrt{\lambda_2}^{-1}$ in the perpendicular direction. It has been demonstrated that if T is a non-degenerate triangle, there exists a unique metric $\mathbf{M}$, induced by a scalar product, in which T is an equilateral triangle with edges of unit length [21].

Similarly, in 3-D, a sphere of unit diameter in a 3×3 metric $\mathbf{M}$ would transform to an ellipsoid whose principal axes have lengths inversely proportional to the square roots of the three eigenvalues of $\mathbf{M}$.

The above discussion deals with a Euclidean metric $\mathbf{M}$ whose entries are all constant. Now consider a function $\mathbf{M}(\vec{\mathbf{x}})$ whose entries vary with $\vec{\mathbf{x}}$. In 2-D, one may diagonalize such a matrix as

$$\mathbf{M}(\vec{\mathbf{x}}) = [\mathbf{P}(\alpha(\vec{\mathbf{x}}))]^{-1} \begin{bmatrix} \lambda_1(\vec{\mathbf{x}}) & 0 \\ 0 & \lambda_2(\vec{\mathbf{x}}) \end{bmatrix} [\mathbf{P}(\alpha(\vec{\mathbf{x}}))] \tag{3.17}$$

If $\mathbf{M}(\vec{\mathbf{x}})$ is also symmetric, as well as positive-definite for all $\vec{\mathbf{x}}$, one arrives at the definition of a Riemannian metric [21, 37]. Given a varying $\mathbf{M}(\vec{\mathbf{x}})$, the arc length of a curve β in this metric, denoted as $d_M(\beta)$, would be given by

$$d_M(\beta) = \int_0^1 \sqrt{\vec{\mathbf{s}}(t)'^T \mathbf{M}(\vec{\mathbf{s}}(t)) \vec{\mathbf{s}}(t)'} dt \qquad for \qquad t \in [0,1] \tag{3.18}$$

where $\vec{\mathbf{s}}(t)$ is a parametric representation of the curve β. Since $\mathbf{M}(\vec{\mathbf{x}})$ is a function of the space coordinates, equation (3.18) defines a Riemannian metric and its evaluation requires numerical integration. For example, the length of an edge between nodes I and J in the Riemannian metric, denoted as $d_M(\vec{\mathbf{x}}_I, \vec{\mathbf{x}}_J)$, is given by equation (3.18) where $\vec{\mathbf{s}}(t) = \vec{\mathbf{x}}_I + (\vec{\mathbf{x}}_J - \vec{\mathbf{x}}_I)t$ for $t \in [0,1]$. The importance of equation (3.18) will be evident in the next Section, entitled "Construction of Interpolation Error Estimate".

3.5 Construction of Interpolation Error Estimate

3.5.1 Introduction

The edge-based error estimate applied in this thesis may be classified as an interpolation error estimate. This type of estimate is problem-independent and, therefore, is relatively simple to build. Its evaluation is not CPU-intensive and its computation can be written in a post-processing subroutine which may be easily added to an existing solver code. Such an error estimate meets the criteria of efficiency and portability demanded by an established industrial code.

The edge-based error estimate is constructed using the bound, equation (3.4), from finite element interpolation theory based on *a priori* error estimates. This error bound is in all respects analogous to the Taylor series truncation error analysis of finite difference approximations of partial differential equations.

3.5.2 Edge-Based Interpolation Error Estimate

Consider a 1-D problem in which $u(x)$ is approximated by a finite element solution $u_h(x)$ obtained using linear interpolation. The approximation error over an element of length h_e , $e_h(x)$, is defined to be

$$e_h(x) = u(x) - u_h(x) \tag{3.19}$$

at a point x where $x \in [0, h_e]$. In the following derivation, for the sake of simplicity, the subscript e will be dropped from the element length h_e . Referring to fig. 3.2, the finite element solution may be written in terms of its nodal values as

$$u_h(x) = u_0 + \left(\frac{u_1 - u_0}{h}\right)x = \left(1 - \frac{x}{h}\right)u_0 + \frac{x}{h}u_1 \tag{3.20}$$

Expansion of u_1 as a Taylor series about $x = 0$ yields

$$u_1 = u_0 + hu_0' + \frac{h^2}{2}u_0'' + \ldots \tag{3.21}$$

A simplification of equation (3.20) leads to

$$u_1 = \frac{u_h h}{x} - \frac{u_0 h}{x} + u_0 \tag{3.22}$$

which is then substituted into equation (3.21) to give the following expression:

$$u_h(x) = u_0 + u_0' x + u'' \frac{xh}{2} + \ldots \tag{3.23}$$

Similarly, expansion of the exact solution $u(x)$ as a Taylor series about $x = 0$ gives

$$u(x) = u_0 + u_0' x + u'' \frac{x^2}{2} + \ldots \tag{3.24}$$

Replacing equations (3.23-3.24) into (3.19) and neglecting higher order terms leads to an expression for the error over an element, that is,

$$e_h(x) = \left(\frac{x^2}{2} - \frac{xh}{2} \right) \frac{d^2 u_h}{dx^2} \tag{3.25}$$

Following the demonstration by Peraire et al. [40], the root-mean-square value of the error over an element spanning the interval $[0,h_e]$ can be evaluated as

$$
\begin{aligned}
E_{el}^{rms} &= \left\{ \int_0^{h_e} \frac{e_h{}^2}{h_e} dx \right\}^{1/2} \\
&= \frac{h_e^2}{\sqrt{120}} \left| \frac{d^2 u_h}{dx^2} \right|
\end{aligned}
\tag{3.26}
$$

The interpolation error for this 1-D problem is proportional to the product of the second derivative of the variable u and the square of the element length. Knowledge of the second derivatives over an element provides an interpolation error estimate, ε_I, taking the form:

$$
\varepsilon_I \cong C h_e{}^2 \left| \frac{d^2 u_h}{dx^2} \right|_e
\tag{3.27}
$$

where C is some multiplicative constant. It should be noted that in the construction of ε_I, one is only concerned with the absolute value or magnitude of the error and not in its change of sign. The approach taken in this thesis is thus to equidistribute the error over the *edges* of the element. By imposing that ε_I be equal in all elements, the new length h_e must satisfy the condition

$$
h_e{}^2 \left| \frac{d^2 u_h}{dx^2} \right|_e = constant\ C'
\tag{3.28}
$$

In an adaptive nodal redistribution scheme C' would denote some arbitrary constant, while in a remeshing or grid refinement/coarsening method C' would represent a user-specified error tolerance.

Extending these ideas to the 3-D case, the second derivative of the solution variable u_h in the direction of a unit vector $\vec{\mathbf{e}}$ may be written as

$$\frac{\partial^2 u_h}{\partial \vec{\mathbf{e}}^2} = \vec{\mathbf{e}}\mathbf{H}(\vec{\mathbf{x}})\vec{\mathbf{e}}^T \tag{3.29}$$

where $\mathbf{H}(\vec{\mathbf{x}})$ represents the 3×3 solution Hessian:

$$\mathbf{H}(\vec{\mathbf{x}}) = \left\{u_{h,ij}\right\} = \begin{pmatrix} \frac{\partial^2 u_h}{\partial x^2} & \frac{\partial^2 u_h}{\partial x \partial y} & \frac{\partial^2 u_h}{\partial x \partial z} \\ \frac{\partial^2 u_h}{\partial x \partial y} & \frac{\partial^2 u_h}{\partial y^2} & \frac{\partial^2 u_h}{\partial y \partial z} \\ \frac{\partial^2 u_h}{\partial x \partial z} & \frac{\partial^2 u_h}{\partial y \partial z} & \frac{\partial^2 u_h}{\partial z^2} \end{pmatrix} \tag{3.30}$$

$\mathbf{H}(\vec{\mathbf{x}})$ may be decomposed as

$$\mathbf{H}(\vec{\mathbf{x}}) = \mathbf{Q}^T(\vec{\mathbf{x}}) \begin{pmatrix} \lambda_1(\vec{\mathbf{x}}) & 0 & 0 \\ 0 & \lambda_2(\vec{\mathbf{x}}) & 0 \\ 0 & 0 & \lambda_3(\vec{\mathbf{x}}) \end{pmatrix} \mathbf{Q}(\vec{\mathbf{x}}) = \mathbf{Q}^T(\vec{\mathbf{x}})\Lambda(\vec{\mathbf{x}})\mathbf{Q}(\vec{\mathbf{x}}) \tag{3.31}$$

where $\mathbf{Q}(\vec{\mathbf{x}})$ is the matrix of eigenvectors and $\Lambda(\vec{\mathbf{x}})$ is the diagonal matrix containing the eigenvalues of $\mathbf{H}(\vec{\mathbf{x}})$. Since one is only interested in the magnitude of the entries of $\mathbf{H}(\vec{\mathbf{x}})$, a modified matrix $\tilde{\mathbf{H}}(\vec{\mathbf{x}})$ will be defined which possesses the same eigendirections as $\mathbf{H}(\vec{\mathbf{x}})$ and the absolute value of the corresponding eigenvalues, that is,

$$\tilde{\mathbf{H}}(\vec{\mathbf{x}}) = \mathbf{Q}^T(\vec{\mathbf{x}}) \begin{pmatrix} |\lambda_1(\vec{\mathbf{x}})| & 0 & 0 \\ 0 & |\lambda_2(\vec{\mathbf{x}})| & 0 \\ 0 & 0 & |\lambda_3(\vec{\mathbf{x}})| \end{pmatrix} \mathbf{Q}(\vec{\mathbf{x}}) = \mathbf{Q}^T(\vec{\mathbf{x}})|\Lambda(\vec{\mathbf{x}})|\mathbf{Q}(\vec{\mathbf{x}}) \tag{3.32}$$

By considering only the absolute value of the eigenvalues, one obtains a symmetric positive-definite matrix $\tilde{\mathbf{H}}(\vec{\mathbf{x}})$. $\tilde{\mathbf{H}}(\vec{\mathbf{x}})$ may also be written as

$$\tilde{\mathbf{H}}(\vec{\mathbf{x}}) = \tilde{\mathbf{R}}^T(\vec{\mathbf{x}})\tilde{\mathbf{R}}(\vec{\mathbf{x}}) \tag{3.33}$$

where $\tilde{\mathbf{R}}(\vec{\mathbf{x}}) = \sqrt{|\Lambda(\vec{\mathbf{x}})|}\mathbf{Q}(\vec{\mathbf{x}})$. The transformation $\tilde{\mathbf{R}}(\vec{\mathbf{x}})^T$ of a unit sphere would be an ellipsoid, rotated through an angle α, whose semi-major and semi-minor axes are reciprocals of the square roots of the eigenvalues, $|\Lambda_i(\vec{\mathbf{x}})|$, for i=1,2, and 3. Thus, one may obtain a directionally stretched grid by mapping a uniform mesh using the coordinate transformation $\tilde{\mathbf{R}}(\vec{\mathbf{x}})^T$.

It can be seen that the second derivative of the solution variable is bounded in any direction $\vec{\mathbf{e}}$ as follows:

$$\left|\frac{\partial^2 u_h}{\partial \vec{\mathbf{e}}^2}\right| = \left|\vec{\mathbf{e}}\mathbf{H}(\vec{\mathbf{x}})\vec{\mathbf{e}}^T\right| \leq \vec{\mathbf{e}}\tilde{\mathbf{H}}(\vec{\mathbf{x}})\vec{\mathbf{e}}^T \tag{3.34}$$

Following the form of the interpolation error estimate in the 1-D case, the adaptation criterion for 3-D problems may be written as

$$\varepsilon_I \cong Ch^2\vec{\mathbf{e}}\tilde{\mathbf{H}}(\vec{\mathbf{x}})\vec{\mathbf{e}}^T \tag{3.35}$$

or

$$h^2\vec{\mathbf{e}}\tilde{\mathbf{H}}(\vec{\mathbf{x}})\vec{\mathbf{e}}^T = constant \quad C' \tag{3.36}$$

In the adaptive remeshing work carried out by Peraire et al. [40], the elemental root-mean-square error was distributed along the local principal directions or eigendirections. Using equation (3.36) as the guideline for the optimal mesh criterion, he defined $h=\delta_i$ $(i=1,2)$ to be the two local nodal spacings and $\vec{\mathbf{e}}=\vec{\mathbf{e}}_k$ to be the two unit eigenvectors of $\tilde{\mathbf{H}}(\vec{\mathbf{x}})$. Thus, the optimal mesh criterion (equation 3.36) reduces to

$$\delta_1^2|\lambda_1|=\delta_2^2|\lambda_2|=C' \tag{3.37}$$

and the nodal spacings in the eigendirections are computed according to $\delta_i=\left(C'/\lambda_i\right)^{1/2}$. Applying these updated values of nodal spacings and eigendirections, an adapted mesh is regenerated.

The approach undertaken in this thesis is distinctly different in that the error estimate is equidistributed over the edges of the elements, whether they be hexahedra or tetrahedra. The Euclidean length of an edge between nodes I and J is defined as

$$h=d_E(\vec{\mathbf{x}}_I,\vec{\mathbf{x}}_J)=\sqrt{(\vec{\mathbf{x}}_J-\vec{\mathbf{x}}_I)^T(\vec{\mathbf{x}}_J-\vec{\mathbf{x}}_I)} \tag{3.38}$$

and the unit vector along the direction of the edge is given by

$$\vec{\mathbf{e}}_{IJ}=(\vec{\mathbf{x}}_J-\vec{\mathbf{x}}_I)/d_E(\vec{\mathbf{x}}_I,\vec{\mathbf{x}}_J) \tag{3.39}$$

Equation (3.36) becomes

$$(\vec{\mathbf{x}}_J-\vec{\mathbf{x}}_I)\tilde{\mathbf{H}}(\vec{\mathbf{x}})(\vec{\mathbf{x}}_J-\vec{\mathbf{x}}_I)^T=C' \tag{3.40}$$

In this thesis, a mesh with edges of equal lengths is sought in the transformed plane $\tilde{\mathbf{R}}(\vec{\mathbf{x}}$ where $\tilde{\mathbf{R}}(\vec{\mathbf{x}})=\sqrt{|\Lambda(\vec{\mathbf{x}})|}\mathbf{Q}(\vec{\mathbf{x}})$. An optimal mesh is then defined as one in which the lengths of all the edges, with respect to the metric $\tilde{\mathbf{H}}(\vec{\mathbf{x}})$, are equal to $\sqrt{C'}$. Therefore, the edge-based error estimate given by equation (3.40) is precisely the length of the edge $[\vec{\mathbf{x}}_I, \vec{\mathbf{x}}_J]$ in the Riemannian metric defined by $\tilde{\mathbf{H}}(\vec{\mathbf{x}})$. The length of an edge in this metric may be interpreted as the 1-D interpolation error taken along this edge. Since $\tilde{\mathbf{H}}(\vec{\mathbf{x}})$ is a function of the space coordinates, this length must be integrated numerically according to

$$d_M(\vec{\mathbf{x}}_I, \vec{\mathbf{x}}_J) = \int_0^1 \sqrt{\vec{\mathbf{s}}(t)'^T \tilde{\mathbf{H}}(\vec{\mathbf{s}}(t)) \vec{\mathbf{s}}(t)'} dt \qquad for \quad t \in [0,1] \tag{3.41}$$

where $\vec{\mathbf{s}}(t) = \vec{\mathbf{x}}_I + (\vec{\mathbf{x}}_J - \vec{\mathbf{x}}_I)t$ parametrically represents the edge. Nodal values of the second derivatives or the entries of the symmetric solution Hessian are stored in a background tetrahedral grid and are used for interpolation of Hessians in the current grid that is being adapted. Only six of the nine entries of $\tilde{\mathbf{H}}(\vec{\mathbf{x}})$ need to be computed due to the symmetric nature of this 3x3 matrix. These six entries are stored at the nodes of a background grid and the value of $\tilde{\mathbf{H}}(\vec{\mathbf{x}})$ at any position of the domain can be interpolated during the adaptive process on this grid. This raises the question regarding the evaluation of the nodal values of the second derivatives.

3.5.3 Evaluation of Nodal Values of Second Derivatives

Since u is approximated by a linear interpolation, the first derivative would be constant over the element and the second derivative would vanish. The calculation of the second derivatives therefore requires special treatment. Consider a flow solution variable $u(\vec{\mathbf{x}})$ which is interpolated by linear tetrahedral shape functions, N_k, giving rise to the approximation

$$u(\vec{\mathbf{x}}) = \sum_{k=1}^{nnode} N_k u_k \tag{3.42}$$

One seeks to approximate the first derivative as well as the second derivative of $u(\bar{\mathbf{x}})$ in a similar manner using linear shape functions:

$$\frac{\partial u(\bar{\mathbf{x}})}{\partial x} = \sum_{k=1}^{nnode} N_k \left.\frac{\partial u}{\partial x}\right|_k \tag{3.43}$$

$$\frac{\partial^2 u(\bar{\mathbf{x}})}{\partial x^2} = \sum_{k=1}^{nnode} N_k \left.\frac{\partial^2 u}{\partial x^2}\right|_k \tag{3.44}$$

When linear shape functions are used for flow variable as in equation (3.42), the first derivative of the variable is constant over the tetrahedral element and the second derivative would be zero. However, one may recover the nodal values of the first and second derivatives based on linear interpolation as in equations (3.43) and (3.44). The recovery procedure is similar for the first and second derivatives and thus, will be demonstrated in detail only for the first derivative.

Using equation (3.42), the first derivative of the flow variable $u(\bar{\mathbf{x}})$ can be approximated by

$$\frac{\partial u(\bar{\mathbf{x}})}{\partial x} = \sum_{k=1}^{nnode} \frac{\partial N_k}{\partial x} u_k \tag{3.45}$$

Upon equating (3.43) and (3.45), one can compute the nodal values of the first derivatives. Consider node i to be fixed. To approach the value of $\frac{\partial u(\bar{\mathbf{x}})}{\partial x}$ at node i, both sides of equation (3.43), as well as equation (3.45), are multiplied by the linear shape function N_i in a scalar product sense. After forming the scalar product, equation (3.43) becomes

$$\left(\sum_{k=1}^{nnode}\frac{\partial N_k}{\partial x}u_k, N_i\right)=\int_W\sum_{k=1}^{nnode}\frac{\partial N_k}{\partial x}u_kN_idW \tag{3.46}$$

$$=\sum_{T\in W}\sum_{k\in T}u_k\int_T\sum_{k=1}^{nnode}\frac{\partial N_k}{\partial x}N_idW \tag{3.47}$$

Because the shape functions are linear, their first derivative is constant by element. Since $N_i=0$ in all tetrahedra elements T which do not contain the node i, equation (3.47) is further simplified to

$$\left(\sum_{k=1}^{nnode}\frac{\partial N_k}{\partial x}u_k, N_i\right)=\sum_{T:i\in T}\sum_{k\in T}u_k\left.\frac{\partial N_k}{\partial x}\right|_T\int_T N_idW \tag{3.48}$$

Similarly, multiplication in the scalar product sense of equation (3.45) with the linear shape function N_i yields

$$\left(\sum_{k=1}^{nnode}N_k\left.\frac{\partial u}{\partial x}\right|_k, N_i\right)=\int_W\sum_{k=1}^{nnode}N_k\left.\frac{\partial u}{\partial x}\right|_k N_idW \tag{3.49}$$

$$=\sum_{T\in W}\sum_{k\in T}\left.\frac{\partial u}{\partial x}\right|_k\int_T N_kN_idW \tag{3.50}$$

$$=\sum_{T:i\in T}\sum_{k\in T}\left.\frac{\partial u}{\partial x}\right|_k\int_T N_kN_idW \tag{3.51}$$

Mass lumping at node i would then give the following approximation

$$\left(\sum_{k=1}^{nnode} N_k \left.\frac{\partial u}{\partial x}\right|_k , N_i\right) \approx \sum_{T:i\in T} \left.\frac{\partial u}{\partial x}\right|_i \int_T N_i \sum_{k\in T} N_k dW \tag{3.52}$$

Since the sum of the shape functions over a tetrahedral element is one, the above equation further simplifies to

$$\left(\sum_{k=1}^{nnode} N_k \left.\frac{\partial u}{\partial x}\right|_k , N_i\right) \approx \sum_{T:i\in T} \left.\frac{\partial u}{\partial x}\right|_i \int_T N_i dW \tag{3.53}$$

$$\left(\sum_{k=1}^{nnode} N_k \left.\frac{\partial u}{\partial x}\right|_k , N_i\right) \approx \left.\frac{\partial u}{\partial x}\right|_i \int_{supp(i)} N_i dW \tag{3.54}$$

where *supp(i)* denotes the support of node *i* and is defined as the union of elements *T* such that *i* belongs to *T*.

Combining equations (3.48) and (3.54) leads to an expression for the desired quantity, namely, the nodal value of the first derivative based on a linear interpolation:

$$\left.\frac{\partial u}{\partial x}\right|_i \int_{supp(i)} N_i dW = \sum_{T:i\in T} \sum_{k\in T} u_k \left.\frac{\partial N_k}{\partial x}\right|_T \int_T N_i dW \tag{3.55}$$

$$\left.\frac{\partial u}{\partial x}\right|_i = \frac{\displaystyle\sum_{T:i\in T} \sum_{k\in T} u_k \left.\frac{\partial N_k}{\partial x}\right|_T \int_T N_i dW}{\displaystyle\int_{supp(i)} N_i dW} \tag{3.56}$$

Therefore, one may approximate over an element the first derivative of a flow variable based on linear interpolation of its nodal values, as in equation (3.43), where the nodal value of the first derivative is evaluated as in (3.56).

The same procedure may be carried out in the approximation of the nodal values of the second derivatives of the flow variable. In this case, one would arrive at the following expression,

$$\left.\frac{\partial^2 u}{\partial x^2}\right|_i = \frac{\displaystyle\sum_{T:i\in T} \sum_{k\in T} \left.\frac{\partial u}{\partial x}\right|_k \left.\frac{\partial N_k}{\partial x}\right|_T \int_T N_i dW}{\displaystyle\int_{supp(i)} N_i dW} \tag{3.57}$$

where the nodal value of the first derivative is given by equation (3.56). Thus, to evaluate the nodal value of the second derivative by (3.57), one needs to first compute the nodal values of the first derivatives by (3.56). This two-step process is rather time-consuming and since the recovery procedure is approximate, an alternative method would be to rewrite equation (3.57) in a more compact form as

$$\left.\frac{\partial^2 u}{\partial x^2}\right|_i = \frac{\displaystyle\int_{supp(i)} N_i \frac{\partial^2 u}{\partial x^2} dW}{\displaystyle\int_{supp(i)} N_i dW} \tag{3.58}$$

where the numerator is computed following an integration by parts. The nodal value of the second derivatives would then reduce to

$$\left.\frac{\partial^2 u}{\partial x^2}\right|_i = \frac{-\int\limits_{supp(i)} \frac{\partial u}{\partial x}\frac{\partial N_i}{\partial x} dW + \int\limits_{\Gamma_{supp(i)}} \frac{\partial u}{\partial x} N_i d\Gamma}{\int\limits_{supp(i)} N_i dW} \tag{3.59}$$

where $\Gamma_{supp(i)}$ represents the surface of triangular faces bounding the tetrahedral elements comprising the support of node *i*.

FIGURES

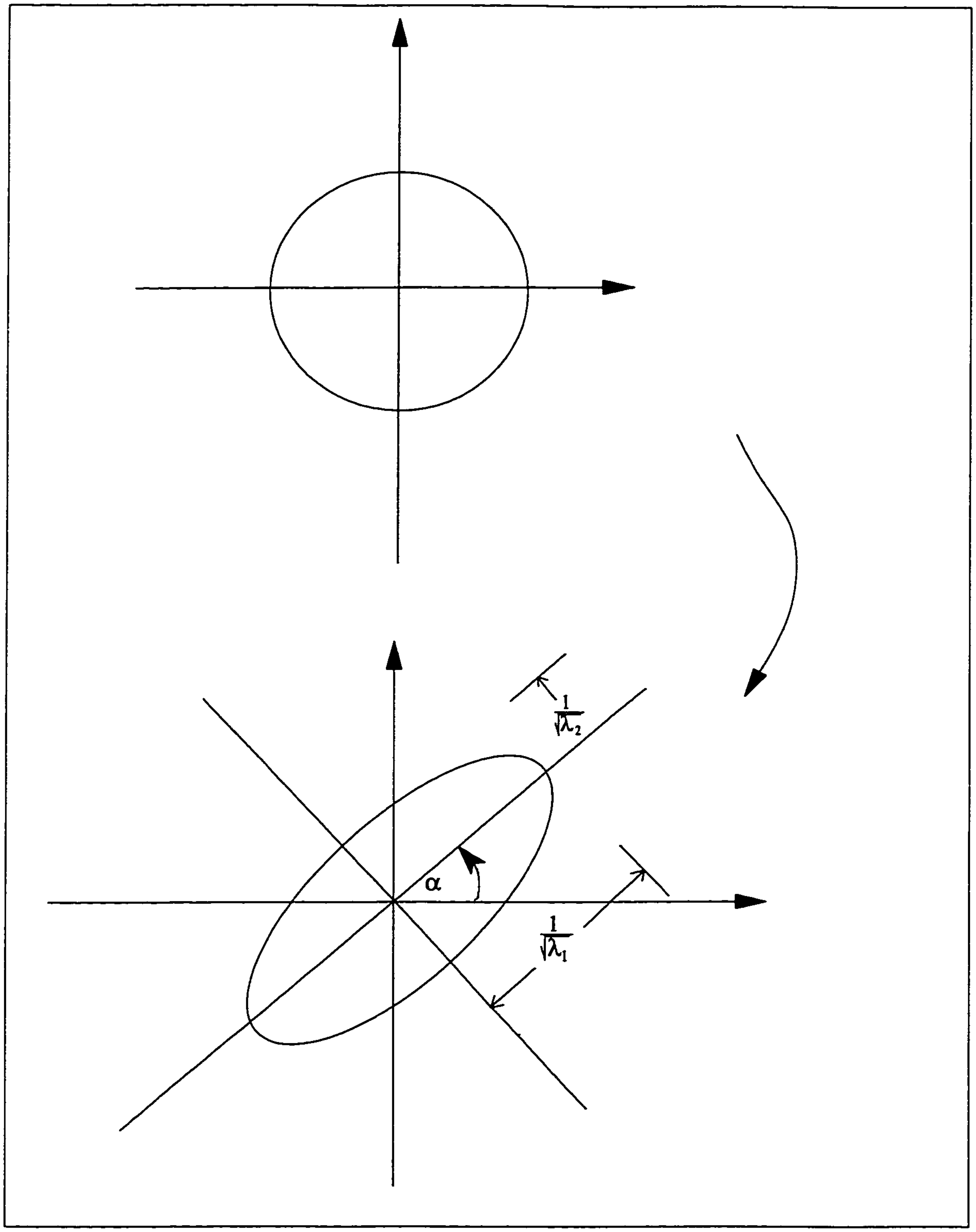


Figure 3.1 Transformation mapping unit circle in metric space to ellipse in physical space

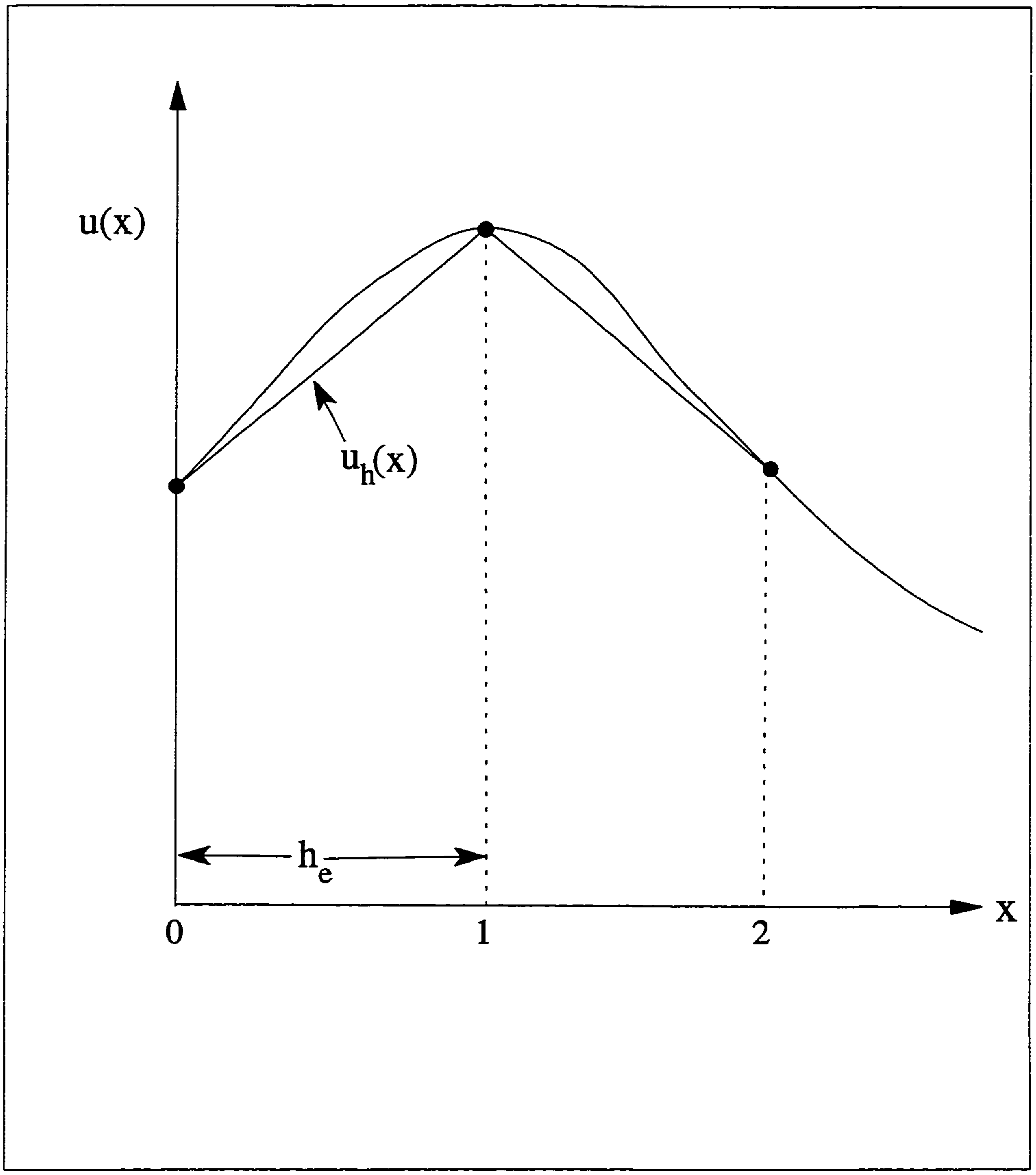


Figure 3.2 Finite element approximation of solution u(x)

4. Adaptive Strategies

4.1 Introduction

The adaptive strategy modifies the grid under the guidance of the error estimate to improve the quality of the numerical solution. Having built an interpolation error estimate to detect changes in the gradients of flow solution variables, that is $(\vec{\mathbf{x}}_J - \vec{\mathbf{x}}_I)\tilde{\mathbf{H}}(\vec{\mathbf{x}})(\vec{\mathbf{x}}_J - \vec{\mathbf{x}}_I)^T =$ constant C', one is now in a position to couple this error estimate to an adaptive strategy. One of the objectives in this research is to directionally orient the mesh to more accurately and cost-effectively capture flow anisotropies such as boundary layers, shocks, vortices and contact discontinuities. Thus, the use of an appropriate adaptive strategy is crucial for achieving the desired grid, namely, a directionally adapted mesh.

One contribution of this thesis has been the development of a 3-D adaptive mesh movement scheme applicable on both structured and unstructured grids [66-68]. This proposed adaptive strategy is thought to be the first 3-D implementation of an improved spring analogy-based algorithm originally applied on 2-D quadrilateral meshes [69].

An adaptive nodal displacement strategy was the first logical foray into the field of 3-D grid adaptation due to its ease of portability into existing codes. However, during the process of validating the mesh movement method on selected test cases, one realized that there are limitations to applying an adaptive mesh movement method on the complex 3-D flow problems and geometries encountered in industry. Using isotropic tetrahedral grid generation or isotropic refinement, for that matter, to properly capture the intricacies of the geometry and the complicated flow phenomena often associated with these cases would lead to an unrealistically large grid size. An anisotropic nodal displacement method, as presented in this thesis, is dependent on the size of the initial non-adapted mesh. If the initial mesh is too coarse, nodes are moved away from one flow region to another resulting in poor resolution of certain flow features. It would be impractical to generate very large sized grids merely to show that mesh movement could eventually resolve the flow physics of such problems. Furthermore, and contrary to 2-D experience [13], as the complexity of the geometry increases, the mesh quality constraints imposed on the node movement strategy limit the possible range of grid point displacement. It is clear that in the solution of complex 3-D geometries involving sharp directional flow features, an anisotropic mesh adaptation procedure which incorporates *all* strategies such as refinement/coarsening, nodal displacement and edge swapping, would be a more practical approach and one which would lead to a reasonable final grid size. Such a comprehensive mesh adaptation procedure , entitled "MOM-3D" (Mesh Optimization Method), has been recently developed at Concordia University's CFD Laboratory in collaboration with ICEM-CFD Engineering and under the sponsorship of Pratt & Whitney Canada [68]. Its spirit and mesh movement method is, however, derived from this thesis.

Sections 4.2-4.6 and 4.8 deals exclusively with the mesh movement adaptive strategy. Section 4.7 devotes a brief discussion to the combined strategies of mesh refinement/coarsening, mesh movement and edge swapping, which represents the ongoing and future work of the CFD Laboratory.

4.2 Nodal Movement Algorithm

The proposed mesh movement scheme progressively redistributes a fixed number of nodes so as to equidistribute the interpolation error over the edges of the elements. The mesh may be viewed as a network of springs whose stiffness constants represent the edge-based error estimate (fig. 4.1). Accordingly, the position of the nodal points may be interpreted as the solution of an energy minimization problem, yielding for each node I,

$$\frac{\partial P_I}{\partial \bar{\mathbf{x}}_I} = \frac{\partial \sum_J (\bar{\mathbf{x}}_I - \bar{\mathbf{x}}_J)^2 K_{IJ}}{\partial \bar{\mathbf{x}}_I} = 0 \tag{4.1}$$

where P_I denotes the potential energy of the springs sharing the node I, and K_{IJ} represents the stiffness constant for the edge between nodes I and J. The stiffness constant for a particular edge is defined as

$$K_{IJ} = \frac{d_M(\bar{\mathbf{x}}_I, \bar{\mathbf{x}}_J)}{d_E(\bar{\mathbf{x}}_I, \bar{\mathbf{x}}_J)} \tag{4.2}$$

where $d_M(\bar{\mathbf{x}}_I, \bar{\mathbf{x}}_J)$ represents the length of the edge $[\bar{\mathbf{x}}_I, \bar{\mathbf{x}}_J]$ in the Riemannian metric defined by equation (3.41), reproduced again below,

$$d_M(\bar{\mathbf{x}}_I, \bar{\mathbf{x}}_J) = \int_0^1 \sqrt{\bar{\mathbf{s}}(t)'^T \tilde{\mathbf{H}}(\bar{\mathbf{s}}(t))\ \bar{\mathbf{s}}(t)'}\, dt \tag{3.41}$$

and $d_E(\bar{\mathbf{x}}_I, \bar{\mathbf{x}}_J)$ is the length of the same edge in the Euclidean metric, that is,

$$d_E(\bar{\mathbf{x}}_I, \bar{\mathbf{x}}_J) = \sqrt{(\bar{\mathbf{x}}_J - \bar{\mathbf{x}}_I)^T (\bar{\mathbf{x}}_J - \bar{\mathbf{x}}_I)} \tag{4.3}$$

By lagging $\vec{\mathbf{x}}_J$ and K_{IJ} at the previous iteration m, equation (4.1) becomes:

$$\sum_J (\vec{\mathbf{x}}_I^{m+1} - \vec{\mathbf{x}}_J^{m})\, K_{IJ}^m = 0 \tag{4.4}$$

resulting in a nonlinear problem since the stiffness constant is a function of coordinates. Upon solving for the change in the position vector, the following expression

$$\Delta\vec{\mathbf{x}}_I = \frac{\sum_J (\vec{\mathbf{x}}_I^{m} - \vec{\mathbf{x}}_J^{m})\, K_{IJ}^m}{\sum_J K_{IJ}^m} \tag{4.5}$$

is obtained and the solution is then updated, node by node, according to the equation

$$\vec{\mathbf{x}}_I^{m+1} = \vec{\mathbf{x}}_I^{m} + \omega\, \Delta\vec{\mathbf{x}}_I \tag{4.6}$$

with ω representing a relaxation factor. The convergence of this scheme can be enhanced by considering a Gauss-Seidel iterative algorithm where the most recently computed values of $\vec{\mathbf{x}}_J$ and K_{IJ} are used in equation (4.5). An adaptive iteration is defined as a loop sweeping all the nodes of the domain by this Gauss-Seidel method. Convergence is deemed to be achieved when the relative metric error over all edges between two consecutive adaptive iterations is below a user-specified tolerance limit.

4.3 Treatment of Boundary Nodes

Adapting a mesh involves node displacement not only in the interior of the solution field but also on the surfaces of the aerodynamic bodies. The mesh adaptation procedure must at all times respect the original CAD surface definition. The nodal displacement algorithm, which is applied to boundary nodes to determine their optimal positions, is the same as that for internal nodes. However, the coordinates of the optimal position may lie off the curve or surface to which the node originally belonged. These boundary nodes are therefore projected back to the nearest corresponding curve or surface.

This is a second and important contribution of this thesis. Realizing the need to maintain the fidelity of the CAD definition of geometric surfaces, an earlier decision was made to associate this development with an established commercial code. An evident choice was ICEM-CFD Engineering's mesh generation package, *ICEM-CFD*, one of the most widely used mesh generation codes in the industry. The adaptive code is thus linked to *ICEM-CFD*'s output interface library, which provides access to the grid geometry and the capability to project back to the nearest curve, surface or family. A family is defined as a set of surfaces, curves, and/or points and, for example, may represent an inlet, exit, wall or symmetry plane. The initial non-adapted grid is generated using *ICEM CFD*'s *Hexa, Tetra* and/or *Prism* mesh generation modules and the input to any of these meshers is a set of surfaces, curves and prescribed points. The curves are necessary since they cause the mesher to follow discontinuities in surfaces. If no curves are specified at a surface boundary, the mesher is free to mesh quadrilaterals or triangles over the surface edge. Similarly, prescribed points are needed to force the mesher to recognize sharp corners in the prescribed curves. The mesher will approximate the surface with quadrilaterals or triangles, the curves with edges, and will place vertices on the prescribed points [70, 71]. When the initial grid is generated, a file is created identifying the curve, surface or family to which a boundary node belongs. This file will be read by an interface routine at the beginning of the adaptive code and the nodal information will be stored in arrays to be used each time a boundary node has to be projected back to its respective curve, surface or family. The identification of prescribed points is important in preserving the geometric configuration of the aerodynamic body during the mesh adaptation procedure since these points should remain fixed and not be subjected to the nodal displacement algorithm.

4.4 Control of Mesh Quality

Before a node can actually be displaced to its new position, tests are performed to check the quality of all elements connected to this node at its new position. Due to the geometric flexibility of the tetrahedron, as compared to the hexahedron or prism, it is not surprising that fewer constraints on elemental degeneracy need to be imposed. For tetrahedral grids only a geometric "shape factor" criterion, defined as

$$\textit{geometric shape factor} = \frac{vol^4}{\left[\sum_{i=1}^{6}(area)^2\right]^3} \tag{4.7}$$

where *vol* refers to the volume of the tetrahedral element and *area* represents the area of each of its six triangular faces, is computed. A geometric shape factor of 1.0 corresponds to an equilateral tetrahedron and since one aims to attain highly anisotropic elements in this work, acceptable shape factor values typically range from 10^{-10} to 10^{-1}. However, for hexahedral and prismatic grids, quality assessments are performed based on the values of the determinant of the Jacobian at Gauss-Quadrature integration points, the shape factors of the three/six component tetrahedra of the prism/hexahedron as well as the minimum and maximum angles of the quadrilateral faces. Typical minimum and maximum angles range from 10°-20° and 160°-170°, respectively, at the final adaptation cycle.

4.5 Interpolation of Riemannian Metric on Tetrahedral Mesh

The desire to accurately update the metric error along the edges has led to the definition of two types of meshes within the adaptive code. For each adaptive cycle, there exists a *background* as well as a *current* mesh. The background grid remains unchanged throughout the adaptive cycle and is identical to the current grid at the beginning of the cycle. The background mesh stores the six entries of the Riemannian metric, $\tilde{\mathbf{H}}(\vec{\mathbf{x}})$, at each node and serves as a reference table from which the metric error in the current grid can be interpolated. In contrast, the current mesh is constantly evolving as its grid points are being redistributed to more optimal positions. As the next adaptive cycle commences at the next Newton iteration, the last current mesh of the previous cycle becomes the background grid for the new cycle.

The interpolation of $\tilde{\mathbf{H}}(\vec{\mathbf{x}})$ from the background grid to points in the current grid is crucial to preserving the accuracy of the error estimate as the adaptation procedure progresses. Given any point C of the current mesh, with coordinates $\vec{\mathbf{x}}_C$, one seeks to find the element *B* of the background grid in which it lies. Since the interpolation operation is called upon many times during an adaptive iteration, it is important that a very fast and robust search algorithm be implemented to locate element *B*.

A search algorithm is based on the evaluation of the shape functions of C with respect to the coordinates of the points belonging to B:

$$\vec{\mathbf{x}}_C = \sum_i N_i \vec{\mathbf{x}}_i \tag{4.8}$$

For tetrahedral meshes a system of three equations in the three unknowns, ξ, η, and ζ, is obtained,

$$\begin{aligned} x_C &= N_1\hat{x}_1 + N_2\hat{x}_2 + N_3\hat{x}_3 + N_4\hat{x}_4 \\ y_C &= N_1\hat{y}_1 + N_2\hat{y}_2 + N_3\hat{y}_3 + N_4\hat{y}_4 \\ z_C &= N_1\hat{z}_1 + N_2\hat{z}_2 + N_3\hat{z}_3 + N_4\hat{z}_4 \end{aligned} \tag{4.9}$$

where

$$\begin{aligned} N_1 &= 1-\xi-\eta-\zeta \\ N_2 &= \xi \\ N_3 &= \eta \\ N_4 &= \zeta \end{aligned} \tag{4.10}$$

The system of equations (4.9) may be simplified to

$$\begin{bmatrix} x_C - \hat{x}_1 \\ y_C - \hat{y}_1 \\ z_C - \hat{z}_1 \end{bmatrix} = [A] \begin{bmatrix} \xi \\ \eta \\ \zeta \end{bmatrix} \qquad \text{where } [A] = \begin{bmatrix} \hat{x}_2 - \hat{x}_1 & \hat{x}_3 - \hat{x}_1 & \hat{x}_4 - \hat{x}_1 \\ \hat{y}_2 - \hat{y}_1 & \hat{y}_3 - \hat{y}_1 & \hat{y}_4 - \hat{y}_1 \\ \hat{z}_2 - \hat{z}_1 & \hat{z}_3 - \hat{z}_1 & \hat{z}_4 - \hat{z}_1 \end{bmatrix} \tag{4.11}$$

and the four tetrahedral shape functions are determined directly from the unknowns,

$$\begin{bmatrix} \xi \\ \eta \\ \zeta \end{bmatrix} = [A]^{-1} \begin{bmatrix} x_C - \hat{x}_1 \\ y_C - \hat{y}_1 \\ z_C - \hat{z}_1 \end{bmatrix} \tag{4.12}$$

The point C lies inside element B if

$$0 \leq N_i \leq 1 \qquad \text{for} \quad i = 1,4 \tag{4.13}$$

Since the background grid consists entirely of tetrahedral elements, this is the only shape function criterion that needs to be applied to see if a current point lies within a background element. However, the current mesh may be composed of all element types such as hexahedra, prisms and tetrahedra. So, at the start of each adaptive cycle, these elements must be subdivided into tetrahedra to generate the accompanying tetrahedral background mesh necessary for data interpolation. A hexahedron is subdivided into six tetrahedra and a prism is split into three tetrahedra. If a current point is found in any of the element's component tetrahedra using the shape function criterion, it is considered to be inside the element.

Given that the numerical integration of the Riemannian metric length of an edge requires that the value of $\mathbf{M}(\bar{\mathbf{x}})$ at each integration point along the edge be interpolated, a fast "neighbor-to-neighbor" search algorithm for data interpolation was implemented [72]. This type of search routine assumes that for any given point in the current mesh to be interpolated, an element of the background grid that is in the vicinity is known. Such a routine requires that a nodal array pointing to the nearest element of the background mesh, as well as an array storing the element-face connectivities, be created.

At the beginning of the adaptive procedure, the nodal array, PNO2EBG[1:NNODE], pointing to the nearest element of the background mesh is initialized using the nodal connectivities of the current mesh. If the current grid contains hexahedra or prisms, these elements must first be subdivided into tetrahedra before this array can be initialized. The neighbor-to-neighbor search algorithm may be summarized as follows:

```
    DO 100   Loop over all edges connected to node being displaced
        DO 200   Loop over all points along edge to be interpolated
                 Obtain a good starting element IEBG to begin search
                 (note:  IEBG = PNO2EBG[end node of edge] )
300              For IEBG, evaluate $N_i$ for $i$=1,4 from equation (4.13)
                     IF equation (4.13) is satisfied THEN
                     Exit
               ELSE
                     Set IEBG to neighboring element*
                     GOTO 300
               ENDIF
200     CONTINUE
100 CONTINUE
```

*The next element in the search path is that adjacent to the face opposite the node with the minimum shape function value.

The neighbor-to-neighbor search algorithm performs well in the interior of the domain but is known to have difficulties on the boundary, especially in situations where the tetrahedral elements are highly stretched. In such cases, the background element, which is closest to the current point C before the search failed, is known and a back-up brute force search routine is called whereby a loop is performed over all its neighboring elements, evaluating their shape functions with respect to $\vec{x}_C$. If C is still not found among these neighbors, the search expands to the adjacent elements of these neighbors.

4.6 Optimization of Mesh Movement Algorithm

In an effort to reduce the percentage of flow solver CPU time consumed by grid adaptation, it is not necessary that all the nodes of the domain be swept at each adaptive iteration. After each iteration, the maximum relative metric error over the edges connected to a particular node is stored in a nodal array. Rather than sweeping over all the nodes of the domain, only those nodes whose stored relative metric error between consecutive adaptive iterations is greater than the user-specified convergence criterion will be subjected to the mesh movement process. As the number of adaptive iterations increase, more nodes will attain this convergence limit and, consequently, a smaller percentage of the nodes in the domain will undergo the displacement algorithm. The percentage of solver CPU time taken up by grid adaptation has thus been significantly reduced with the implementation of this convergence criterion (see Chapter Six, "Numerical Results").

4.7 Combined Mesh Refinement/Coarsening, Mesh Movement and Edge-Swapping Strategies

A limitation of any mesh movement strategy is its dependence on the initial grid point distribution due to the fixed nodal connectivity. The addition of mesh refinement and/or coarsening, and edge-swapping strategies increases the flexibility of the grid adaptation procedure by permitting the nodal connectivity to be modified.

When mesh movement is the only adaptive strategy considered, it may be considered a minimal error mesh problem [37] in which the number of vertices in the domain are fixed and one seeks a mesh which minimizes the maximum interpolation error over the edges. In such a situation, the objective is to equidistribute the interpolation error over the edges of the domain. The inclusion of mesh refinement and coarsening as well as edge swapping strategies implies that the interpolation error can no longer be simply equidistributed. It becomes a maximum efficiency mesh problem [37] in which the maximum interpolation error is specified to meet a target edge length with respect to the chosen metric.

An edge is refined or split when its metric edge length is too long or above a specified threshold. An edge is coarsened or collapsed when its metric edge length is too short or below a set threshold. Edge swapping is a strategy in which the edges are reconnected in a manner to more equally distribute the interpolation error along the edges of the associated tetrahedra. To choose among the possible configurations of tetrahedra resulting from an edge being swapped, a new criterion has been defined to measure the quality of the tetrahedral element. This criterion is a 3-D extension of the mesh quality previously described for triangular elements in [13]. The mesh quality is a function of the lengths of the tetrahedral edges in the chosen metric and measures the degree to which the error estimate has been equidistributed over the edges. Its value ranges from a minimum of zero, which corresponds to a completely flattened tetrahedron, to a maximum of one, which corresponds to an equilateral tetrahedron with respect to the chosen metric. An edge will be swapped if the mesh quality of the resulting configuration of tetrahedra has improved, where the quality of the configuration is defined by the tetrahedral element with the poorest quality.

Selecting an appropriate sequence of the adaptive strategies is a key step in obtaining an efficient adaptive solution. The goal is to decrease the interpolation error to the desired level in an optimal manner. This entails the addition of degrees of freedom in regions of the flow field where errors are high and deletion of degrees of freedom where errors are low. However, over-refinement would lead to a mesh with more degrees of freedom than necessary, which is expensive, and excessive coarsening would not achieve the desired error tolerance, necessitating further adaptive meshes to drive down the error.

In practice one begins with a fairly coarse uniform mesh and a complete adaptive solution typically consists of a series of three or four adaptive meshes. This tends to offset the described competing effects. An attempt to achieve the error tolerance in a single adaptive step would result in over-refinement, since the error on the initial mesh tends to be spread out over the larger elements. However, to minimize this tendency towards over-refinement, one may initialize the adaptive procedure by commencing with a few sweeps of the mesh movement algorithm to more equally distribute the error and introduce some degree of desired anisotropy. A conservative strategy which coarsens too much would likewise be inefficient, requiring too many meshes for sufficient resolution. Hence, this sequencing of adaptive strategies may be viewed as a highly nonlinear optimization problem.

The main difficulty in choosing an appropriate sequence of adaptive strategies lies in the optimization of the relative sizes of the elements. If regions with high errors are over-refined relative to the refinement and/or coarsening of other areas, then these over-refined elements will possess comparatively lower errors in the solution of the adapted mesh. These regions of lower errors will then be coarsened in the subsequent adaptive mesh, in conjunction with the over-refinement of the other areas, which now possess comparatively larger errors. The oscillation between refinement and coarsening would continue on subsequent adaptive meshes and effectively impedes the attainment of an optimal solution. This emphasizes the importance of a sound sequence of adaptive strategies.

The mesh modification algorithm begins with a selected number of iterations of the mesh movement and edge swapping strategies. This equidistributes the error over the edges and minimizes the tendency of the initially coarse mesh to over-refine. The next step is to split those edges with the largest metric edge lengths and collapse those with the smallest so as to reduce the range of metric lengths and bring its average value closer to the target error level. Once all edges have been considered for splitting or collapsing, mesh movement and edge swapping operations are repeated. After the entire sequence has been executed a specified number of times, one obtains an adapted grid.

4.8 Summary of Grid Adaptation Procedure

The incoming grid from the flow solver is termed the current mesh and may contain tetrahedral, hexahedral or prismatic elements. At the beginning of each adaptive cycle, an equivalent background grid consisting of purely tetrahedral elements must be created for interpolation purposes. Using the incoming flow solution on the background grid, nodal values of the Hessian of the flow variable are recovered and then modified to the corresponding Riemannian metric values. With these nodal values, the metric lengths of all edges of the incoming grid are numerically integrated and these will be used in computing the edge-based error estimate, defined to be the spring constant for that edge.

The grid adaptation procedure may be summarized as follows:

```
      DO 10  iter = 1, max. no. of adaptive iterations
        DO 20  inod = 1, NNODE
          IF relative metric error is > CONVCRIT THEN
            DO 30 iedge = 1,NEDGE
              Compute spring constant $K_{IJ}$
30          CONTINUE
            Compute new position of inod from equation (4.6)
            IF boundary node THEN
              Project back to corresponding curve or surface
              Perform tests on quality of elements connected to inod
            ELSE
              Perform tests on quality of elements connected to inod
            ENDIF
            Move inod to its new coordinates
            Update the metric length of connecting edges of inod
          ENDIF
20      CONTINUE
        Update nodal array for relative metric error
        IF relative metric error for all nodes < CONVCRIT exit loop 10
10    CONTINUE
```

NEDGE represents the number of edges sharing the node *inod* and CONVCRIT is the user-specified convergence criterion for the relative metric error.

FIGURES

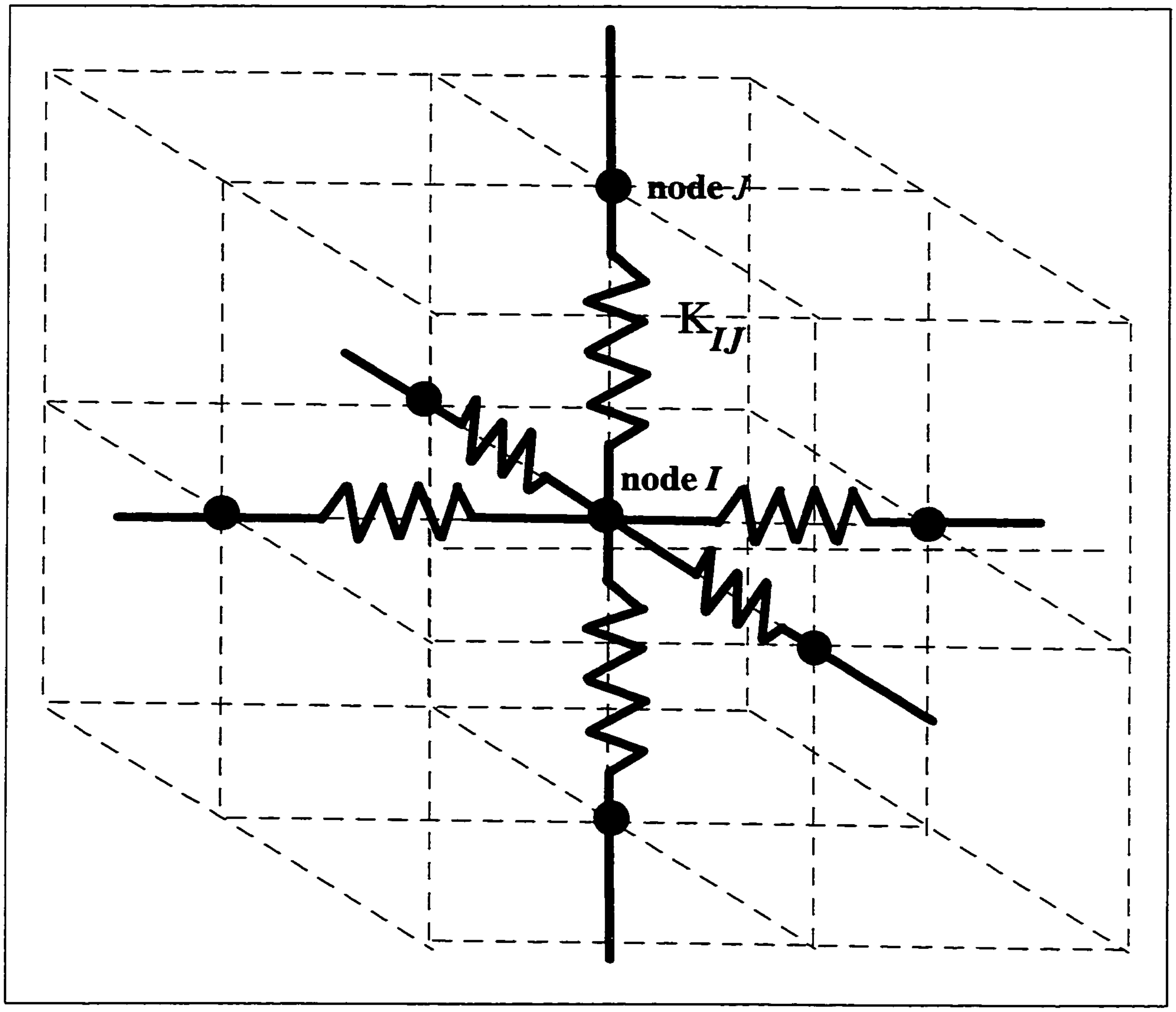


Figure 4.1 Spring analogy for mesh movement strategy

5. Solution Procedure

5.1 Introduction

This chapter describes the flow solver algorithm as well as the coupling of the solver with the grid adaptation method. The numerical scheme used to iterate for spatial nonlinearity will be discussed. The strategy of a centered scheme with artificial dissipation has been implemented for stabilization purposes. The grid adaptation algorithm has been embedded in the flow solver procedure and the relationship between the adaptive parameters and artificial dissipation scheme will be discussed.

5.2 Flow Solver Algorithm

A flowchart outlining the solution procedure of *NS3D* is given in fig. 5.1. As detailed in Chapter Two, the nonlinear governing equations are linearized by the Newton method and spatially discretized by the Galerkin finite element method. The steady state solution of the equations is obtained by an implicit time-marching approach based on the first order Gear scheme. The Newton linearization procedure results in a set of linear equations for ρu, ρv, $\rho w, p$ and T_0. The continuity and momentum equations (2.2-2.3) are solved in a coupled manner for the mass flux components, ρu, ρv, ρw, and pressure p. To reduce overall memory requirements, the energy equation (2.4) is solved for the total temperature in a segregated manner. The static temperature field is updated using the total temperature, mass flux components and pressure and density is updated using the equation of state (2.5).

A strategy of a centered scheme with artificial dissipation is implemented to stabilize the convergence of the system of the coupled continuity-momentum equations at high Reynolds numbers. The coefficients of these artificial dissipation terms are represented symbolically as λ and μ_{art} in fig. 5.1. One weakness of the Newton algorithm is that the initial approximation must be sufficiently close to the final solution in order to obtain convergence. To partially overcome this weakness, the solution procedure involves a series of steps in which the amount of artificial dissipation in the numerical scheme is progressively unloaded or decreased. These unloading steps result in a successive series of solutions which are obtained at decreasing levels of artificial dissipation and serve as progressively improved approximations to initiate the next Newton loop. The unloading is typically carried out in five steps, yielding four intermediate solutions as well as the desired final solution which is obtained using the smallest amounts of artificial dissipation possible. The overall residual is defined in the L_2-norm as $\left\|\overline{res}\right\|_2^{n+1} = \sqrt{\left(Res_{\rho\vec{V}}^{n+1}\right)^2 + \left(Res_p^{n+1}\right)^2}$ and *toler* refers to the orders of magnitude by which this overall residual must decrease to attain convergence. Although it is not necessary to drop the overall residual of intermediate solutions by more than two orders of magnitude, the residual of the final solution must satisfy a stricter convergence criterion.

Within a Newton iteration the coupled continuity and momentum equations as well as the energy equation are solved using iterative solvers such as GMRES (Generalized Minimum Residual Method) and PCGS (Preconditioned Conjugate Gradient Squared Method). Since an iterative solver is used at each Newton iteration to solve the ill-conditioned linear system coming from the continuity-momentum equations, it has also been found highly beneficial for the convergence of the iterative scheme to introduce additional dissipation in the coefficient matrix itself. This dissipation, denoted symbolically as μ_{art}^{lhs} and λ^{lhs}, is similar to that appearing in the residuals of the system of equations. Typically, the values specified for μ_{art}^{lhs} and λ^{lhs} in the iterative scheme are greater than those of the residuals of the system of equations. This is equivalent to freezing the iterative matrix at a lower Reynolds number than that at which the residuals are calculated.

NS3D solves the system of steady, compressible flow equations by a time-marching procedure. The discretized time-dependent terms increase the diagonal dominance of the coefficient matrix **K** and hence improve its condition number. The size of the time step is dynamically chosen, as determined by the flow problem, and this helps to ensure the convergence of the linear system.

5.3 Coupling of Flow Solver with Grid Adaptation

The flowchart in fig. 5.2 provides an overview of the coupling of the flow solver to the grid adaptation scheme. Starting with an initial non-adapted grid, the adaptive scheme is called at each unloading step when the overall residual has decreased by the order of magnitude specified by *toleradp*. After the grid adaptation algorithm has attained convergence, the output is an adapted grid and the accompanying interpolated solution. This interpolated solution is used to begin the next Newton iteration. When the residual further drops by another specified order of magnitude, *toler*, the Newton loop is considered to be converged. Within each unloading step, an improved solution is computed on the resulting adapted grid. By coupling the grid adaptation with the flow solver, one obtains not only a less dissipative solution but a more optimal grid for that particular solution. With each unloading step, there is not only a progressive decrease in the amounts of artificial dissipation but also a relaxation of the adaptive parameters controlling the quality of the elements. For hexahedral elements the range spanning the minimum and maximum angles of the quadrilateral faces is progressively increased with each unloading step. For tetrahedral elements, the shape factor is decreased with each unloading step, thus allowing a greater degree of anisotropy to be introduced into the adapted mesh. The loosening of the constraints on the elemental quality leads to a wider range of nodal displacement as the unloading steps proceed.

The above approach assumes that no solution is provided on the initial grid. However, if there existed an accompanying converged solution, one can use the incoming solution to compute the error estimate and adapt the grid. One may then compute the finite element solution on the adapted grid using the same or lower levels of artificial dissipation. It will be shown that with grid adaptation smaller amounts of artificial dissipation are required for the same level of convergence.

FIGURES

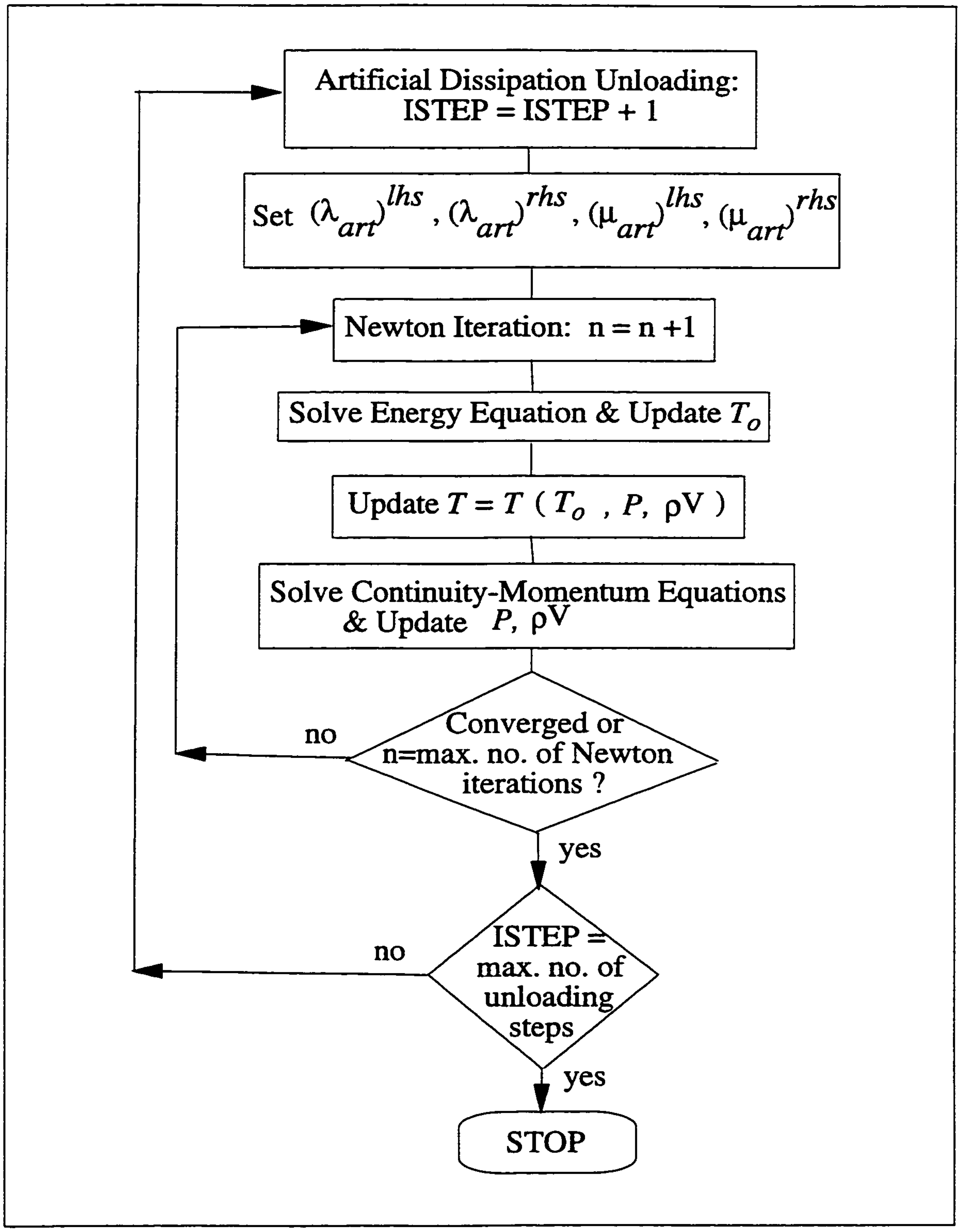

Artificial Dissipation Unloading: ISTEP = ISTEP + 1
Set $(\lambda_{art})^{lhs}$, $(\lambda_{art})^{rhs}$, $(\mu_{art})^{lhs}$, $(\mu_{art})^{rhs}$
Newton Iteration: n = n +1
Solve Energy Equation & Update T_o
Update $T = T(T_o, P, \rho V)$
Solve Continuity-Momentum Equations & Update P, ρV
Converged or n=max. no. of Newton iterations ?
no
yes
ISTEP = max. no. of unloading steps
no
yes
STOP

Figure 5.1 Flow solver algorithm

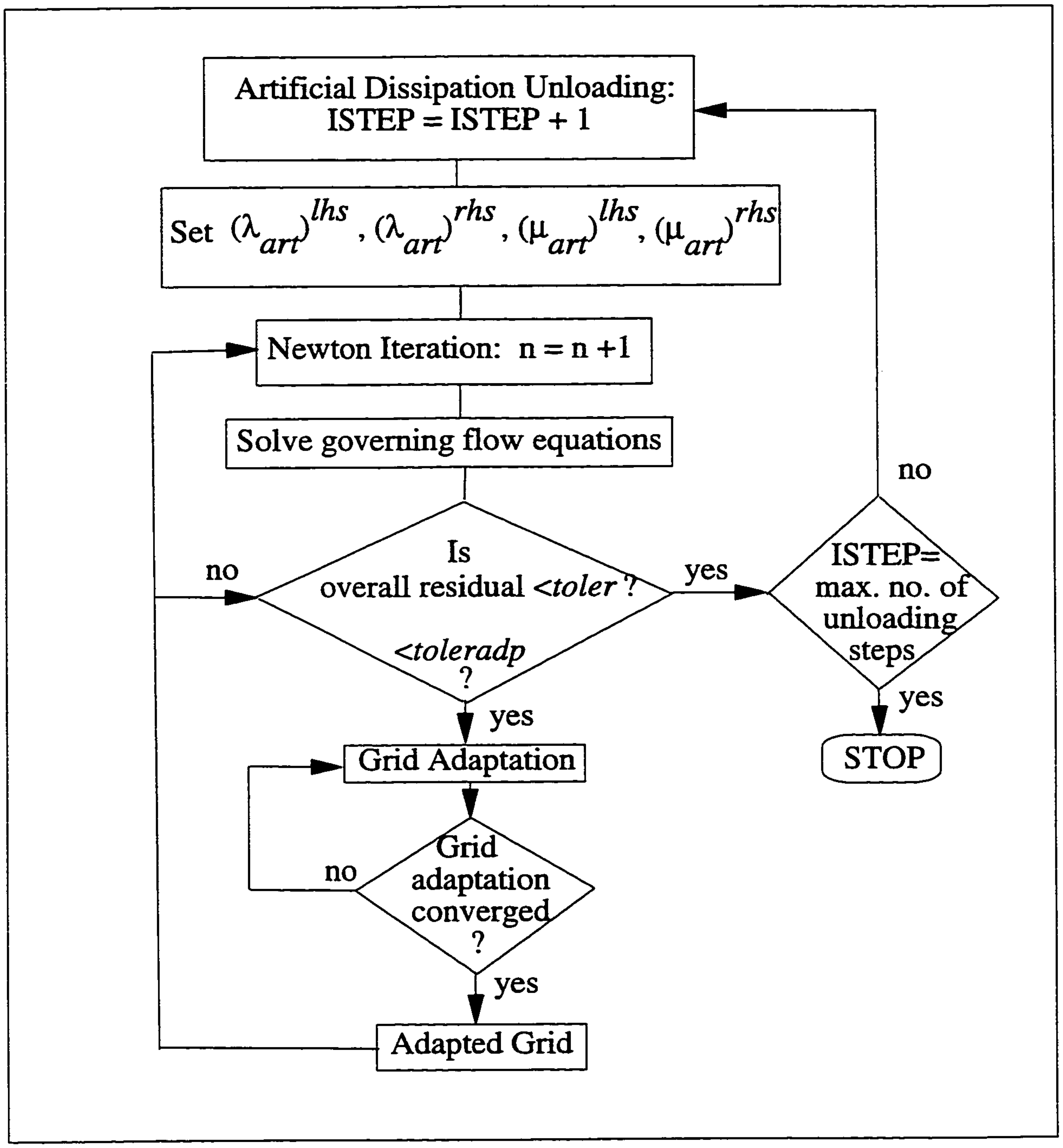

Artificial Dissipation Unloading: ISTEP = ISTEP + 1
Set $(\lambda_{art})^{lhs}$, $(\lambda_{art})^{rhs}$, $(\mu_{art})^{lhs}$, $(\mu_{art})^{rhs}$
Newton Iteration: n = n +1
Solve governing flow equations
Is overall residual <toler ? <toleradp ?
no
yes
ISTEP= max. no. of unloading steps
no
yes
STOP
Grid Adaptation
Grid adaptation converged ?
no
yes
Adapted Grid

Figure 5.2 Algorithm of flow solver coupled to grid adaptation procedure

6. Numerical Results

6.1 Summary of Test Cases

Numerical results have been obtained on both structured and unstructured grids. These results span a range of flow regimes and types, including inviscid and viscous flows, as well as internal and external aerodynamics.

Sections 6.2-6.3 present analytical test cases on tetrahedral and hexahedral meshes which demonstrate the capability of the mesh movement strategy to equidistribute the interpolation error of a known function over the edges of such grids.

Section 6.4 describes the numerical predictions of grid adaptation on the second stage stator of the UTRC (United Technologies Research Center) Large Scale Rotating Rig [73]. It should be remarked that the degree of mesh movement was severely limited in this test case since no nodal displacement was allowed on boundary curves and surfaces.

Sections 6.5-6.6 examine transonic flow over a 10% circular arc which is a 3-D extension of an oft used 2D case [74]. This case was chosen to test the capability of the mesh adaptation procedure to improve the resolution of shocks on both unstructured and structured meshes.

Section 6.7 considers laminar viscous flow in a square duct with a 90° bend [75-77]. Six different grid sizes for this geometry were investigated and this test case was used to study the solution error of non-adapted and adapted grids as a function of mesh size. One sought to determine the size of the non-adapted grid that would be equivalent to that of a given adapted grid for the same level of solution error.

Section 6.8 presents the results of adaptive mesh movement on laminar non-reacting flow in a hydrogen fuel can combustor [78]. This case shows that isotropic unstructured mesh generation may not necessarily be the cure all for resolving the flow features of a problem. It also convincingly shows that a converged solution may only be obtained after applying grid adaptation.

Section 6.9 describes inviscid flow over a NASA wing-pylon-nacelle configuration [79, 80]. It is the only case in this thesis which applies the full range of adaptive strategies, that is, grid refinement/coarsening, nodal movement and edge swapping. The reason for this is that the mesh movement procedure by itself, applied successfully in all previous test cases, was found not to be sufficient to tackle the complex flow phenomena and geometry of this configuration. The main difficulty is the exceedingly thin nacelle leading edge which creates a rapid flow acceleration around the outer nacelle lip. Properly capturing this acceleration with isotropic tetrahedral grid generation or isotropic refinement would require an unrealistically large grid size. An anisotropic nodal displacement method, as presented in this work, is dependent on the size of the initial non-adapted mesh. If the initial mesh is too coarse, nodes are moved away from one flow region to another resulting in poor resolution of certain flow features. It would be impractical to generate a very large sized grid merely to show that mesh movement could eventually resolve the flow physics of this problem. It was clear that due to the directional nature of the nacelle lip acceleration (small gradients in the circumferential direction), an anisotropic mesh adaptation procedure which incorporates all strategies such as refinement/coarsening, nodal displacement and edge swapping, would be a more practical approach and one which would lead to a drastic reduction in the grid size.

The wing-pylon-nacelle test case accentuates the need to apply the full range of adaptive strategies to complicated 3-D flow problems and geometries. The development of such a comprehensive mesh adaptation method thus represents the ongoing and future work of Concordia University's CFD Laboratory.

6.2 Analytical Test Case on Tetrahedral Grid

To investigate the effectiveness of the adaptive procedure on unstructured grids, an exact test case is first chosen to demonstrate the capability of the mesh movement strategy to equidistribute the interpolation error of a known function over the edges. An analytical function *f*, possessing strong gradients,

$$f(x,y,z) = arctan\left[1000\left(x^4y^4z^4 - \frac{1}{256}\right)\right] \tag{6.1}$$

has been defined over a [0,1]x[0,1]x[0,2] domain which comprises 2739 nodes and 13,950 tetrahedral elements.

The initial and adapted meshes, along with the corresponding isolines of *f* are shown in figures 6.1 and 6.2, respectively. The adapted mesh greatly improves the resolution of the function in regions where large values of the second derivatives of *f* occur. The jagged nature of the contour lines on the initial grid is almost entirely eliminated on the adapted grid due to a more optimal placement of the nodes and alignment of the stretched tetrahedral elements. The adapted mesh shown in fig. 6.2 was obtained after 250 iterations of the mesh movement algorithm using a relaxation factor of 1.0.

Figure 6.3 shows the convergence history of the mesh movement algorithm. The average nodal displacement dropped by more than three orders of magnitude over 175 iterations and, after this point, no further decrease in the average displacement was observed.

Since equation (6.1) is an analytical function, the second derivatives can be calculated exactly and used directly in the computation of the edge metric error. A distribution of the number of edges versus the exact error over these edges for both the initial and adapted grids is presented in figure 6.4. In the ideal adapted case, all the edges would possess the same error but, in practice, a near-Gaussian distribution is attained whereby the maximum edge error has been reduced three-fold.

6.3 Analytical Test Case on Hexahedral Grid

Isolines of the same analytical function (6.1) were plotted on initial and adapted hexahedral grids comprising 2541 (11x11x21) nodes (fig. 6.5-6.6). The adapted grid shown was obtained after 250 iterations of the mesh movement scheme, using a relaxation factor of 1.0.

In fig. 6.7 the convergence history of the nodal displacement algorithm reveals that the average displacement decreased by four orders of magnitude over 700 iterations, beyond which point no important changes in the positions of the vertices are detected.

A distribution of the number of edges versus the exact error over these edges is presented in fig. 6.8. In the adapted case, the distribution has shifted towards a near Gaussian one in which the maximum error over an edge has been reduced by four-fold.

Questions regarding the "quality" of the error estimate often arise in grid adaptation investigations, i.e. how does it represent the "real error", were it known. To address this issue, the number of edges versus the exact and estimated error were plotted for both initial and adapted hexahedral grids (fig. 6.9-6.10). The exact error over the edges was computed using the exact Hessian matrix of the analytical function whereas the estimated error was determined using second derivatives recovered through the weak Galerkin formulation (Section 3.5.3). The estimated error appears to quite closely follow the exact error for both the initial and adapted cases. However, a more detailed analysis of the quality of the error estimate is presented in fig. 6.11(a-c) in which the exact and estimated error, as well as their absolute difference, are plotted versus each of the corresponding 7040 edges in the hexahedral grid. Again, the estimated error displays a very similar trend to that of the exact error over all the 7040 edges.

6.4 Second Stage Stator of UTRC Large Scale Rotating Rig

The second stage stator of the UTRC (United Technologies Research Center) Large Scale Rotating Rig has been chosen to validate the adaptive methodology for turbomachinery flows. Much detailed experimental data [73] are available for the second stage of this large-scale 2.5 stage compressor (2-stage axial-flow compressor with inlet guide vanes). It is considered a difficult test case because of the low Mach number in the flow field, with typical relative Mach numbers less than 0.2, and the presence of corner stalls near the endwalls. Mesh adaptation is expected to alleviate some of the solver's problems for this particular test case.

As shown in fig. 6.12, the second stator possesses 44 blades with its inlet located at station 4 and its exit at station 5. Station 4 is found at 19% of the stator axial chord upstream of the stator leading edge. Station 5 is located at 18% of the stator axial chord downstream of the stator trailing edge. The stator is solved with imposed experimental inlet and exit boundary conditions. An H-mesh was used with 14 points upstream of the blade, 57 points on the blade and 13 points downstream of the blade in the flow direction (fig. 6.13a-6.13c). Twenty-five and 21 planes were used in the hub-to-shroud and blade-to-blade directions, respectively.

Three cycles of adaptation were required to produce the adapted grid shown in fig. 6-14a-6.14c. It should be remarked that in this first validation test case of the adaptive method, the degree of nodal movement was severely limited by the fact that no mesh movement was allowed on boundary curves and surfaces. These curves and surfaces were generated by a non-CAD based in-house mesh generation code. In fact, since this was a turbulent case, all eight nodes of the hexahedral wall layer were restricted from moving so as to preserve the original $y+$ values. Despite these severe constraints, the adapted numerical results showed noticeable improvement over the non-adapted predictions and responded convincingly toward the experimental data (fig. 6.15-6.20).

Figures 6.15a-6.15d reveal comparisons between experimental and computed static pressure distributions for suction and pressure surfaces at 3.0%, 45.1%, 73.4% and 95.5% span. In general, the adapted numerical results agree better with the experimental data than non-adapted results and this holds true from hub-to-tip. In particular, separation in the flow field in the hub region (3.0% span) on the suction surface is accurately captured. At 3.0% near the leading edge the suction surface pressure coefficient is better predicted after adaptation. At 45.1% and 73.4%, the adapted results are closer to experimental data on the suction surface. At 95.5%, the static pressure coefficient is over predicted on suction surface but this is corrected to a certain degree by adaptation. Separation is over predicted at the shroud.

Figure 6.16 depicts the total pressure coefficient at the inlet versus percentage span. Total pressure is a computed quantity since mass and momentum fluxes are imposed. The numerical results for both adapted and non-adapted cases reveal that the total pressure profile of the experimental data is respected but the adapted solution more closely matches the experimental data.

The spanwise distribution of the total pressure coefficient at the exit is shown in fig. 6.17. The adapted results represent a significant improvement over the non-adapted predictions, particularly from the hub to 25% span region, and convincingly matches the experimental data.

Figure 6.18 compares the radial distributions of numerical and experimental values of exit flow angle. The flow angles, after adaptation, are well predicted between 10-90% span. The adaptive procedure improves the solution in the regions of 10-40% and 70-90% span. However, between the hub to 10% span and 90% span to shroud, the observed discrepancy may be attributed to the fact that the nodes associated with wall elements are not allowed to move.

The spanwise distribution of loss, computed as the difference between total pressure coefficient at inlet and exit, is shown in figure 6.19. The prediction of loss, particularly in the vicinity of the hub and shroud, is much improved with grid adaptation.

A plot of blockage versus percentage span is given in fig. 6.20. The blockage factor distribution represents the departure of the actual flow field from axisymmetry assumed in the through-flow analysis [81]. The blockage relates the results computed in the through-flow analysis, which represent circumferentially mass-averaged quantities, to the mass flow, which is related to the area-averaged axial velocity. Details regarding the calculation of the blockage factor is described in [81]. The adapted blockage distribution represents a concrete improvement over the non-adapted results.

Mach number was the scalar variable chosen to build the driving edge-based error estimate in the adaptation case. In each of the three adaptive cycles, a given level of artificial dissipation was specified and the L_2-norm of the solution residual was required to decrease by three orders of magnitude before the nodes were displaced 200 times. This number of iterations may seen rather small for a 3-D case but, since the displacement algorithm is applied only to the volume nodes, it was observed that any further increase in the number of sweeps over the nodes would not result in any noticeable change of the nodal positions. As indicated in the convergence history of the mesh movement scheme (fig. 6.21), the greatest change in the grid point positions occurred in the first adaptive cycle where the average displacement drops from $3x10^{-3}$ to $5x10^{-5}$ over the first 200 iterations. The leveling of the displacement curve in each cycle may be attributed to the low value specified for the allowable maximum edge length. A low limit was necessary to ensure that the resulting hexahedral elements in the adapted grid were not highly skewed. The final adapted solution was obtained using 20% less artificial dissipation than in the non-adapted case. A relaxation factor of 0.9 was employed in the movement scheme in all three cycles. Figure 6.22 provides the error distribution over the edges for the non-adapted and adapted grids. It is clear that the error is more equidistributed over the edges of the adapted grid.

6.5 Transonic Flow over 10% Circular Arc on Tetrahedral Grid

An interesting feature of an adaptive methodology is its ability to capture shocks with high accuracy. To prove this point, inviscid transonic flow in a channel with a 10% circular arc is considered on an unstructured grid comprising 8,885 nodes and 45,873 tetrahedral elements. The length of the channel is 3, its height 1 and its width 0.5. This case is a 3-D simulation of a 2-D flow [74].

The initial and adapted grids, with their corresponding Mach contours for an inlet Mach number of 0.675, are displayed in fig. 6.23-6.24. The observed oscillations in the contour lines on the non-adapted grid disappear in the adapted case as the stretched tetrahedral elements become properly aligned with the shock. An error estimate based on Mach number was used to drive the mesh movement algorithm. The adapted mesh reveals that more nodes have moved to the flow field regions with strong changes in gradients of Mach number such as the shock and leading and trailing edges. Such a mesh leads to a much sharper shock (fig. 6.25(a-b)) and a better resolution of the flow variables in the regions of leading and trailing edges.

The adapted grid shown in fig. 6.24 was created after six cycles of adaptation. A converged solution on the non-adapted grid was used to initiate the adaptive process. The solution procedure included three steps in which the artificial dissipation was progressively lowered and, at each given level of dissipation, two grid adaptations were performed. The flow residual was required to decrease two orders of magnitude between each adaptation at a given level of artificial dissipation and three orders of magnitude between unloading steps of artificial dissipation. After the sixth grid adaptation, the flow residual required a drop of five orders of magnitude for the solution to be considered converged. It can be observed that as the adaptive cycles progressed and the ensuing adapted grid became more stretched, the number of solver iterations increased. Over the six adaptive cycles, the amount of artificial dissipation applied decreased by 33%. Each adaptation procedure was considered to be converged when the relative metric error was below the specified limit of 0.25%. A relaxation factor of 0.9 was used in the mesh movement scheme. The convergence histories of the flow solver and adaptation procedures are shown in fig. 6.26-6.27.

In an effort to optimize the mesh movement algorithm in terms of CPU time, at each adaptive iteration only those nodes that did not meet the relative metric error convergence criterion were subjected to the displacement scheme. As illustrated in fig. 6.28, at the beginning of each of the six adaptive cycles, 100% of the nodes in the domain are swept and undergo the displacement algorithm. However, as the iterations progress, an increasing number of nodes have attained the relative metric error limit and are no longer included in the sweep of the nodes. This pattern becomes more pronounced with each subsequent adaptive cycle and, towards the end of the final adaptive cycle, it is found that the adaptive algorithm is executed for only 10% of the total nodes. The percentage of total CPU time (solution and adaptation) taken by the adaptive procedure, consisting of six adaptive cycles, progressively decreased as the adaptive cycles increased. The first and second adaptive cycles occupied 36% of the total CPU time, the third and fourth cycles took up 14% and the fifth and sixth cycles used only 5% of the total CPU time.

Fig. 6.29 compares the error distribution for the non-adapted and adapted grids. The adaptive process has resulted in the error over the edges becoming more equidistributed as indicated by a shift towards a more Gaussian distribution in the adapted case.

6.6 Transonic Flow over 10% Circular Arc on Hexahedral Grid

The previous test case was repeated on an 8,840 noded (65x17x8) hexahedral grid to demonstrate the independence of the mesh adaptation procedure of element type. The initial and adapted grids are displayed in fig. 6.30-6.31, along with their corresponding Mach number contours. The adapted surface mach distribution on the lower wall of the channel reveals a much sharper shock structure as well as improved resolution at the leading and trailing edges of the arc (fig. 6.32). The flow solver and adaptation procedure convergence histories are presented in fig. 6.33-6.34. The solution on the final adapted grid was obtained with 30% less artificial dissipation. A relaxation factor of 0.9 was applied in all six nodal displacement cycles and adaptive procedure was deemed converged when the relative metric error fell below 0.135%. Figure 6.35 shows that the effort to optimize the mesh adaptation in terms of CPU time has resulted in a significant reduction in the percentage of nodes being subjected to the mesh movement strategy at each adaptive iteration. At the end of the first adaptive cycle, approximately 55% of the nodes of the domain were swept whereas only 5% of the nodes had not met the relative metric error limit at the end of the sixth adaptive cycle. The first and second adaptive cycles used 60% of total CPU time and this value decreased to 17% for the fifth and sixth cycles. A plot of the number of edges versus the metric error is shown in fig. 6.36.

6.7 Viscous Flow in a Square Duct with 90° Bend

The ability of the mesh movement method to accurately capture secondary flow features using a low number of optimally placed grid points is demonstrated for laminar viscous flow in a square duct with a 90° bend. This geometry was first studied experimentally by Humphrey et al. in 1977 [75].

The problem was non-dimensionalized using the square root of the inlet area as the unit length and the average inflow velocity as the unit velocity. The test case was solved at a Reynolds number of 790 based on the unit length and velocity. The computational domain begins with a straight inflow section of five units of length upstream of the bend and ends with an outflow section of similar length downstream of the bend. The curved surfaces have inner and outer radii of 1.8 and 2.8 units of length, respectively. A fully developed velocity profile was prescribed at the inlet using the analytical solution given by White [82].

The aim of this test case was to investigate the relationship between some error properties of non-adapted and adapted grids. Six different grids were used in a non-adaptive setting and possessed the following dimensions: 11x11x31 (3,751 nodes), 21x21x61 (26,901 nodes), 31x31x91 (87,451 nodes), 36x36x106 (137,376 nodes), 41x41x121 (203,401 nodes), and 51x51x151 (392,751 nodes). Among these grids, four of them were also subjected to grid adaptation: 11x11x31, 2x21x61, 31x31x91 and 41x41x121. Figure 6.37 presents the initial and adapted surface meshes of a 21x21x61 grid of the square duct while figure 6.38 provides a mid y-plane cut of the same geometry.

Each of the four grids studied in the adaptive setting underwent five cycles of adaptation. When the flow residual dropped by three orders of magnitude, the nodes were displaced 400 times. However, on the final adaptive cycle, the solution was considered converged only when the residual had decreased by five orders of magnitude. As the adaptive cycles proceeded, the amount of artificial dissipation was progressively decreased. The reduction in the level of artificial dissipation required for the final adapted solution ranged from 50-80% less than for the initial non-adapted solution. The nodal displacement convergence history, plot of the nodal percentage undergoing the adaptive algorithm and the error distribution between the non-adapted and adapted grids are quite similar in pattern and trend for all the four adapted cases and, therefore, fig. 6.39-6.41 displays these plots only for the 31x31x91 (87,451 nodes) grid.

Figure 6.42(a-b) shows the effect of mesh movement on increasingly finer grids in terms of the streamwise velocity profiles plotted along radial lines taken at 60° and 90° cross-sections. The 60° velocity profile is measured at y=0.25 which is half-way between the x-z plane wall and the x-z symmetry plane. The 90° profile is taken at y=0.5 or the x-z symmetry plane.

Figure 6.43(a-b) compares the streamwise velocity profiles at 60° and 90° cross-sections on the 21x21x61 adapted grid with the experimental data of Humphrey [75] as well as the results of Yeo et al. [77] and Lin et al. [76], which are obtained with different numerical schemes. The adapted results agree very well with Yeo's and Lin's predictions which were obtained on significantly finer grids. As noted by Lin et al. [76], the discrepancies between the experimental and numerical data would tend to suggest that the flow conditions in Humphrey's experiment do not correspond to those used for the calculations.

One objective of this study was to determine the size of a non-adapted grid that would be equivalent to that of a given adapted grid for the same level of solution error. It is assumed that the finest adapted grid (41x41x121) would provide the best solution. The difference between the solutions obtained on each of the six non-adapted grids and the adapted 41x41x121 grid is calculated as an error in the L_2-norm and plotted as the top curve in fig. 6.44. The lower curve in the figure is obtained by computing the L_2-norm error between the solutions on each of the four adapted grids and the adapted 41x41x121 grid. For this purpose, the L_2-norm error between nodal solutions $u1$ and $u2$ on the two concerned grids has been defined as

$$\sqrt{\frac{\sum_{i=1}^{nnode}\left(u1_i - u2_i\right)^2}{nnode}} \tag{6.2}$$

Referring to fig. 6.44, one observes that, for the same level of error (for example, error=0.535), the adapted 21x21x61 (26,901 nodes) grid is equivalent to a non-adapted grid of approximately 270,000 nodes. It is interesting to note that the solution on the adapted 26,901 node grid required a total solution and adaptation time of 7.1x10^4 CPU seconds whereas the non-adapted cases of 203,401 nodes (41x41x121) and 392,751 nodes (51x51x151) needed 3.7x10^5 and 2.5x10^5 CPU seconds for solution time, respectively. Thus, one can achieve not only the same level of solution error by solving and adapting on a significantly coarser grid but, more importantly, the time required to obtain the solution on the adapted grid was approximately four to five times faster. A comparison of the streamwise velocity profiles at 60° and 90° cross-sections obtained on the adapted 21x21x61 and non-adapted 41x41x121 and 51x51x151 grids is shown in fig. 6.45(a-b).

Up to this point, all the grids considered for the square duct have been composed of hexahedral elements. It has been observed that, for the same number of nodes, a tetrahedral grid will give a less accurate solution than a hexahedral grid. However, one advantage of the tetrahedral element is that it possesses a geometric flexibility which is more conducive to grid adaptation. It is for these reasons that non-adapted and adapted solutions were obtained on a tetrahedral mesh consisting of 26,991 nodes and 144,939 elements. This grid size was selected so that comparisons could be made with the 21x21x61 (26,901 nodes) hexahedral grid. The error between the solutions on the non-adapted tetrahedral mesh and the adapted 41x41x121 grid was calculated to be higher than that of the similar sized hexahedral grid. This is the expected result and, as the data points labeled *'Tetra. Unadapted'* and *'Hexa. Unadapted'* in fig.

6.44 indicate, the non-adapted tetrahedral grid had a difference in solution error of 0.972 while the non-adapted 21x21x61 hexahedral grid had a lower value of 0.875. However, the adapted hexahedral grid had a higher error than the adapted tetrahedral mesh, that is, 0.535 versus 0.374 as shown by the labels *'Hexa. Adapted'* and *'Tetra. Adapted'*. The lower solution error in the adapted unstructured grid may be attributed to the flexibility of the tetrahedral element, thus allowing grid points to move with less constraints to more optimal positions as dictated by the error estimate. If one draws a horizontal line across the curves in fig. 6.44, one can say that, for the same level of solution error, the adapted 26,991 noded tetrahedral grid is equivalent to an adapted hexahedral of about 50,000 nodes or a non-adapted hexahedral grid of approximately 400,000 nodes. Figure 6.46 compares the streamwise velocity profiles along radial lines at 60° cross-section for the adapted hexahedral and tetrahedral grids and the unstructured grid provides a better prediction of the two velocity maxima. It should be noted that the tetrahedral case underwent the same grid adaptation protocol as the hexahedral one.

6.8 Laminar Non-Reacting Flow in Hydrogen Fuel Can Combustor

Combustor analysis offers one of the most promising applications of unstructured grids in gas turbine engines. Due to their geometrical complexity, they do not lend themselves well to multi-block grid generation. In addition, grid adaptation may play a very important role in resolving the complicated nature of combustor flow.

A high shear direct injection experimental can combustor was chosen for its relative geometric simplicity compared to gas turbine combustors while keeping the flow complexity level nearly the same [78]. The hydrogen fuel is injected radially into a swirling air flow and the mixture undergoes a sudden expansion at the entry to the combustor can. This is a non-trivial test case involving swirling flow, multiple inlets, severe temperature gradients and complex flow features at the hydrogen jets and the combustor backstep region.

The current analysis has been performed on the full 360° geometry for non-reacting flow and shows the advantages of automatic tetrahedral meshing coupled with an adaptive nodal displacement scheme. This test case is also meant to demonstrate the still important improvements of mesh adaptation even if its use is restricted to a post-processing application. The original ICEM-TETRA unstructured grid is shown in fig. 6.47.

The flow solution on the initial non-adapted grid was quite stable during the first four of the five unloading steps of artificial dissipation. However, after 40 iterations of the final step where the artificial dissipation was the lowest, the solution became unstable and quickly diverged. This indicates that the grid is not suitable for resolving the flow features. The traditional approach to ensure convergence is to increase the artificial dissipation, which is known to degrade the solution quality.

The appropriate approach is to generate an adapted grid from the last stable solution. By using only one mesh adaptation cycle, which consisted of 200 iterations of the nodal displacement scheme at a relaxation factor of 0.9, and restarting the analysis from an interpolated solution on the newly adapted grid, a converged solution is attained after less than 100 iterations. This solution was obtained using the same level of dissipation as requested in the final unloading step. Figure 6.48 displays the flow solver convergence history.

Figure 6.49 shows the adapted surface mesh of the combustor geometry. Comparisons of the adapted grid with the original grid at axial and mid-longitudinal sections of the can combustor are provided in fig. 6.50-51. It is clear that the grid adaptation algorithm has no difficulty resolving multiple "hot spots" in the solution. The hydrogen fuel jets, the swirler walls and the combustor entry backstep are all well refined in the new grid. The amount of nodal movement is quite significant considering that only 200 sweeps over all the nodes were requested. The coarseness of the mesh on the axis sheds light on the nature of the flow field in this region as grid points are moved away from this low activity region. Figure 6.52 displays an axial cross-section showing the velocity vectors at the injection location.

6.9 Inviscid Flow over a Wing-Pylon-Nacelle Configuration

The NASA wing-pylon-nacelle configuration was chosen to validate the adaptive scheme due to the availability of extensive experimental data [79, 80]. The development of CFD codes to simulate complex interactions and physical flow phenomena associated with engine-nacelle integration is a very challenging task and this experimental setup provides a simplified version of the problem on which computational techniques may first be developed.

As described in [79], the wing-pylon-nacelle interaction was isolated by removing the wing-body and nacelle-body interactions and removing the effects due to sweep, taper and twist of the wing. The experimental configuration consists of a long duct, flow-through nacelle mounted on a swept pylon under an unswept wing. The wing has a supercritical cross-section, constant in the spanwise direction. The nacelle is axisymmetric with a NACA inlet section followed by a cylindrical section and circular-arc boat tail. The nacelle axis is parallel to the wing chord. The leading edge is straight and is swept at 75°. A horizontal cut through the pylon comprises a leading edge section, a section of constant thickness, and a boattail section. The leading edge section and boattail sections are the same on each horizontal cut while the length of the constant thickness section varies due to the difference in sweep angle between the leading and trailing edge. Inviscid analysis was performed for a subsonic case at Mach number 0.6 and zero angle of attack.

The main difficulty presented by this configuration is the exceedingly thin nacelle leading edge which creates a rapid flow acceleration around the outer nacelle lip. To properly capture this acceleration with isotropic tetrahedral grid generation or isotropic refinement would lead to an unrealistically large grid size. It is obvious that due to the directional nature of the nacelle lip acceleration (small gradients in the circumferential direction), anisotropic mesh adaptation would drastically reduce the grid size.

Unlike the previous test cases discussed where only mesh movement was used, the full gamut of adaptive strategies - mesh movement, refinement, coarsening and edge swapping - has been applied to generate the adapted grid. It should be noted that only one cycle of the mesh adaptation procedure has been performed for this test case. Although further adaptation cycles should be carried out, the results of the first adaptation cycle show the tremendous potential of the methodology.

The non-adapted grid has been created using the ICEM-CFD *Tetra* automatic unstructured isotropic grid generator and consists of 265,136 nodes and 1,416,022 elements (fig. 6.53). The element size and growth rate of the elements can be specified on the different CAD entities (curves and surfaces). A solution was obtained on the non-adapted grid using the lowest amount of artificial dissipation possible (μ_{sw}=0.0 and μ_{cw}=0.3). Figures 6.54-6.55 compare the experimental and numerical Cp distributions along the wing, away from the nacelle, and along the nacelle body at two locations, 22.5° and 180° from the top dead center position, respectively. It can be observed that the acceleration at the leading edge is not well resolved for either the wing or nacelle. This grid clearly does not resolve the flow features around the leading edges accurately. To improve the resolution at the leading edges, anisotropic mesh adaptation (mesh movement, mesh refinement/coarsening and edge swapping) was applied. An edge-based error estimate using the Mach number solution on the non-adapted grid was used to drive the adaptive cycle.

In this particular test case, the objective was to apply the adaptive scheme to improve the grid while reducing the grid size. This is made possible due to the anisotropic nature of the adaptive procedure and the resulting optimal placement of grid points. It results in a very coarse grid approximately half the size of the non-adapted grid (132,531 nodes and 709,012 elements) but with a much better resolution at the wing and nacelle leading edges. This grid was generated by specifying a minimum aspect ratio of 0.001. It should be remarked that the majority of elements of the non-adapted grid possessed an aspect ratio between 0.8-1.0. The aspect ratio histogram of the adapted grid (fig. 6.56) clearly depicts that the specified degree of anisotropy has been achieved.

A solution was obtained on the adapted grid using the smallest amount of artificial dissipation possible (μ_{sw}=0.0 and μ_{cw}=0.3). This is the same level of dissipation that was used in obtaining the non-adapted solution. Figures 6.57-6.58 reveal that the leading edge acceleration at the nacelle and wing is better captured. On the wing lower surface, however, the solution does not exhibit the degree of acceleration indicated by the experimental data. This is not surprising since the non-adapted grid solution, which is used to adapt the grid, has the same deficiency. It is expected that multiple adaptive cycles, accompanied by a gradual lowering of the amounts of artificial dissipation, will further improve the quality of the grid and solution. Views of the adapted and original grids are presented in fig. 6.59-6.61 The stretching of the elements and the fineness of the grid at the leading edges are quite noticeable on the adapted grid. The results of the first adaptation cycle have been presented here to show the tremendous potential of the methodology and further work on this test case is continuing at the CFD Laboratory and P&WC.

FIGURES

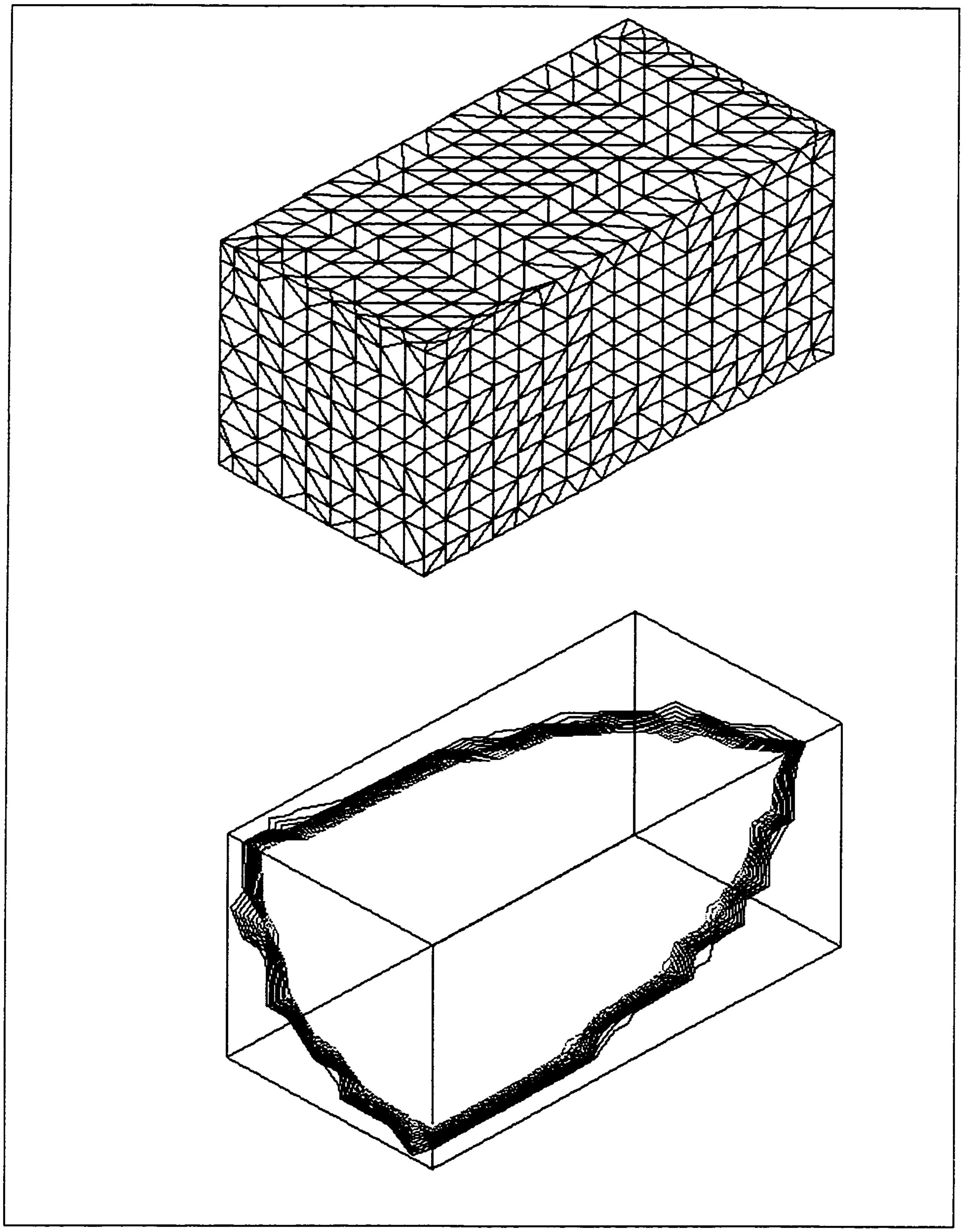


Figure 6.1 Initial non-adapted tetrahedral grid and corresponding isolines of analytical function

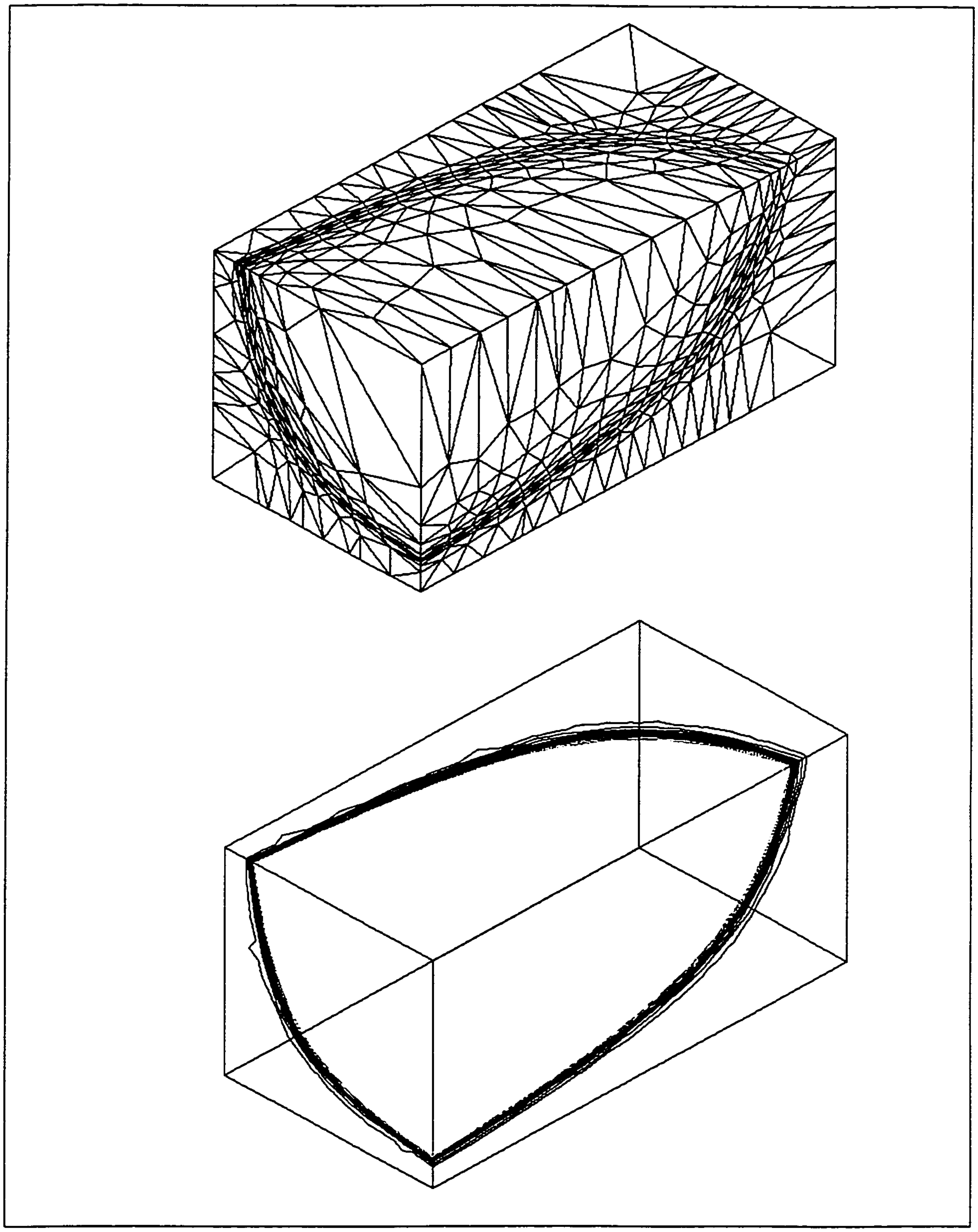


Figure 6.2 Adapted tetrahedral grid and corresponding isolines of analytical function

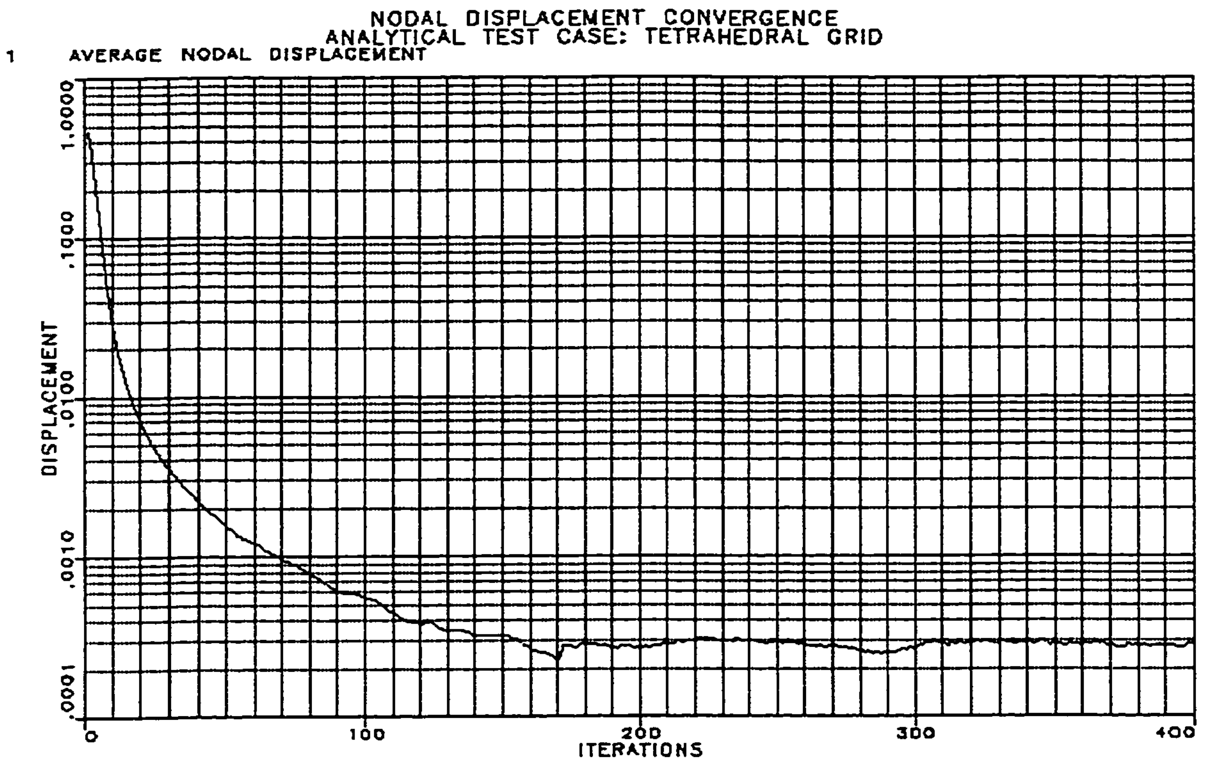


Figure 6.3 Convergence of mesh movement scheme for analytical case on tetrahedral grid

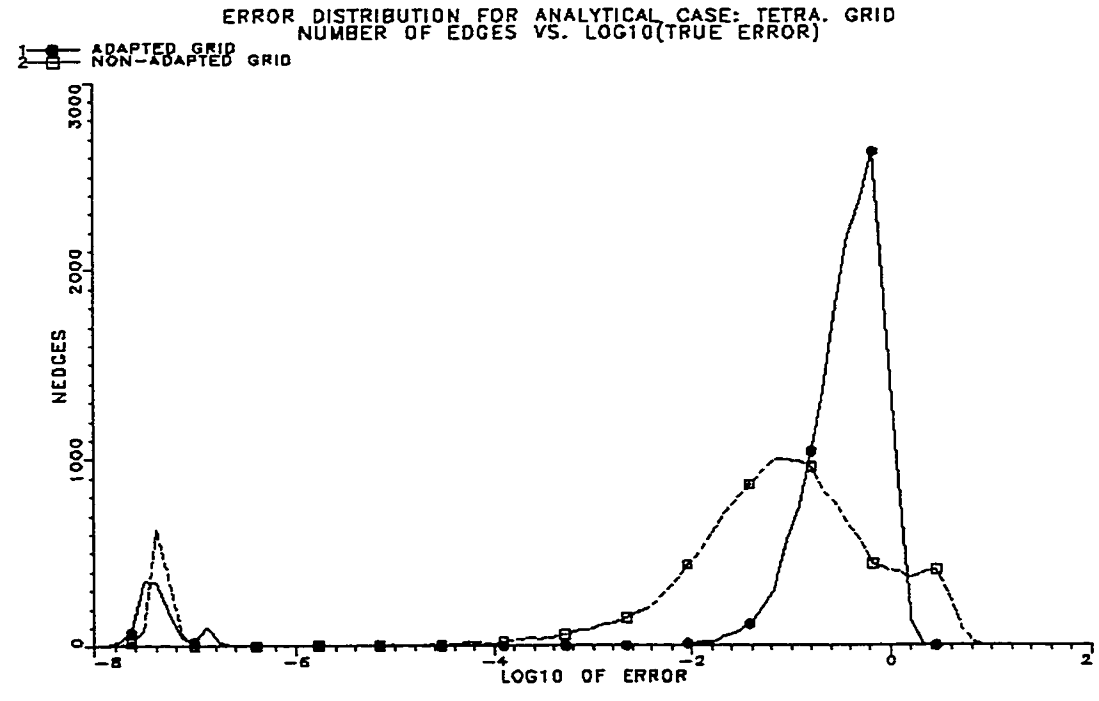


Figure 6.4 Analytical case on tetrahedral grid: error distribution over edges for original and adapted grids

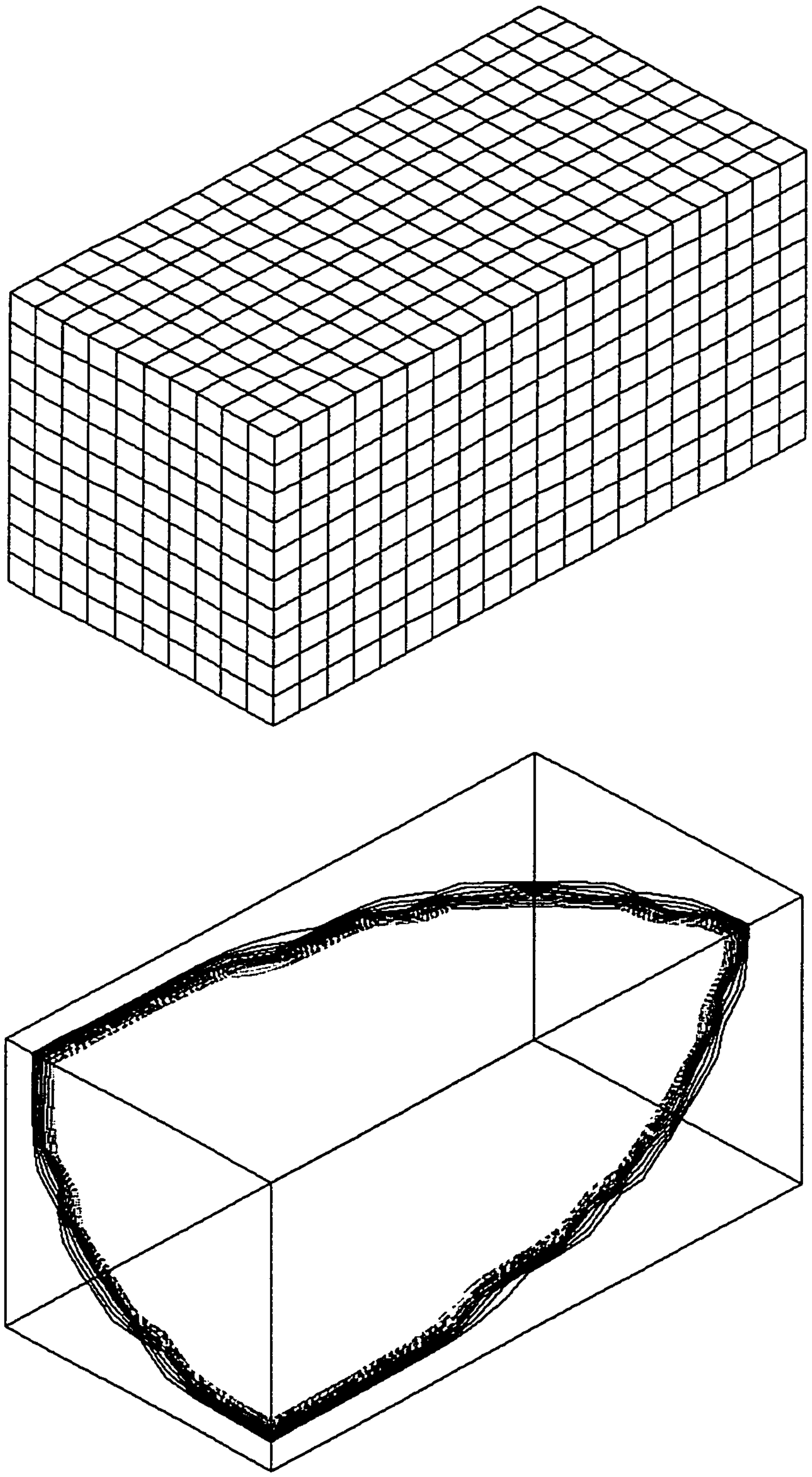


Figure 6.5 Initial non-adapted hexahedral grid and corresponding isolines of analytical function

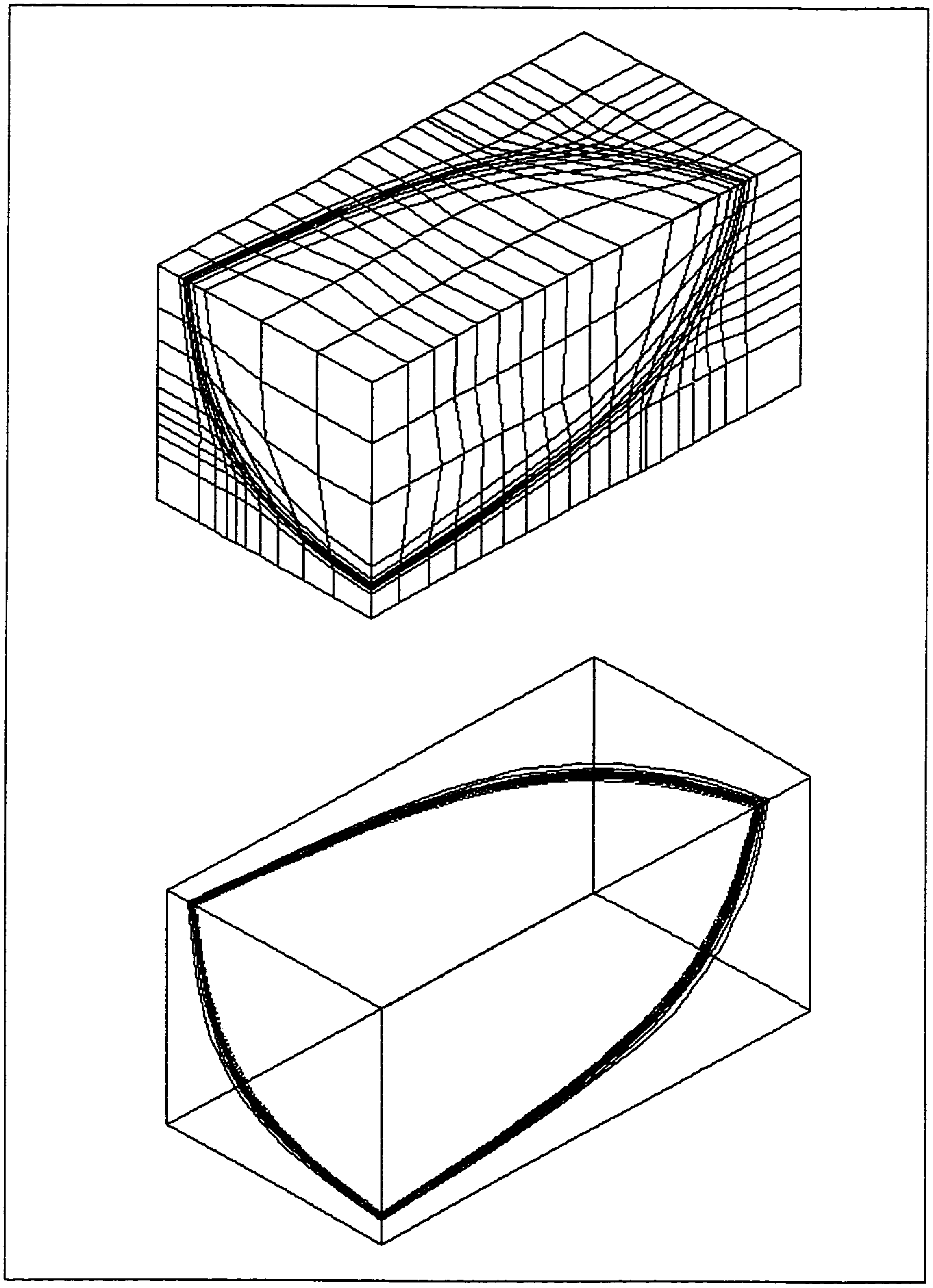


Figure 6.6 Adapted hexahedral grid and corresponding isolines of analytical function

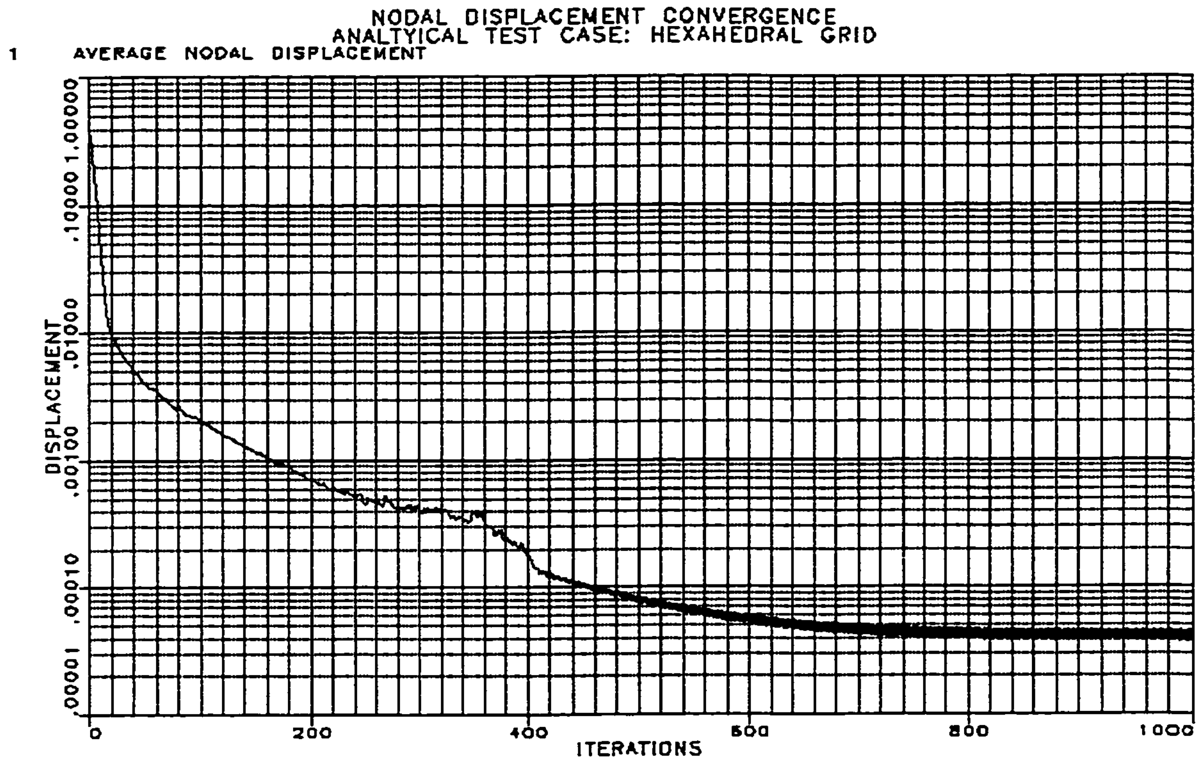


Figure 6.7 Convergence of mesh movement scheme for analytical case on hexahedral grid

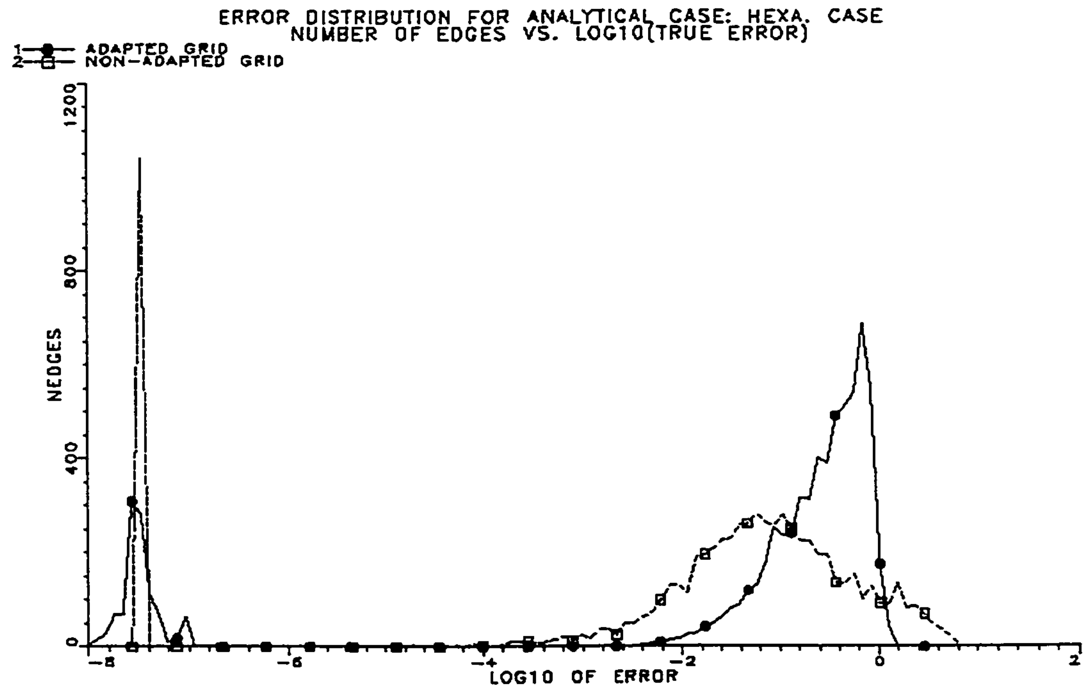


Figure 6.8 Analytical case on hexahedral grid: error distribution over edges for original and adapted grids

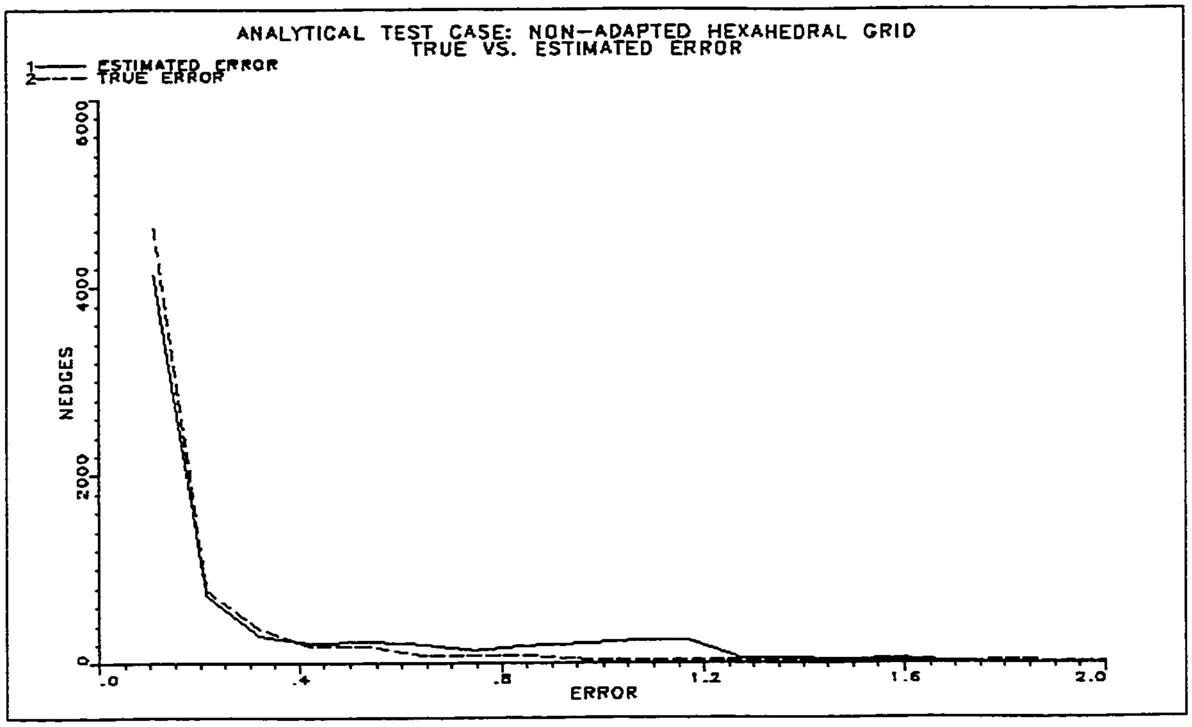


Figure 6.9 Plot of true vs. estimated error for analytical function on non-adapted hexahedral grid

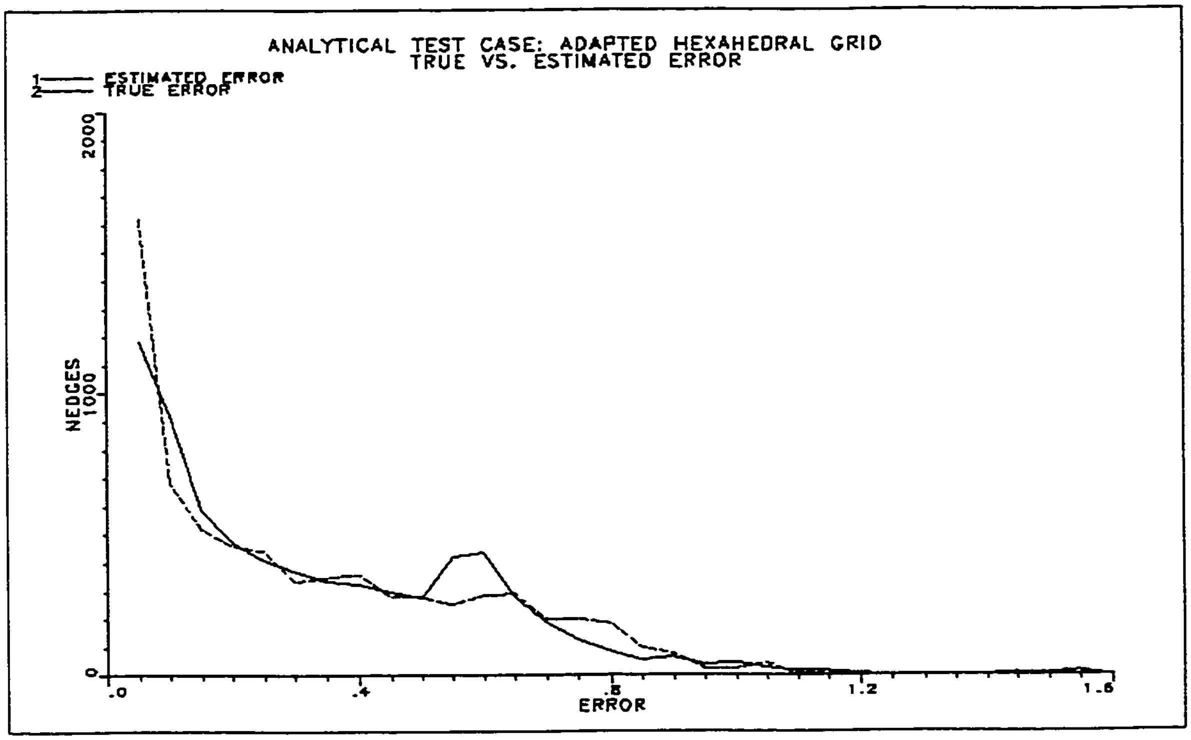


Figure 6.10 Plot of true vs. estimated error for analytical function on adapted hexahedral grid

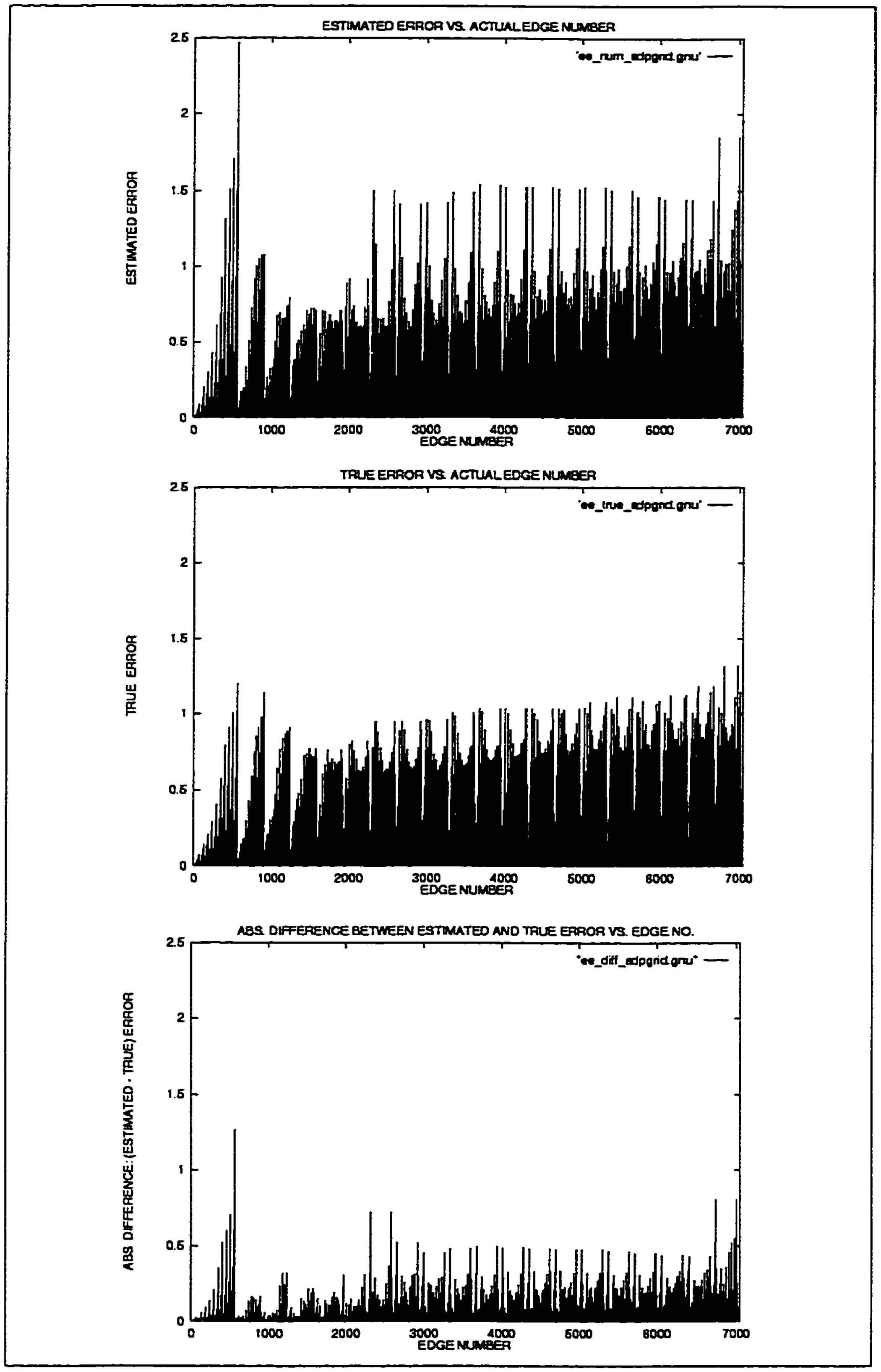


Figure 6.11(a-c) Plots of error versus actual edge number for the analytical test case on adapted hexahedral grid

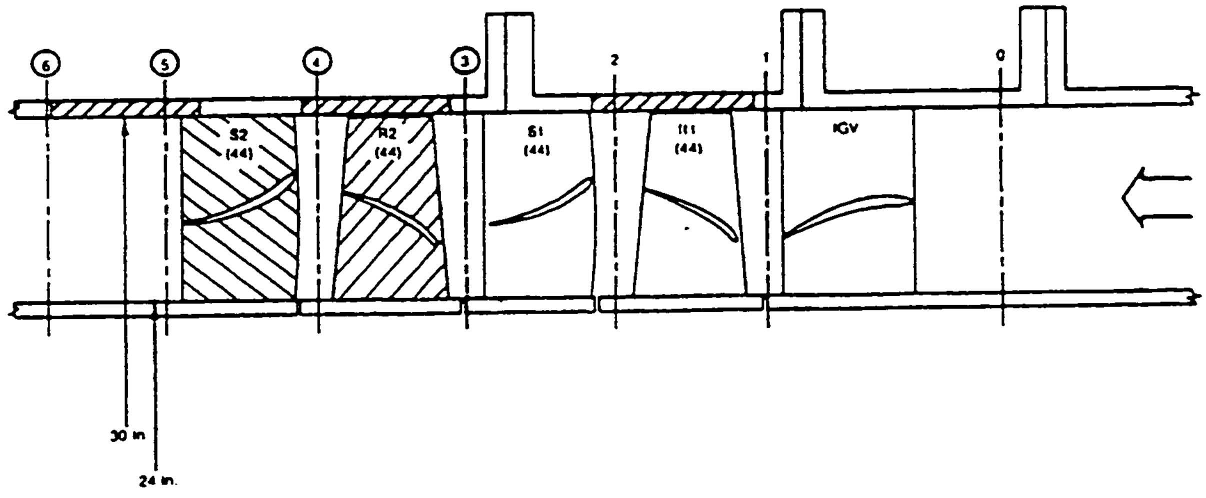


Figure 6.12 UTRC 2.5 Stage compressor model [73]

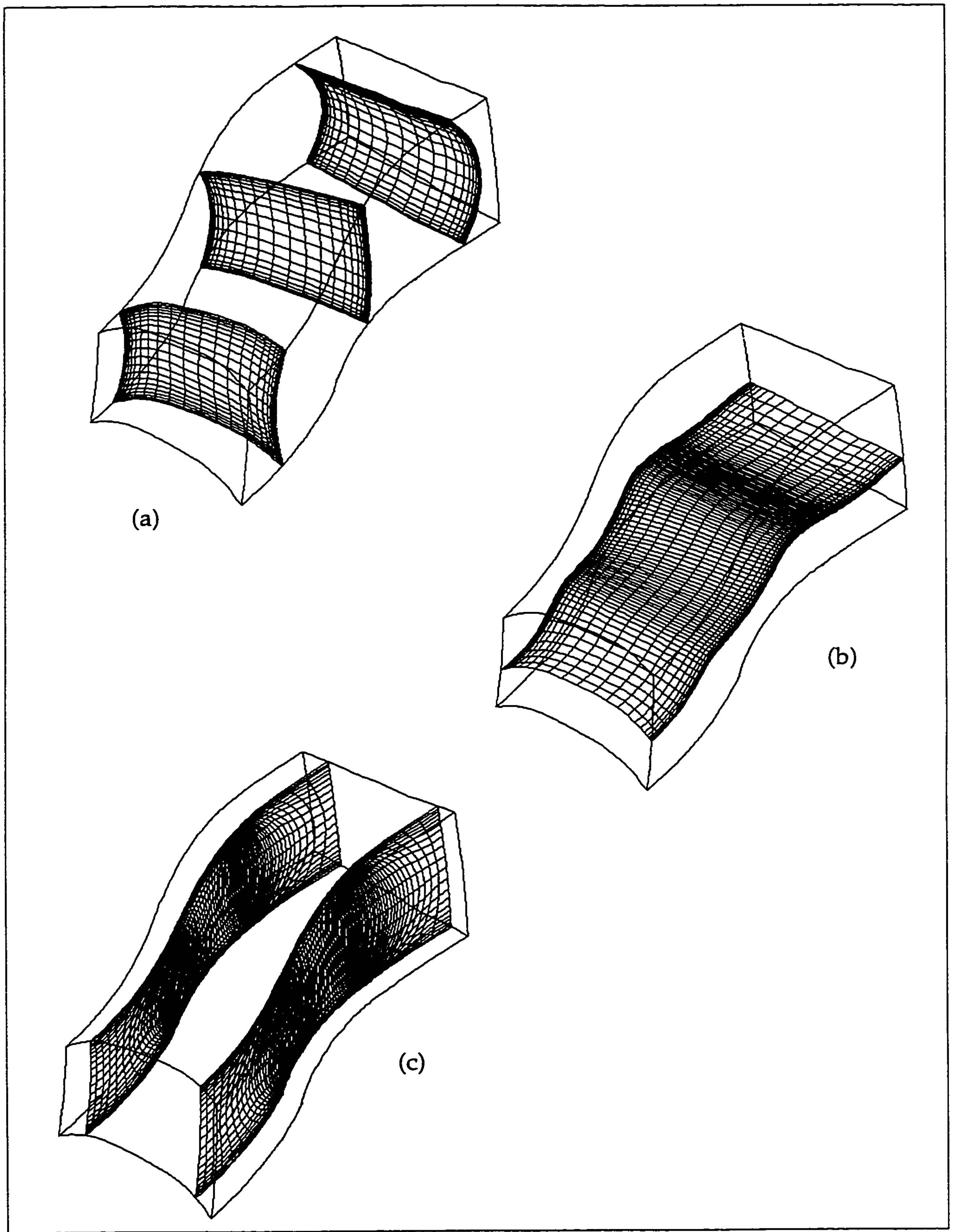


Figure 6.13 Various planes of original grid (84x21x25) of second stage UTRC stator (a) i = 5, 39, 82 planes (b) j = 10 plane (c) k = 9, 18 planes

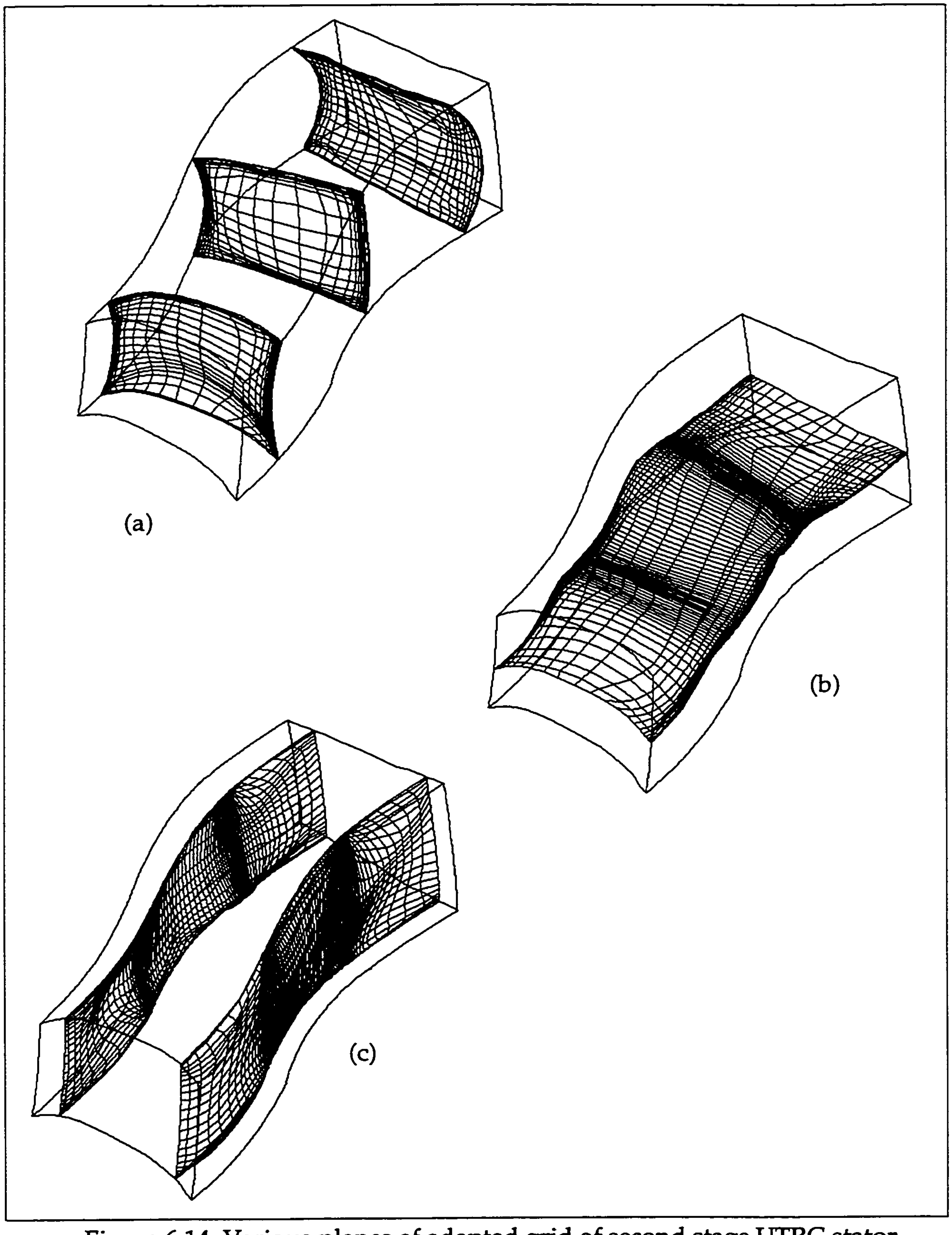


Figure 6.14 Various planes of adapted grid of second stage UTRC stator (a) i = 5, 39, 82 planes (b) j = 10 plane (c) k = 9, 18 planes

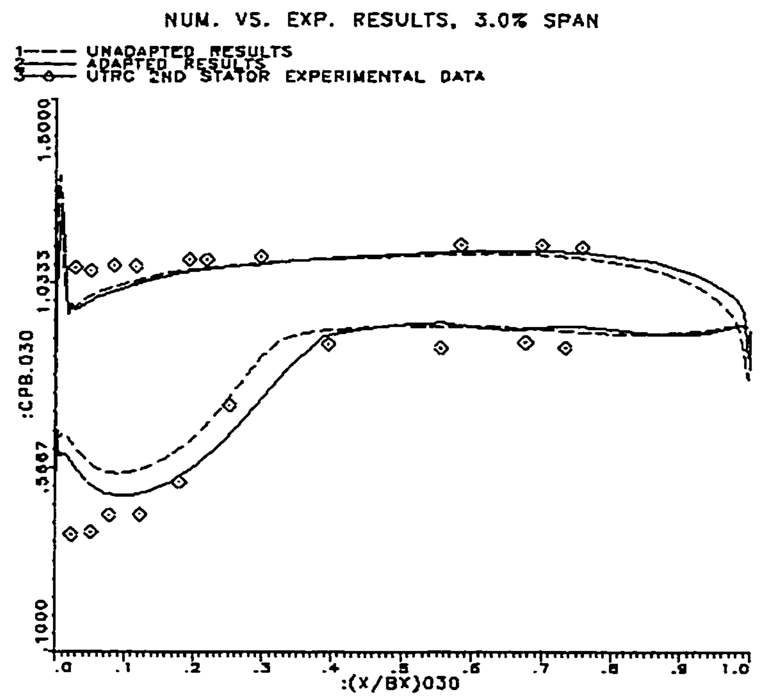


Figure 6.15(a) UTRC stator: Cp distribution at 3.0% span

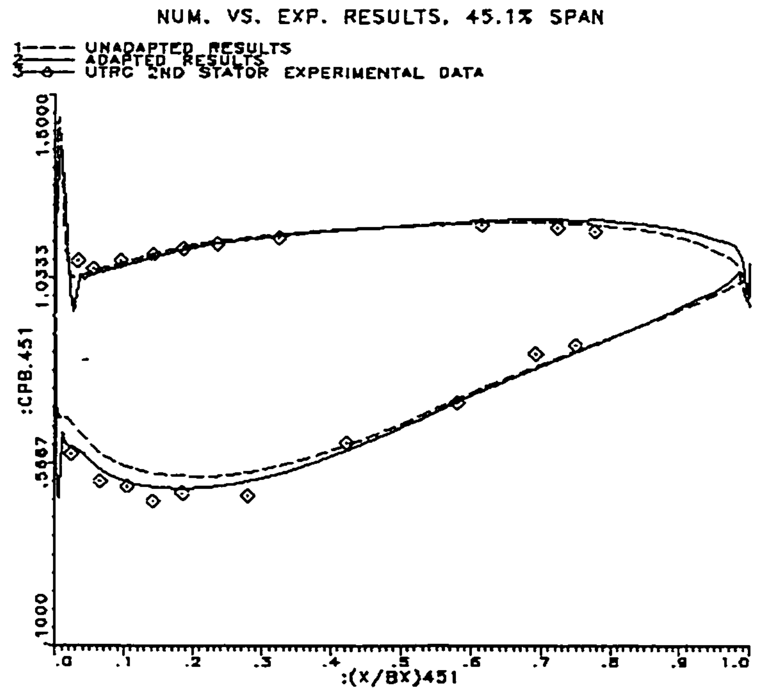


Figure 6.15(b) UTRC stator: Cp distribution at 45.1% span

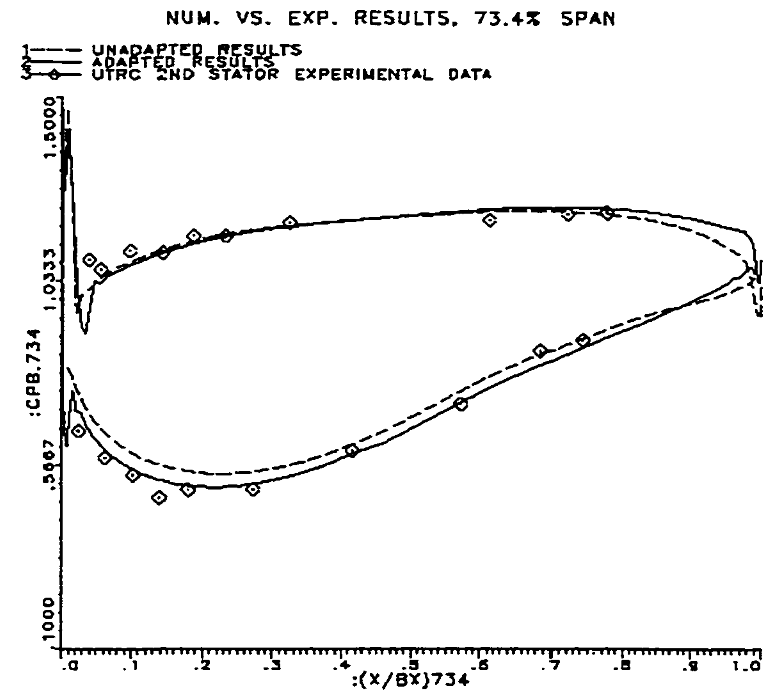


Figure 6.15(c) UTRC stator: Cp distribution at 73.4%

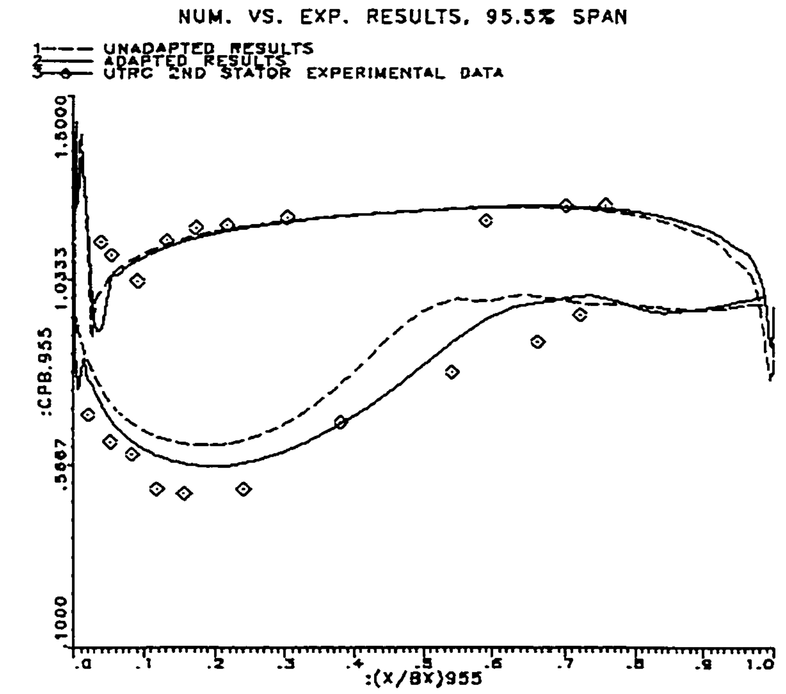


Figure 6.15(d) UTRC stator: Cp distribution at 95.5%

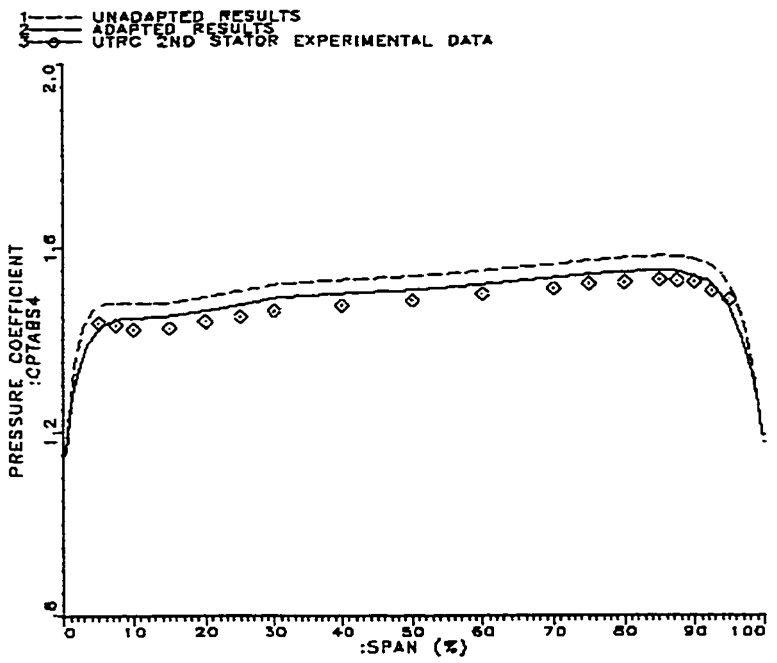


Figure 6.16 UTRC stator: total pressure coefficient at inlet vs.% span

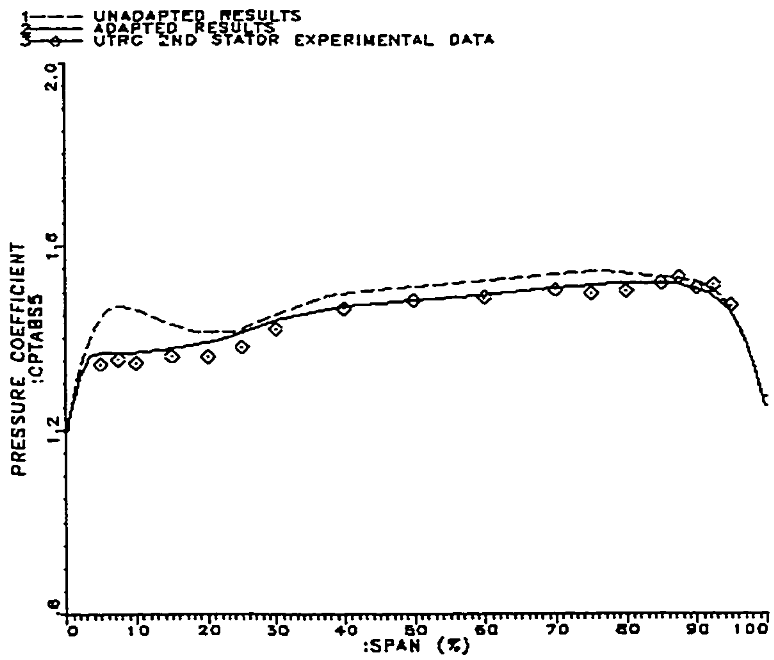


Figure 6.17 UTRC stator: total pressure coefficient at exit vs.% span

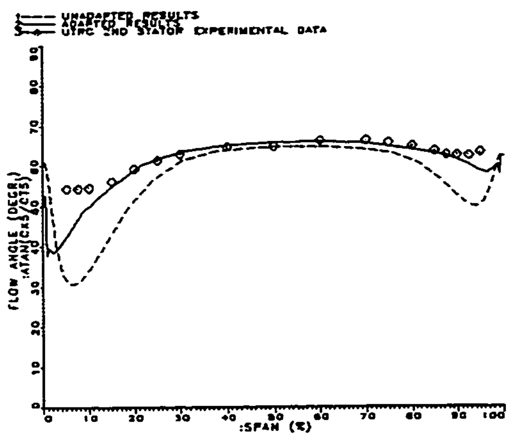


Figure 6.18 UTRC stator: flow angle at exit vs. % span

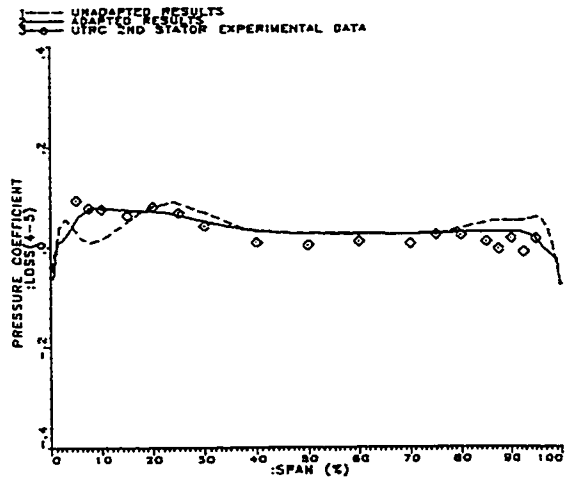


Figure 6.19 UTRC stator: distribution of loss vs. % span

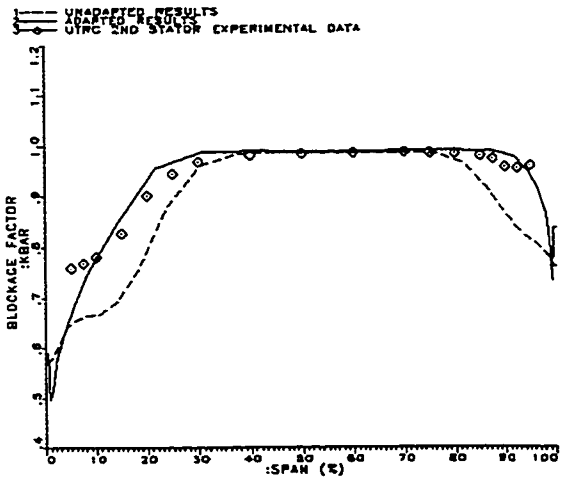


Figure 6.20 UTRC stator: distribution of blockage vs. % span

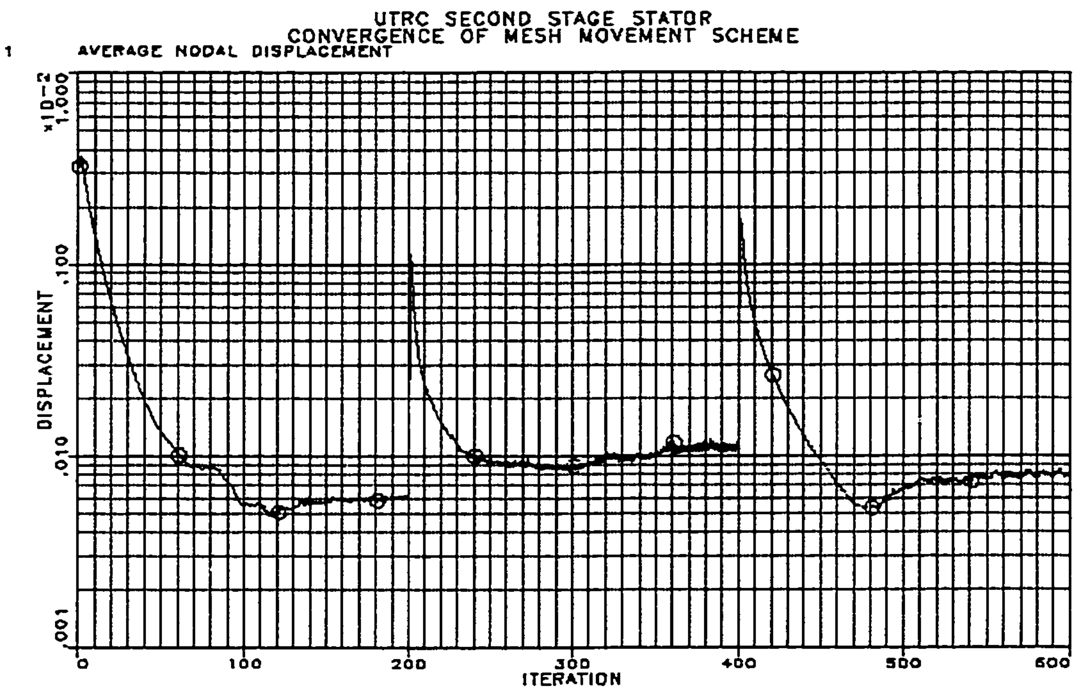


Figure 6.21 UTRC stator: convergence of nodal displacement scheme

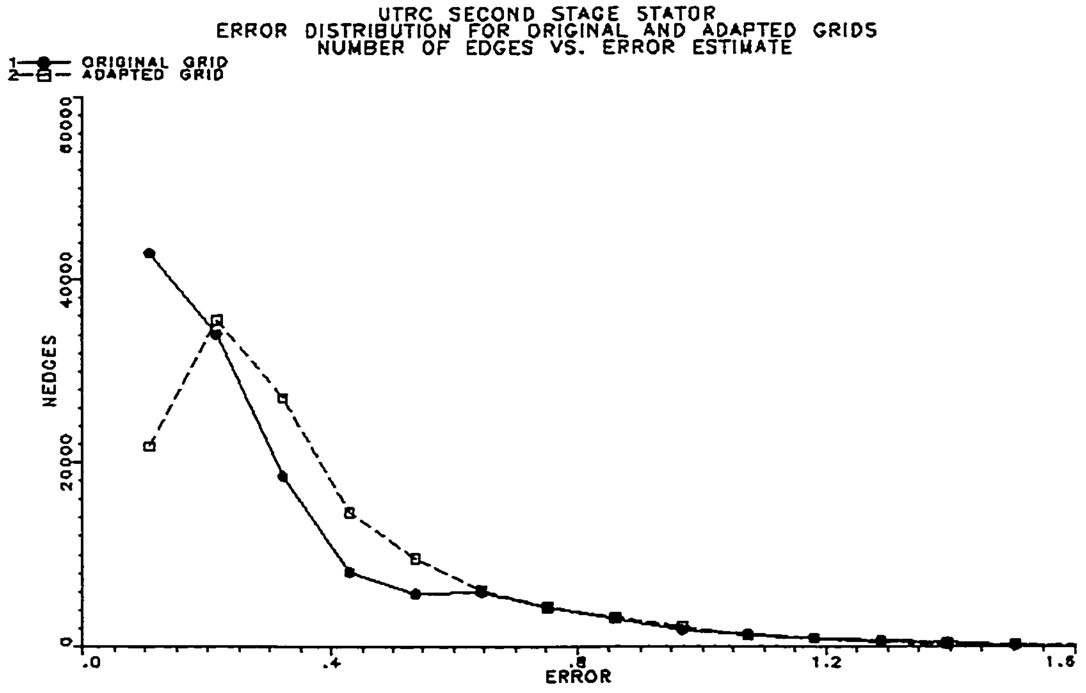


Figure 6.22 UTRC stator: Error distribution over edges for original and adapted grids

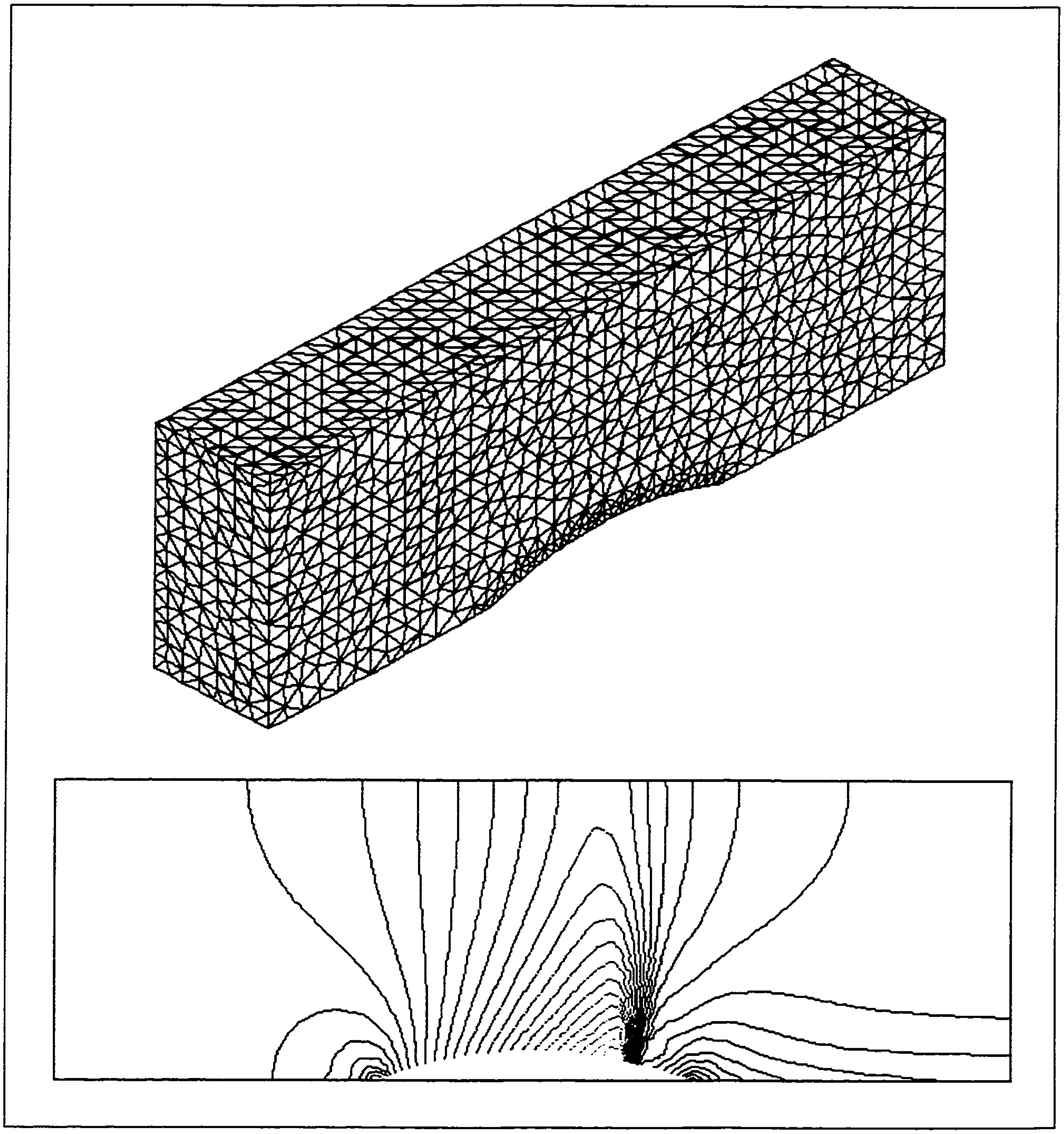


Figure 6.23 Transonic flow over 10% circular arc: original tetrahedral grid (8,885 nodes and 45,873 elements) and Mach contours

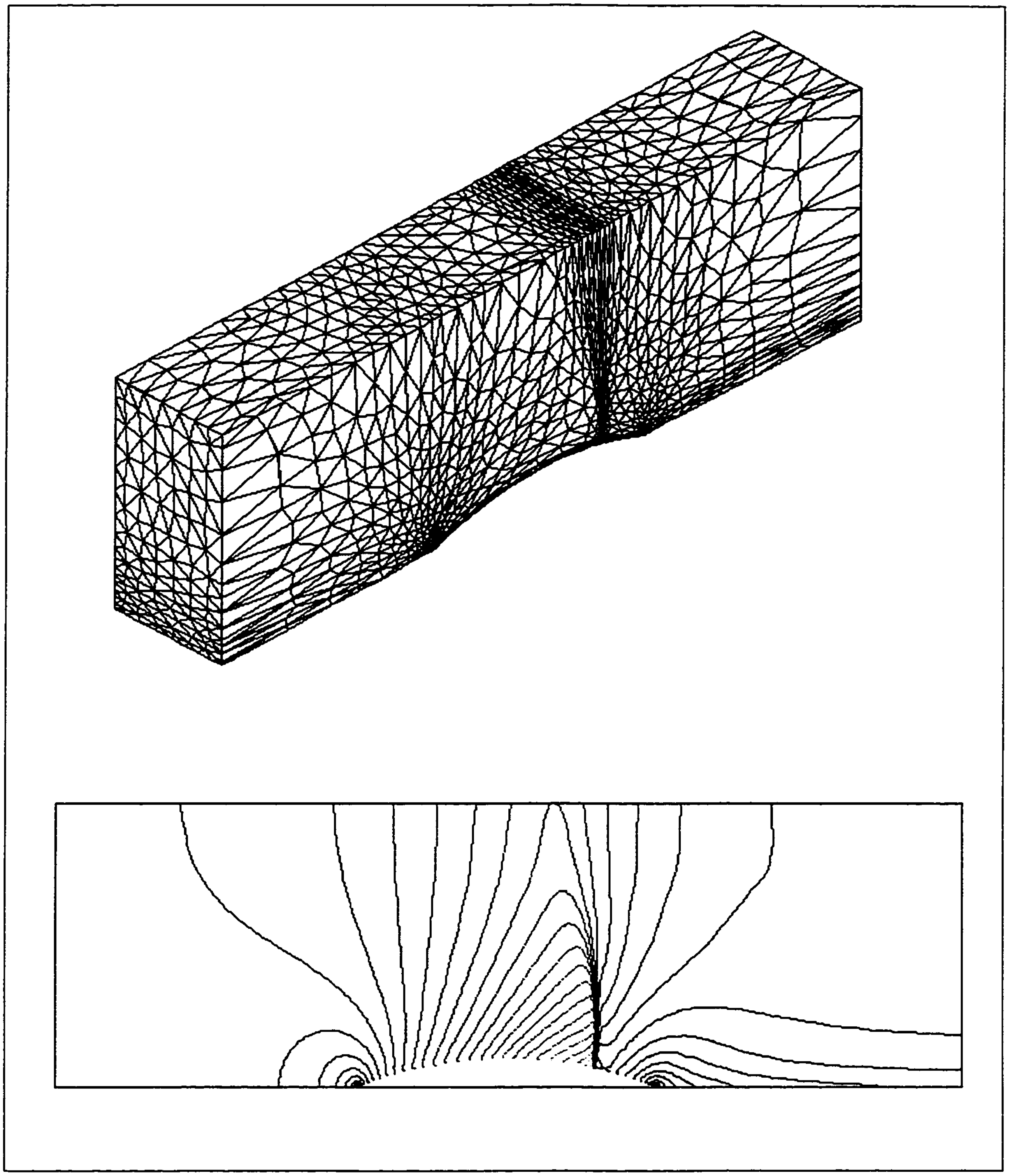


Figure 6.24 Transonic flow over 10% circular arc: adapted tetrahedral grid and corresponding Mach contours

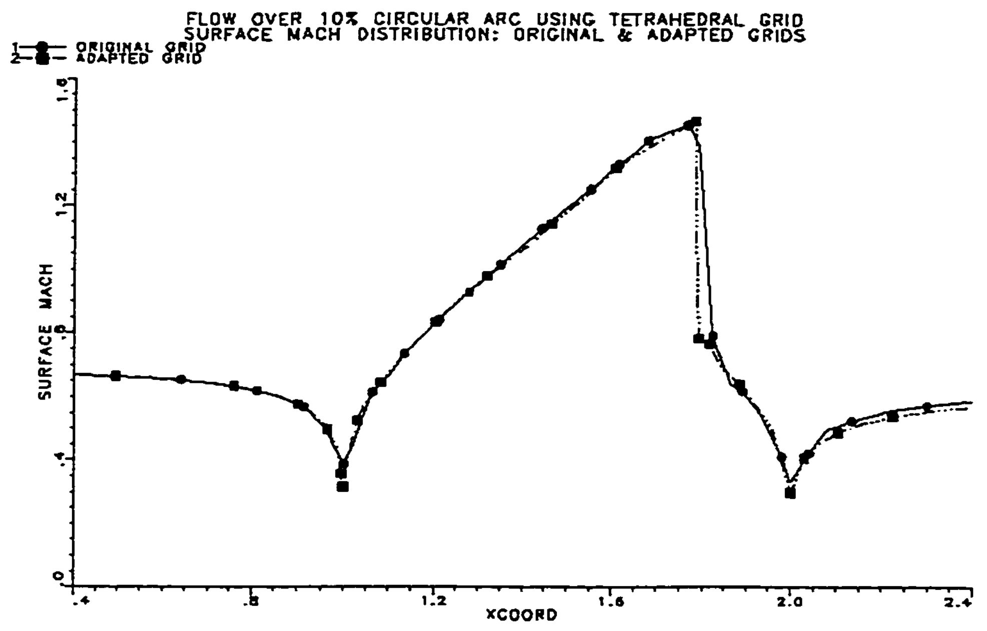


Figure 6.25(a) Surface Mach distribution for original and adapted tetrahedral grids

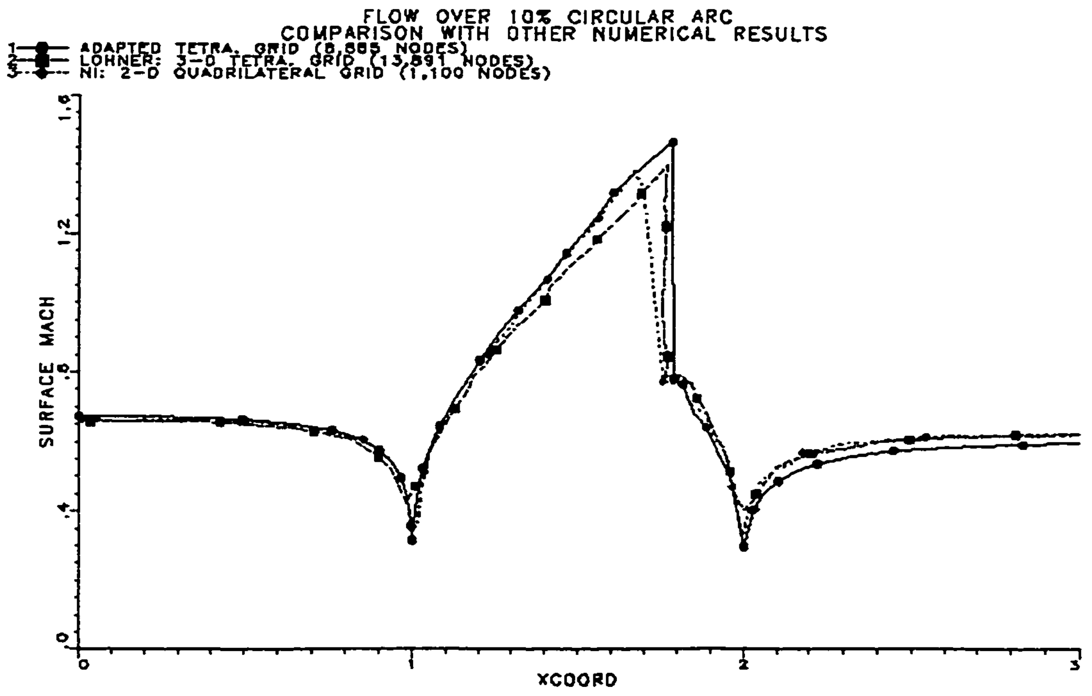


Figure 6.25(b) Comparison of adapted surface Mach distribution with other numerical results (Löhner and Ni)

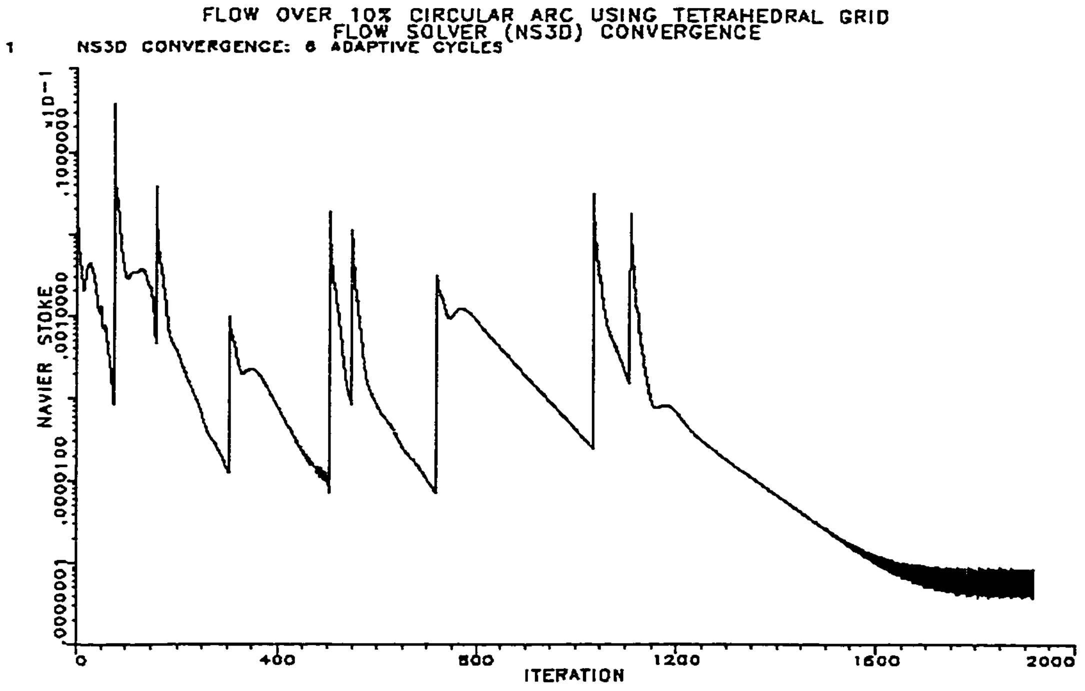


Figure 6.26 Flow solver convergence history over six adaptive cycles for transonic flow over 10% circular arc using tetrahedral grid

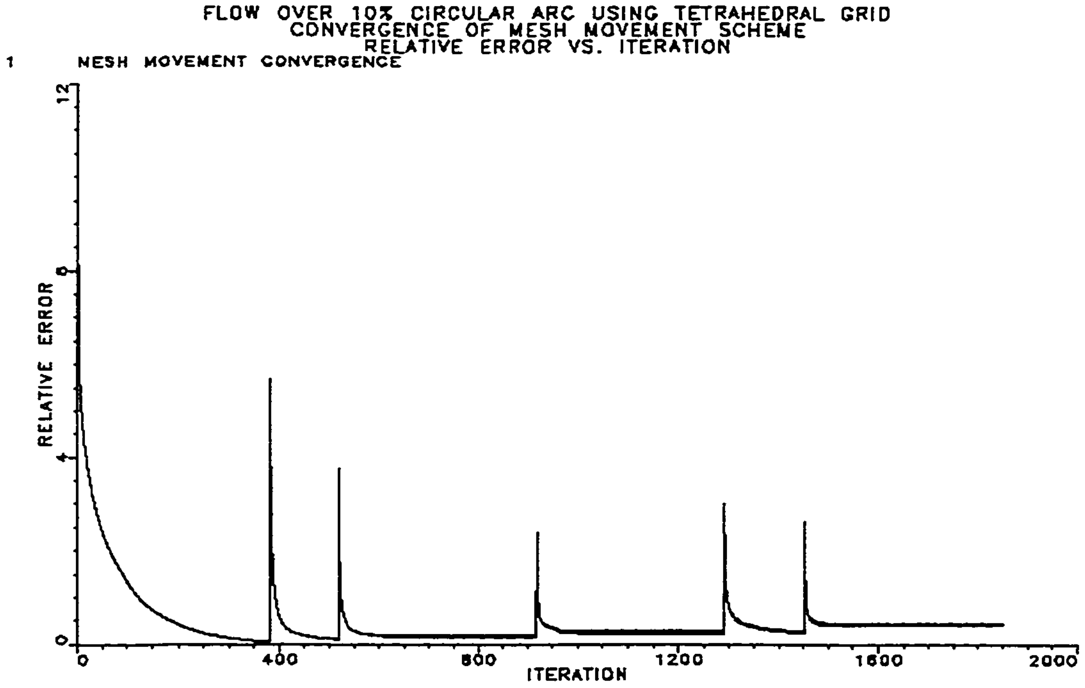


Figure 6.27 Adaptive algorithm convergence history over six adaptive cycles for transonic flow over 10% circular arc on tetrahedral grid

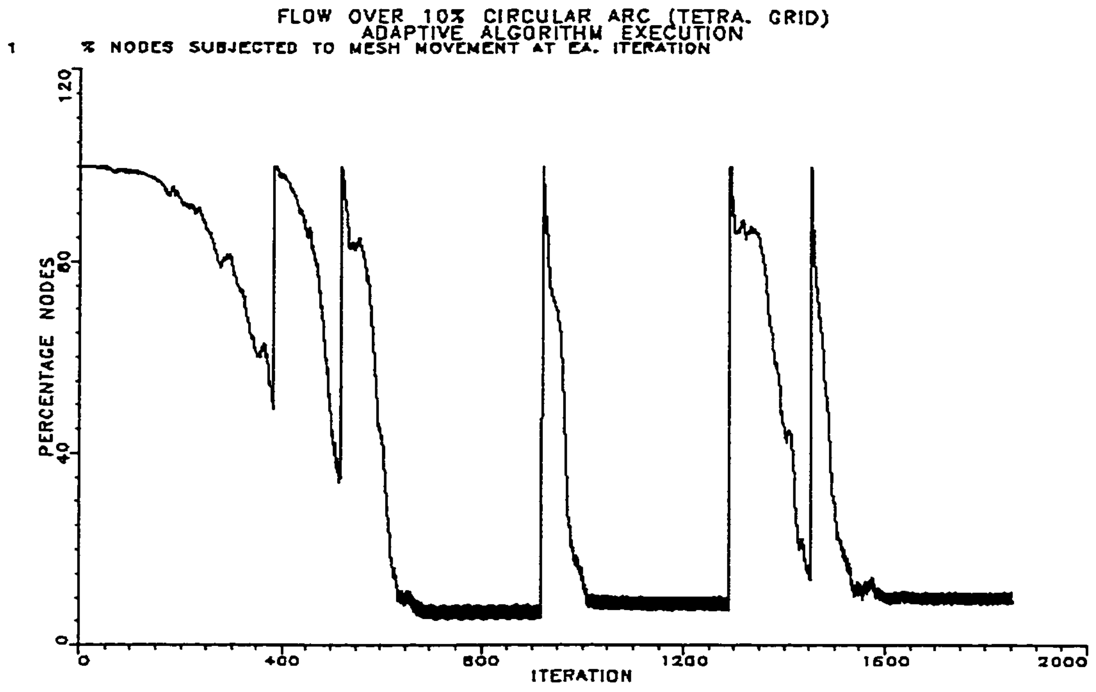


Figure 6.28 Percentage of nodes subjected to nodal displacement strategy for transonic flow over 10% circular arc using tetrahedral grid

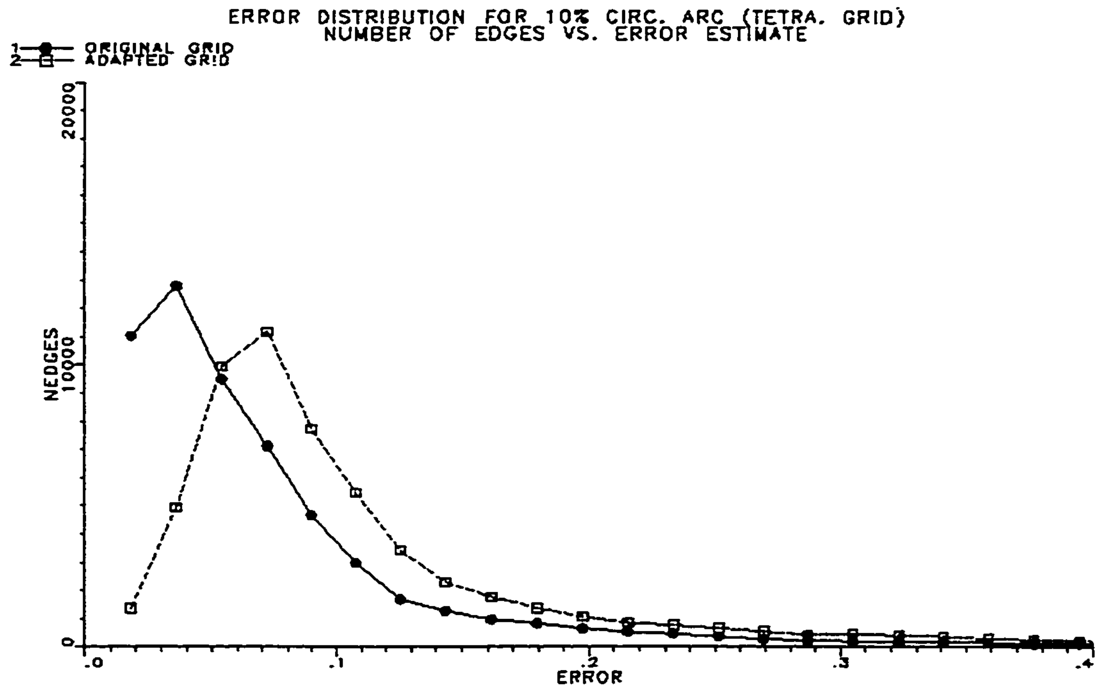


Figure 6.29 Error distribution over edges for original and adapted grids for transonic flow over 10% circular arc using tetrahedral grid

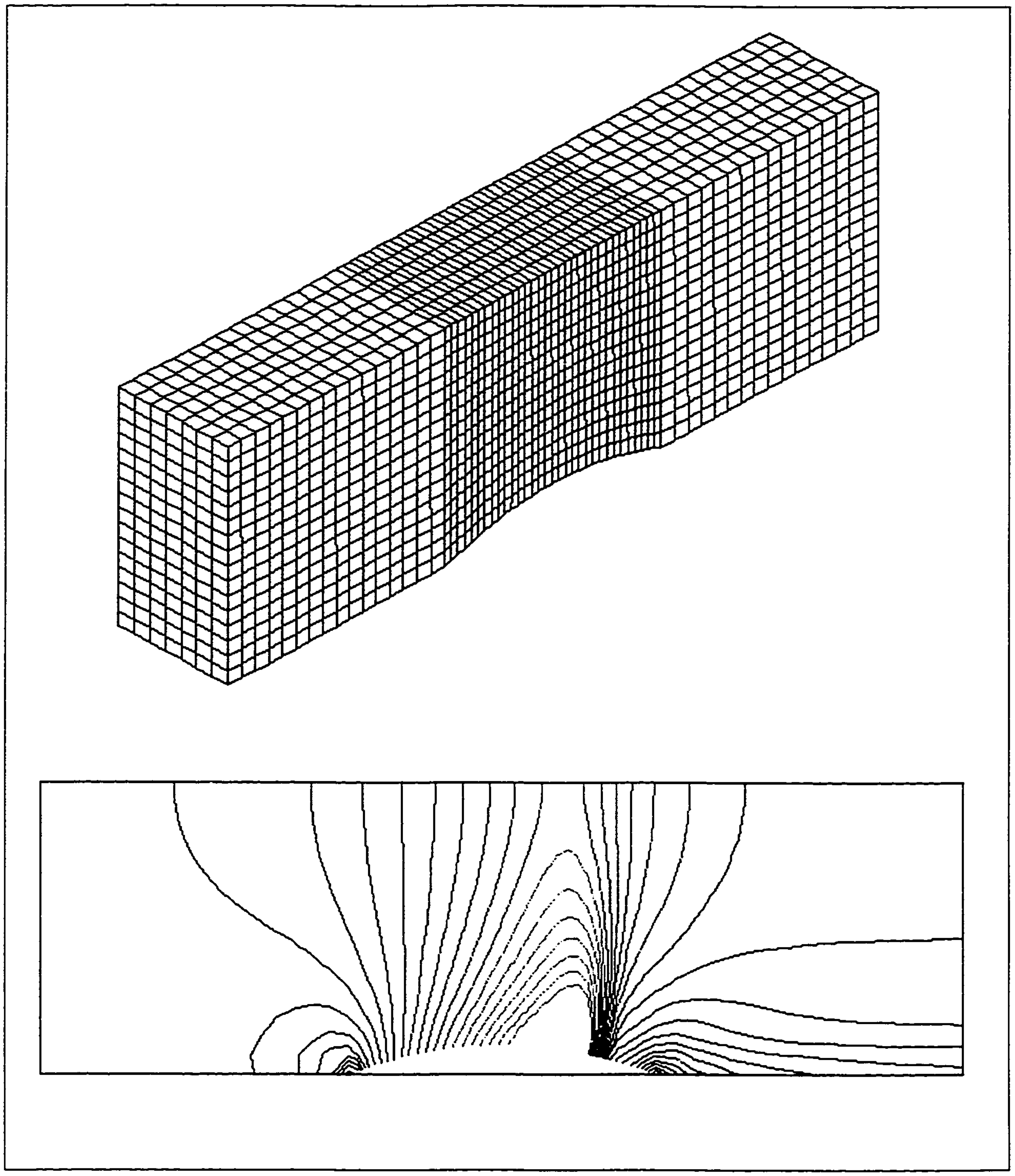


Figure 6.30 Transonic flow over 10% circular arc: original hexahedral grid (8x17x65; 8840 nodes) and Mach contours

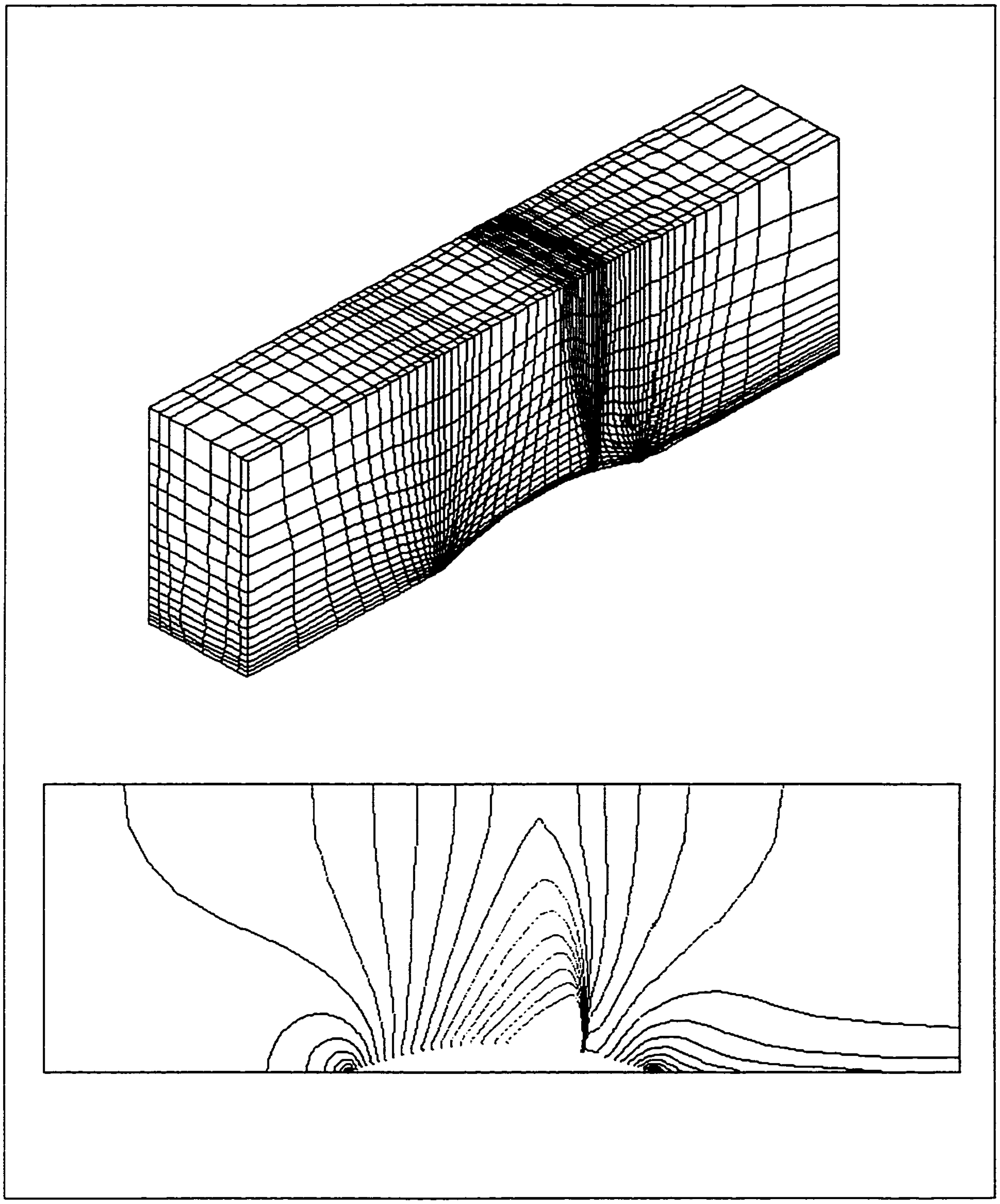


Figure 6.31 Transonic flow over 10% circular arc: adapted hexahedral grid and corresponding Mach contours

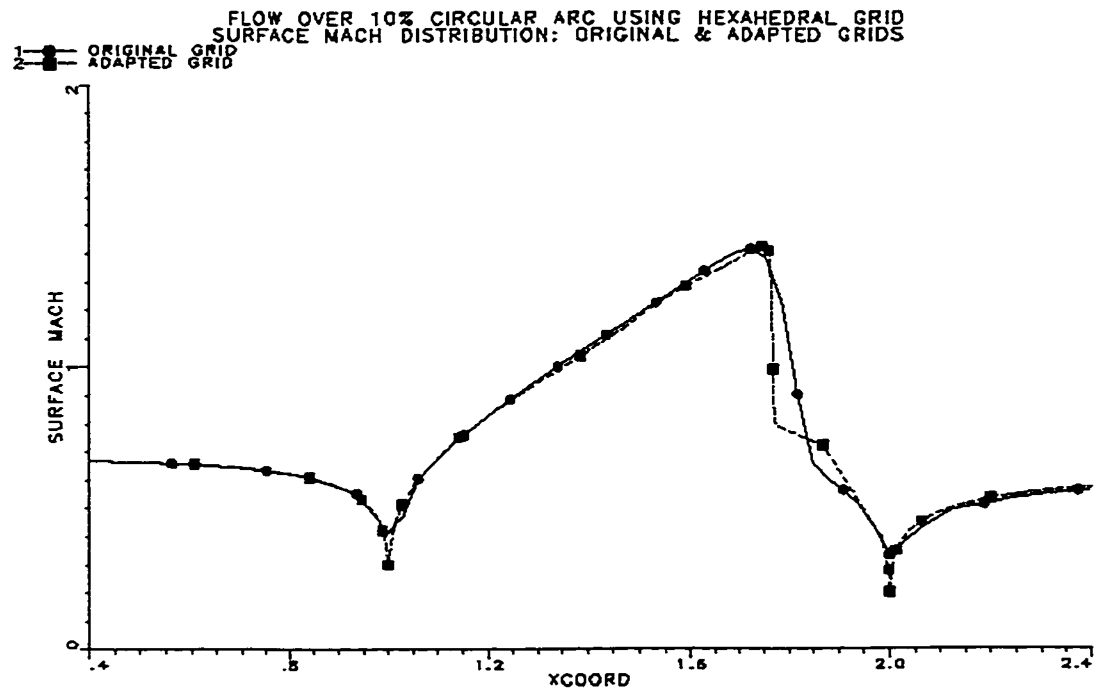


Figure 6.32(a) Surface Mach distribution for original and adapted hexahedral grids

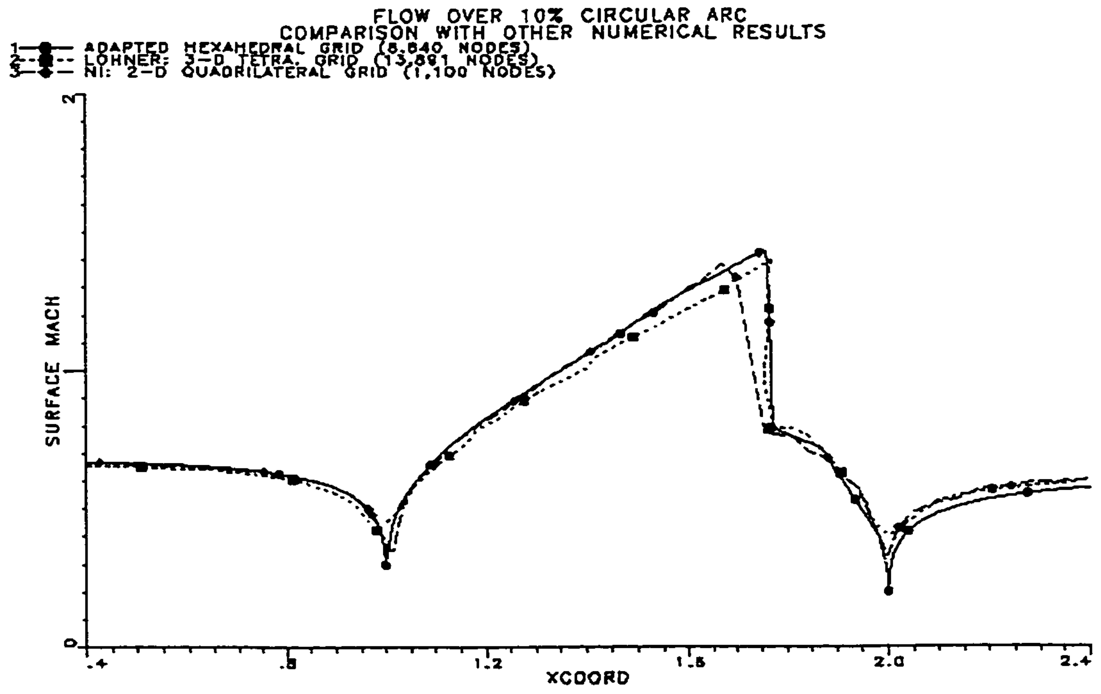


Figure 6.32(b) Comparison of adapted surface Mach distribution other numerical results (Löhner and Ni)

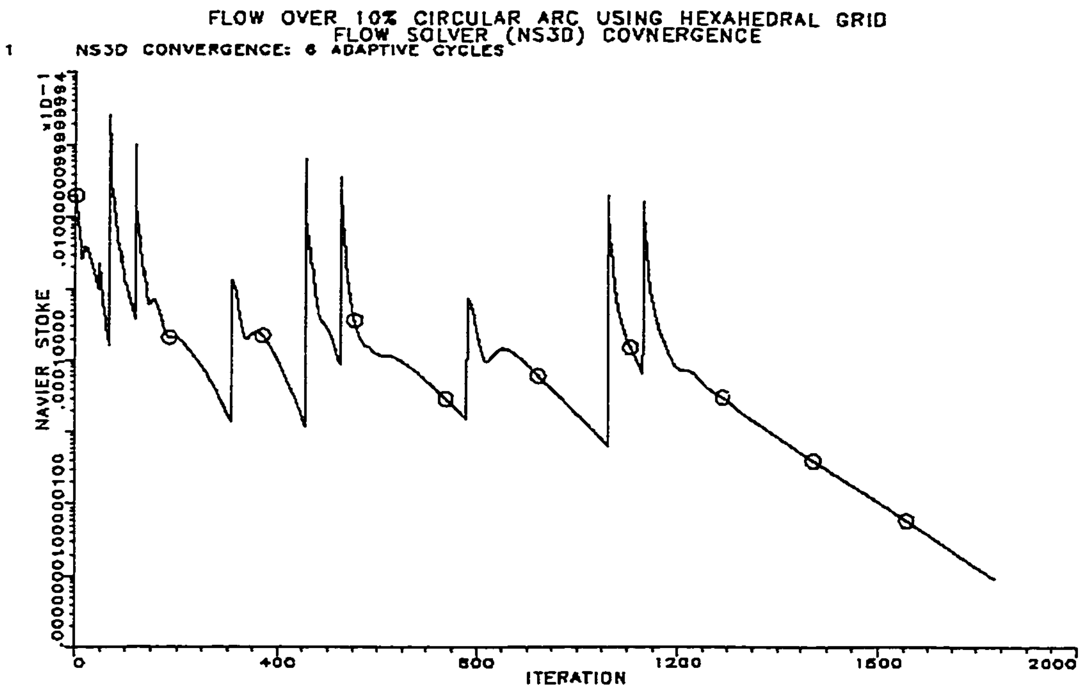


Figure 6.33 Flow solver convergence history over six adaptive cycles for transonic flow over 10% circular arc using hexahedral grid

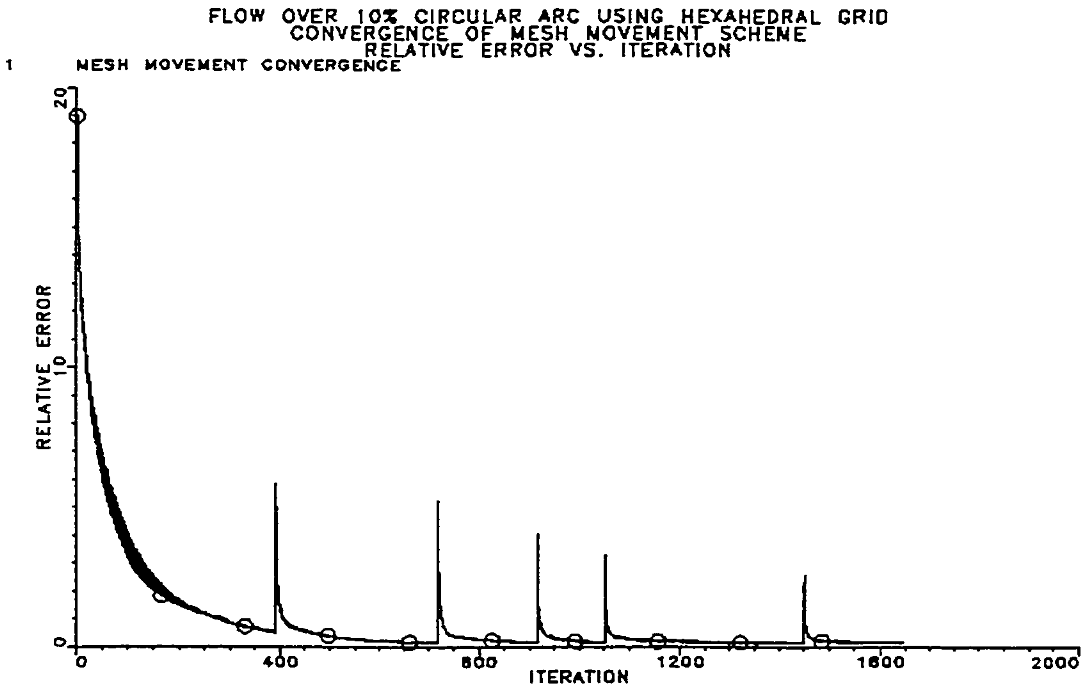


Figure 6.34 Adaptive algorithm convergence history over six adaptive cycles for transonic flow over 10% circular arc on hexahedral grid

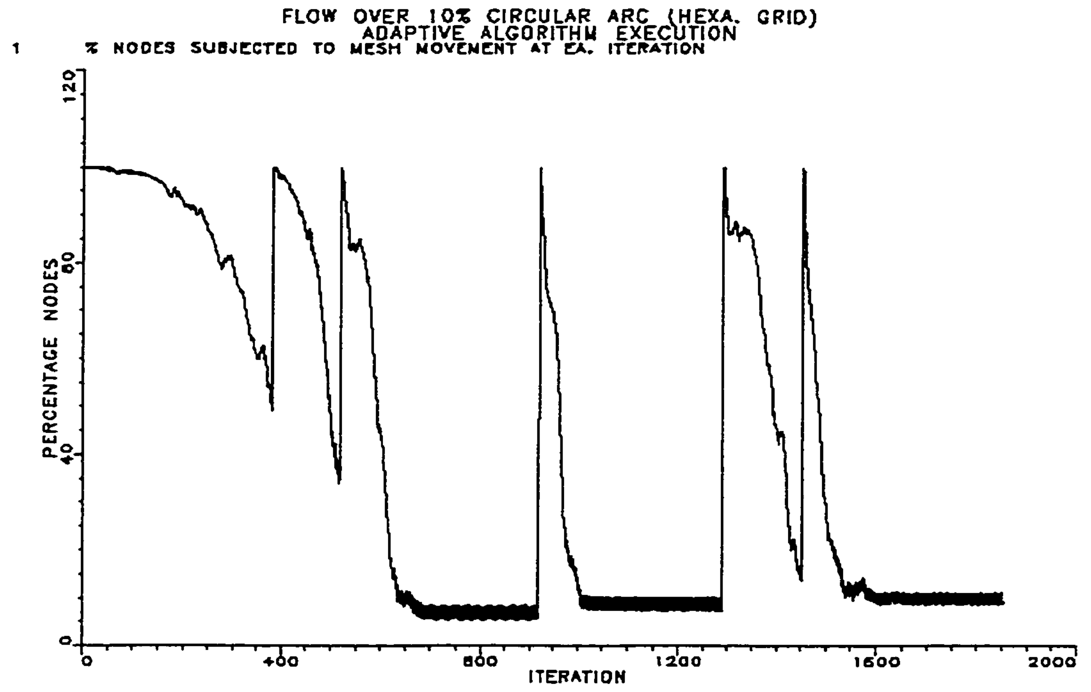


Figure 6.35 Percentage of nodes subjected to nodal displacement strategy for transonic flow over 10% circular arc on hexahedral grid

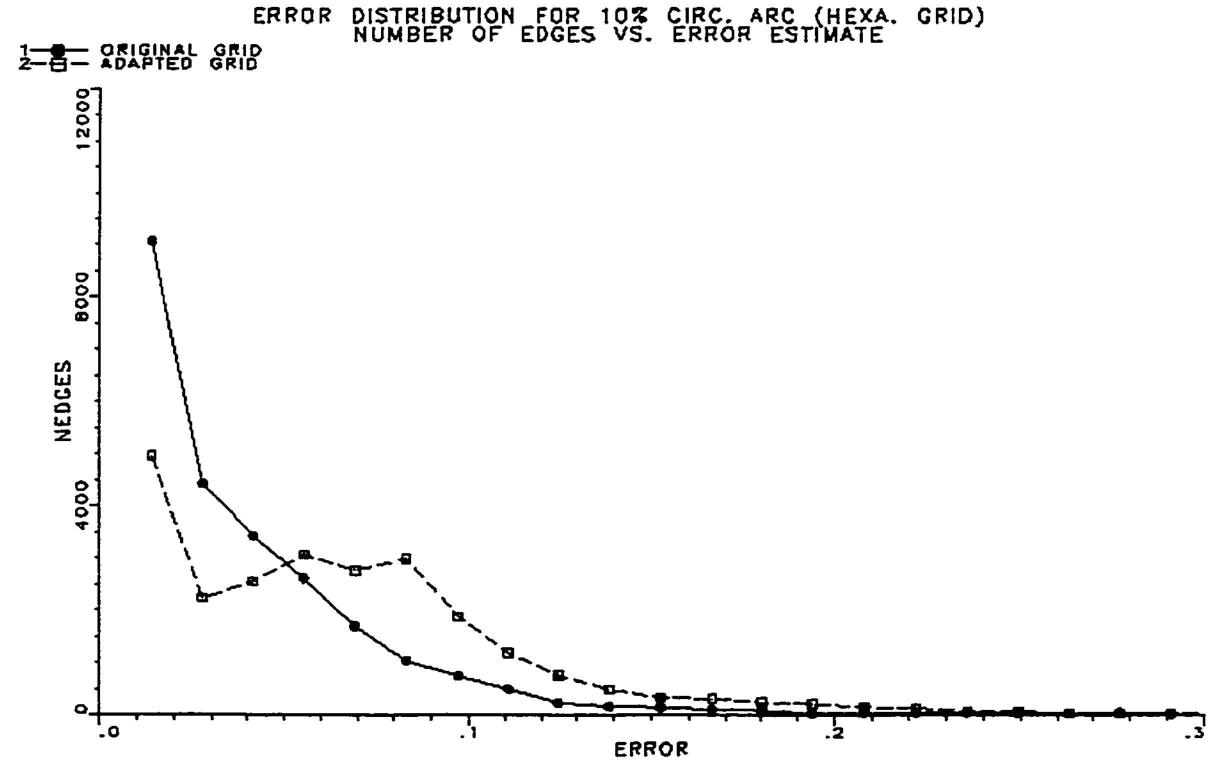


Figure 6.36 Error distribution over edges for original and adapted grids for transonic flow over 10% circular arc on hexahedral grid

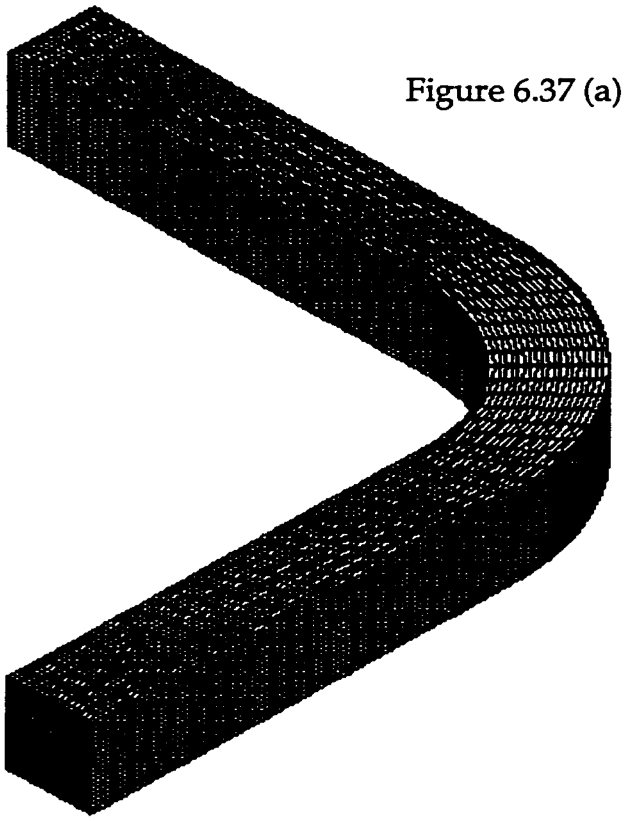


Figure 6.37 (a) Surface mesh of original 21x21x61 square duct grid

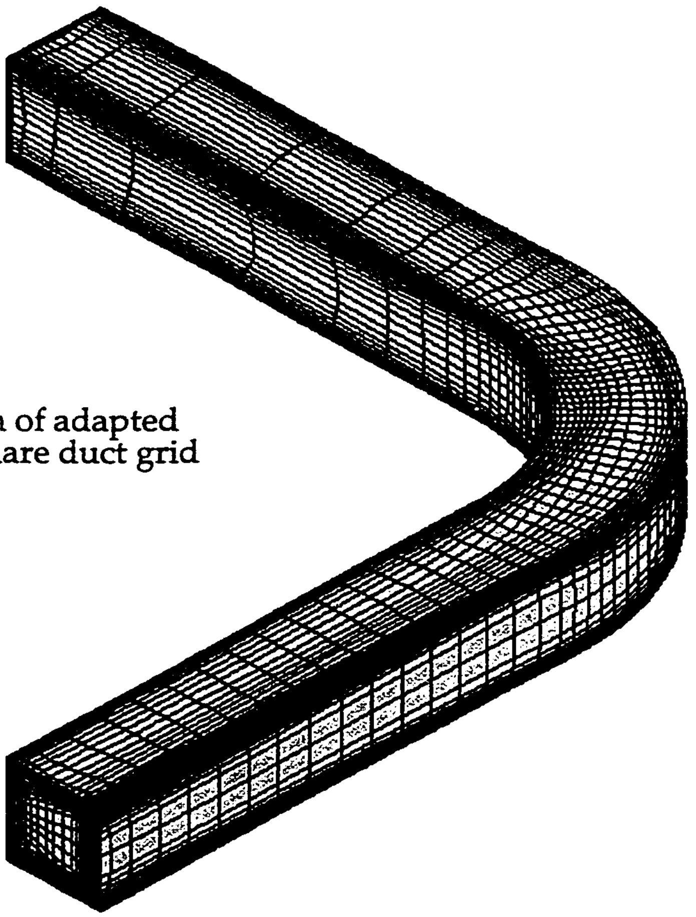


Figure 6.37 (b) Surface mesh of adapted 21x21x61 square duct grid

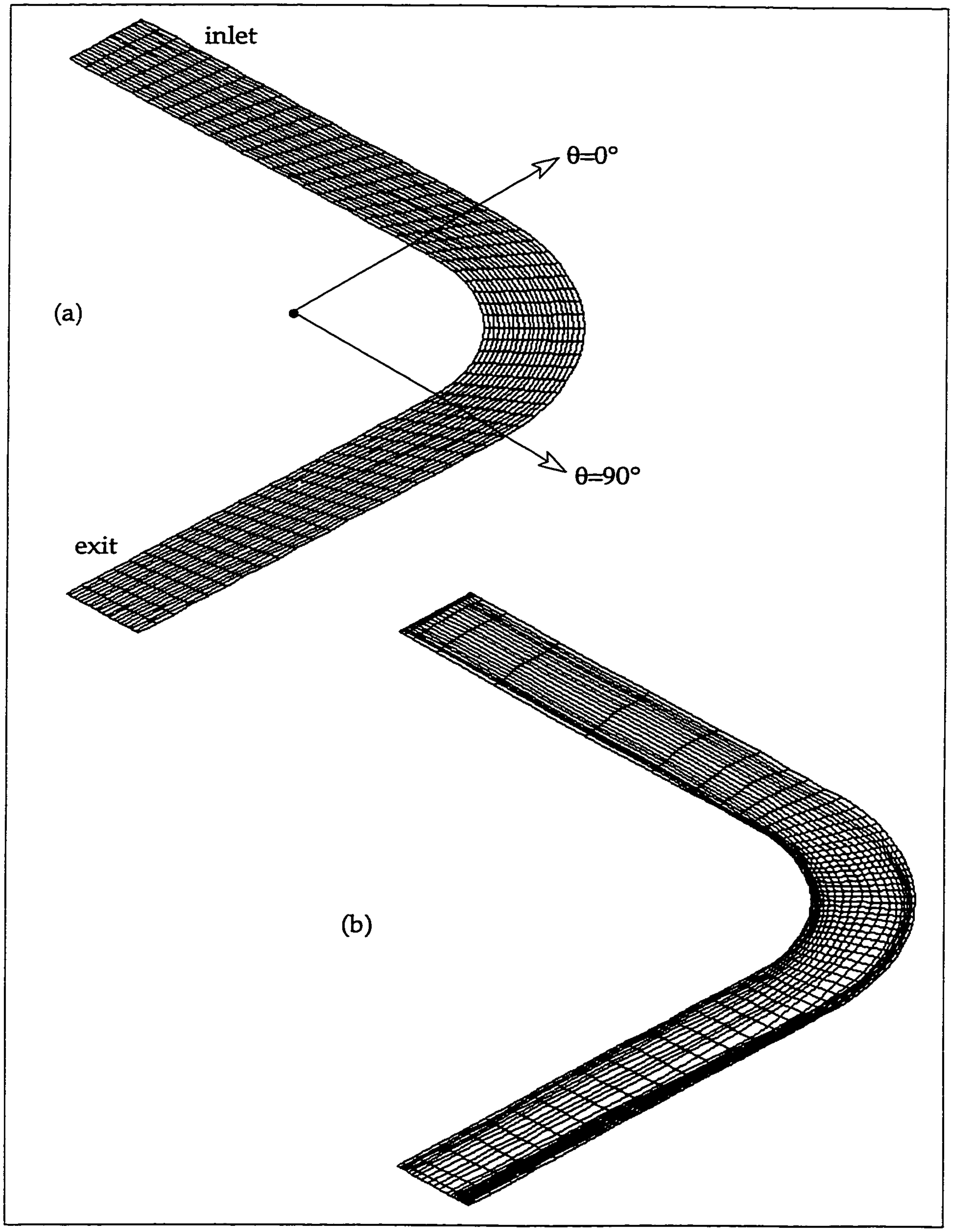


Figure 6.38 Original (a) and adapted (b) 21x21x61 grids at symmetry plane (y=0.5)

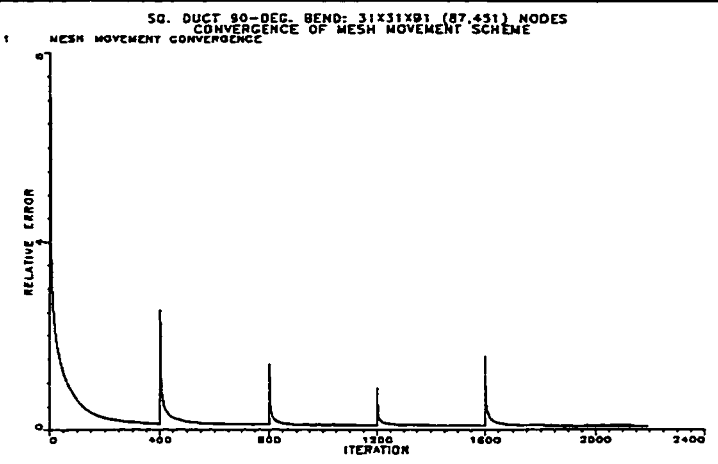


Figure 6.39 Adaptive algorithm convergence history over six adaptive cycles for 90-deg bend

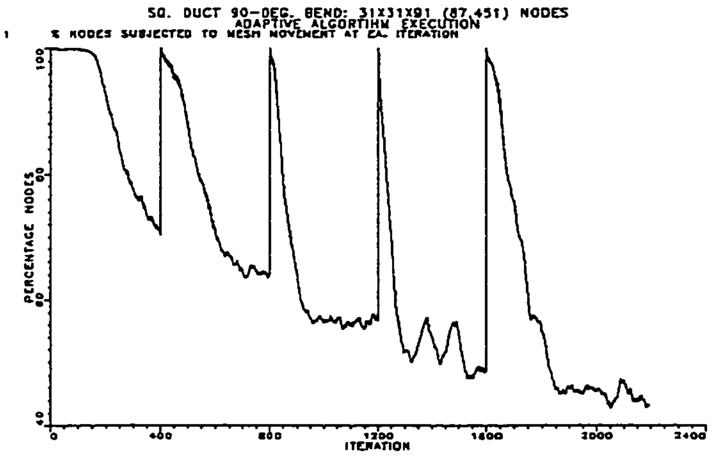


Figure 6.40 Percentage of nodes subjected to nodal displacement strategy for 90-deg bend

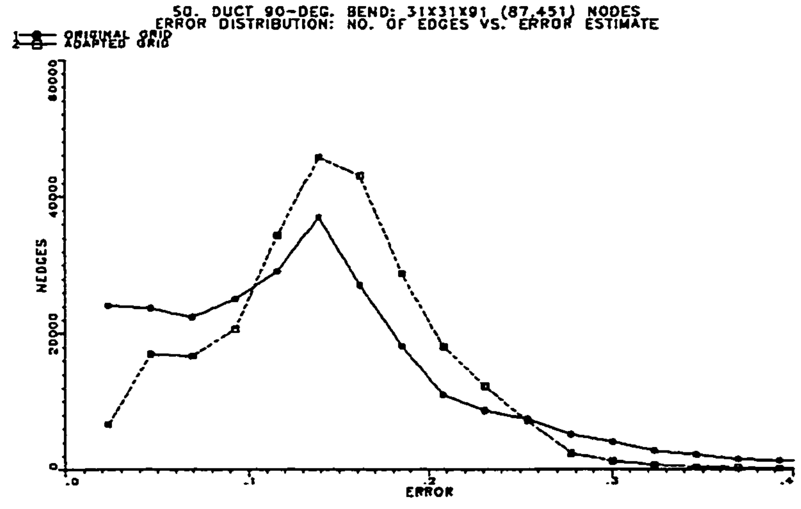
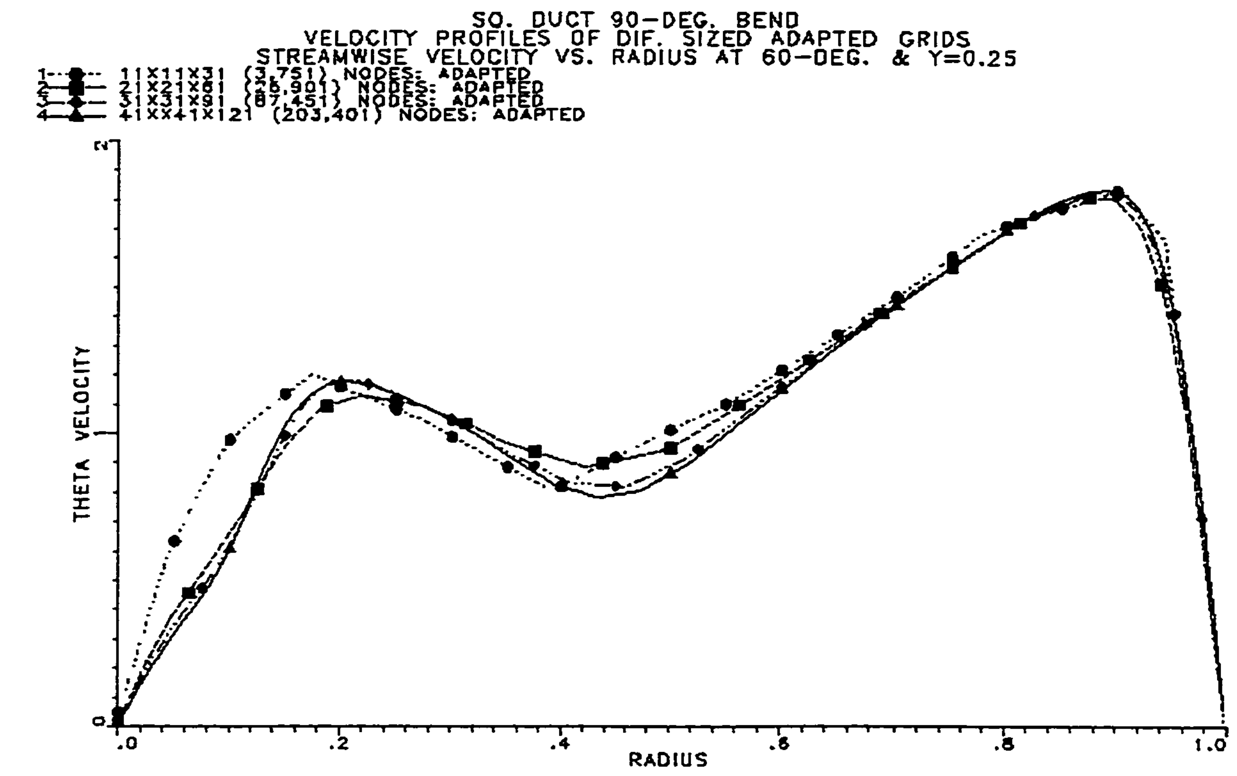


Figure 6.41 Error distribution over edges for original and adapted grids for 90-deg bend



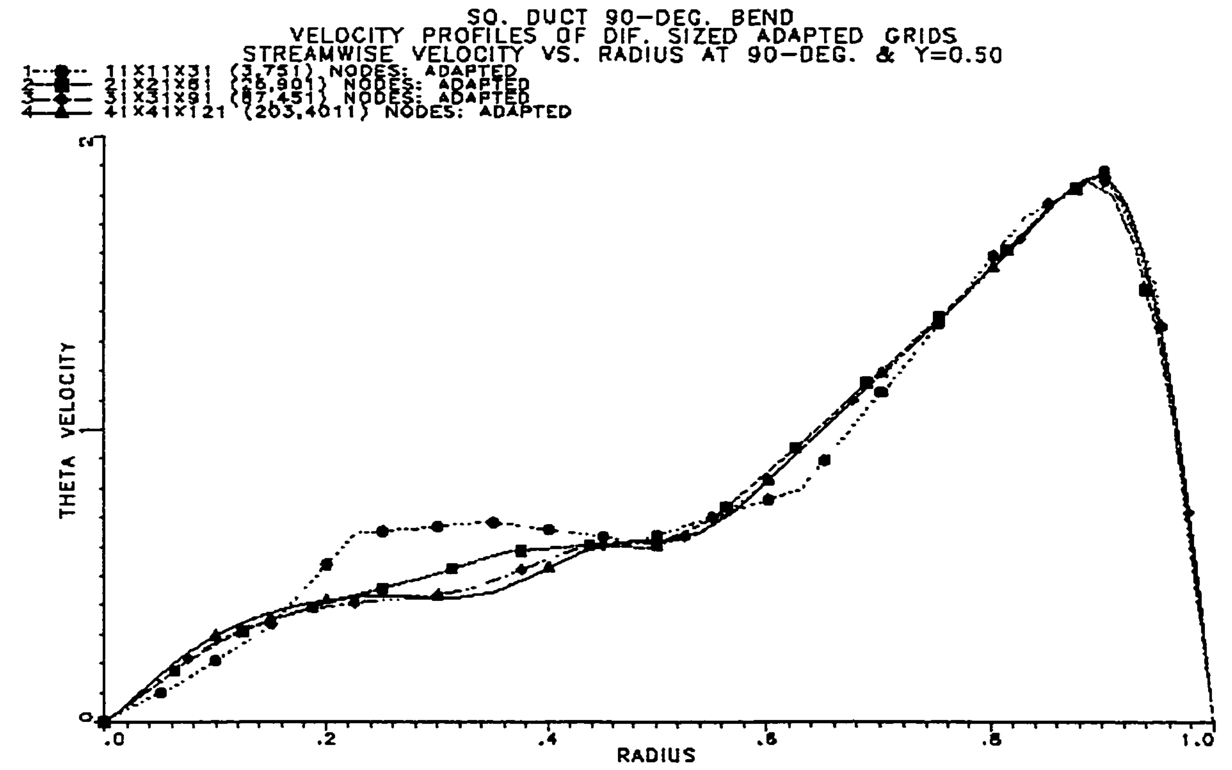
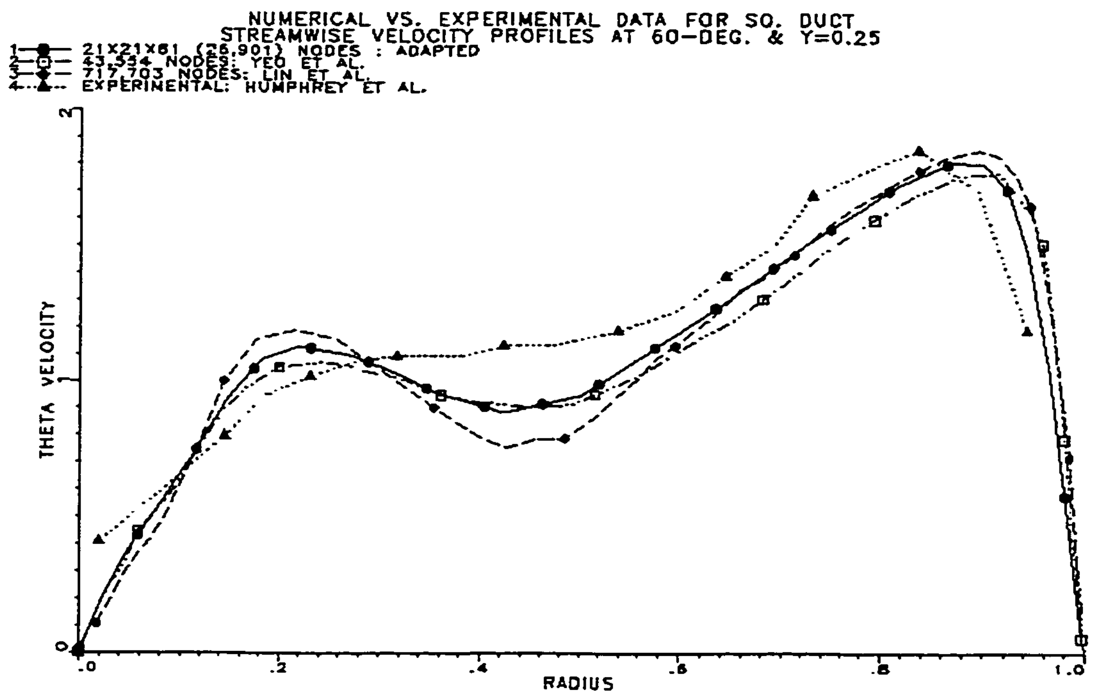


Figure 6.42 Comparison of streamwise velocity profiles for adapted grids of different sizes at (a) 60° & y=0.25 plane (b) 90° & y=0.50 plane



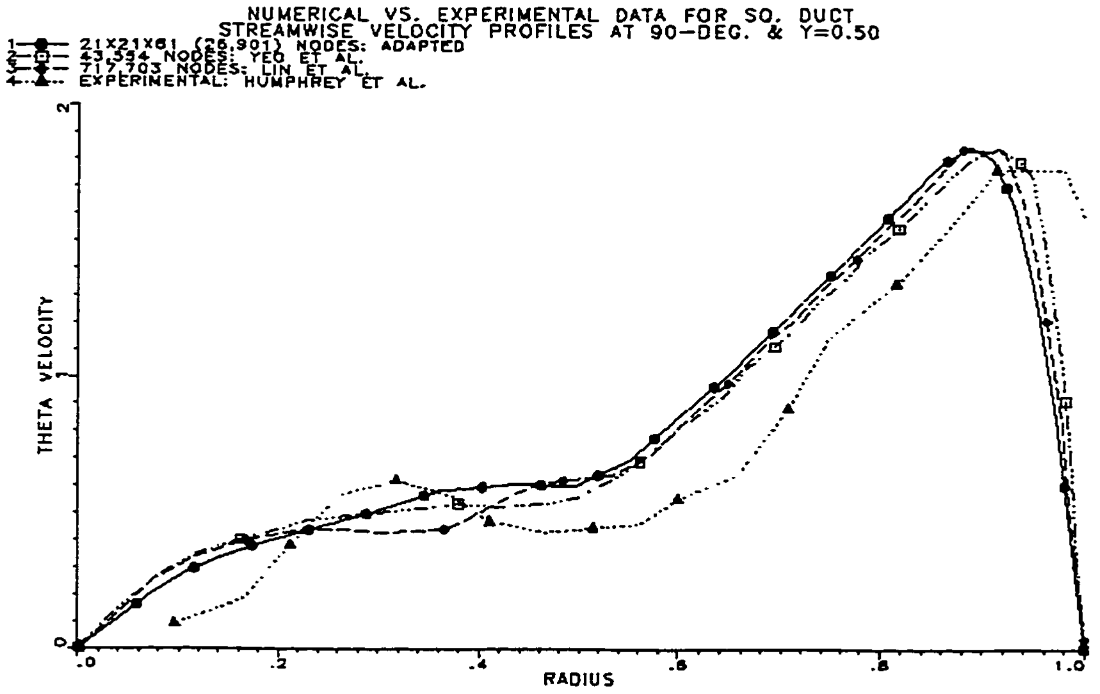


Figure 6.43 Comparison of streamwise velocity profile on 21x21x61 adapted grid with that of experimental and other numerical cases at (a) 60° and y=0.25 plane and (b) 90° and y=0.50 plane

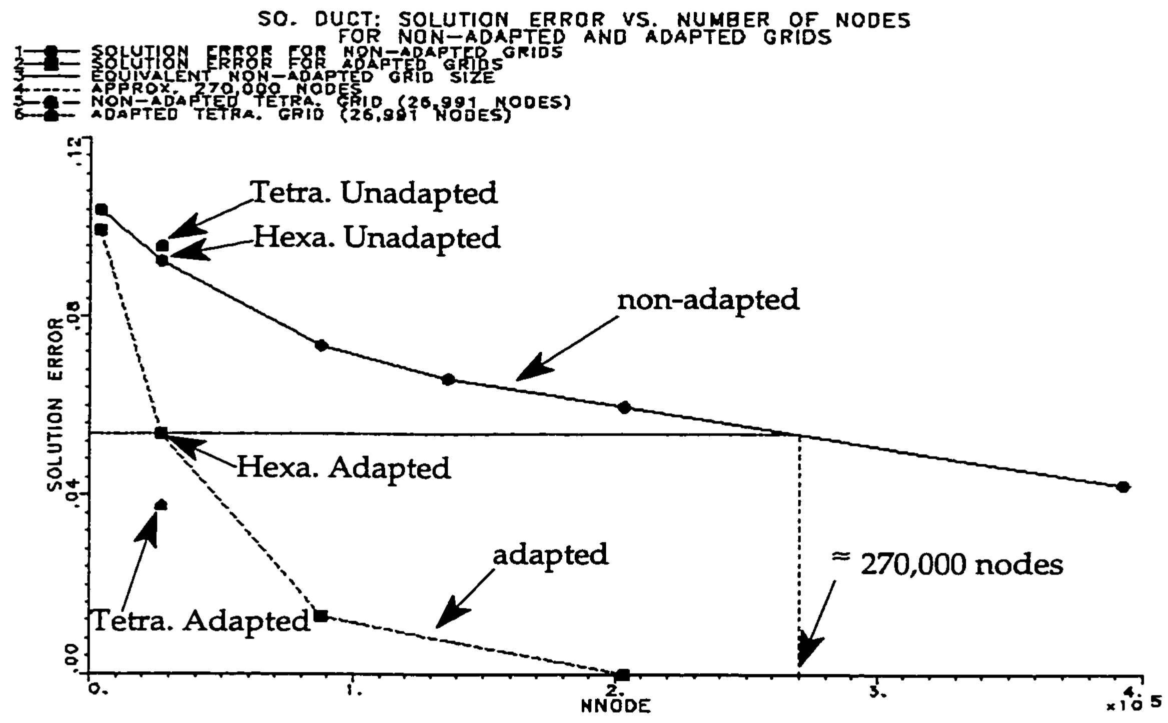
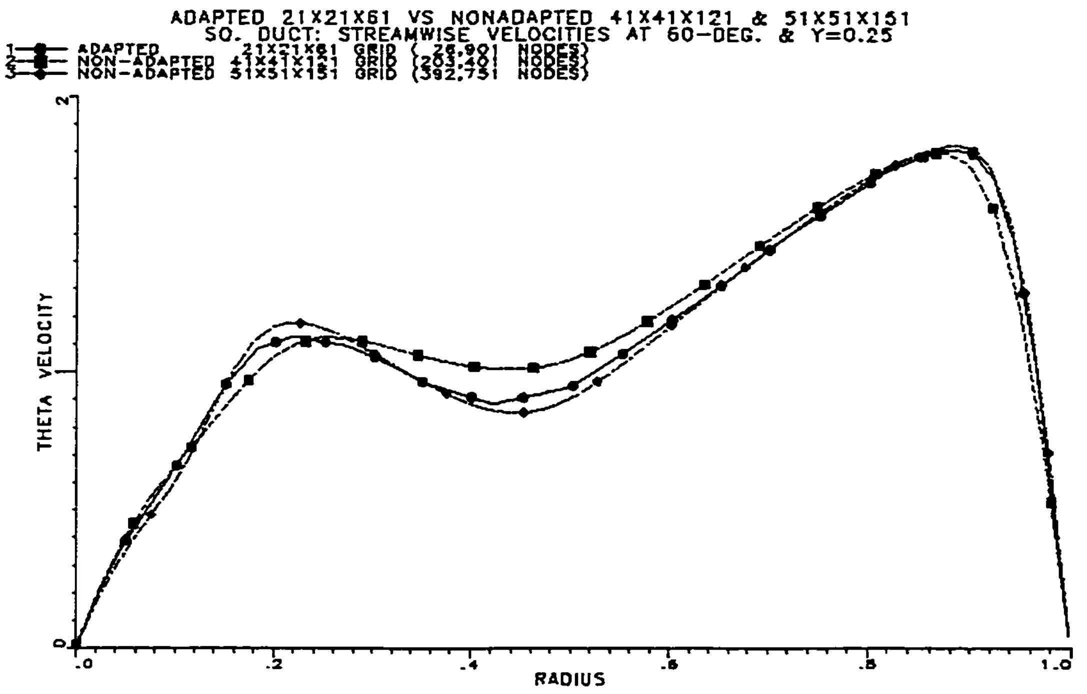


Figure 6.44 Plot of solution error versus grid size for non-adapted and adapted grids for square duct with 90° bend.



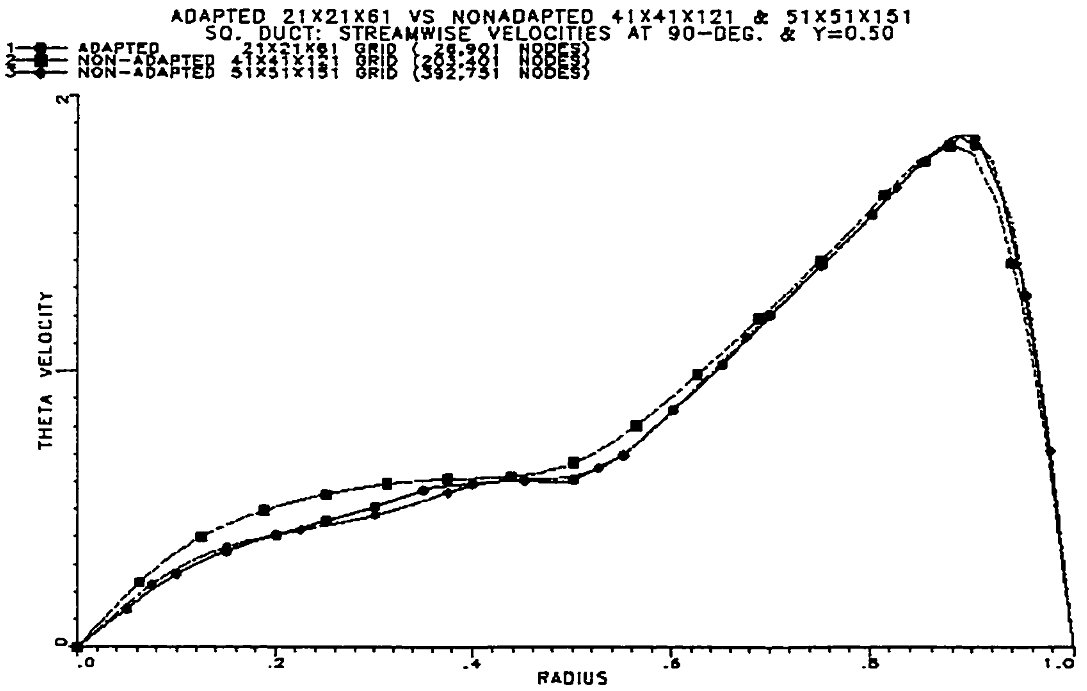


Figure 6.45 Comparison of streamwise velocities on adapted 21x21x61 grid and non-adapted 41x41x121 and 51x51x151 grids at (a) 60° and y=0.25 plane and (b) 90° and y=0.50 plane

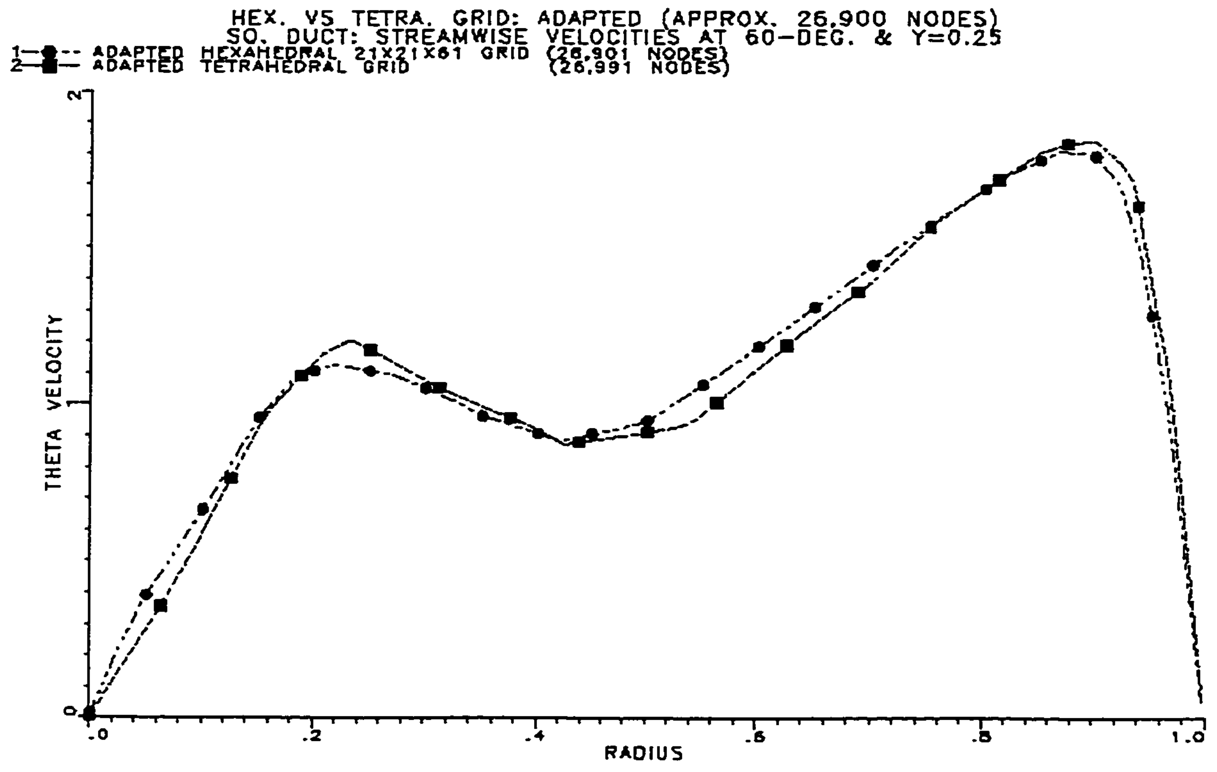


Figure 6.46 Comparison of streamwise velocities at 60° and y=0.25 on adapted hexahedral (26,901 nodes) and tetrahedral (26,991 nodes) grids

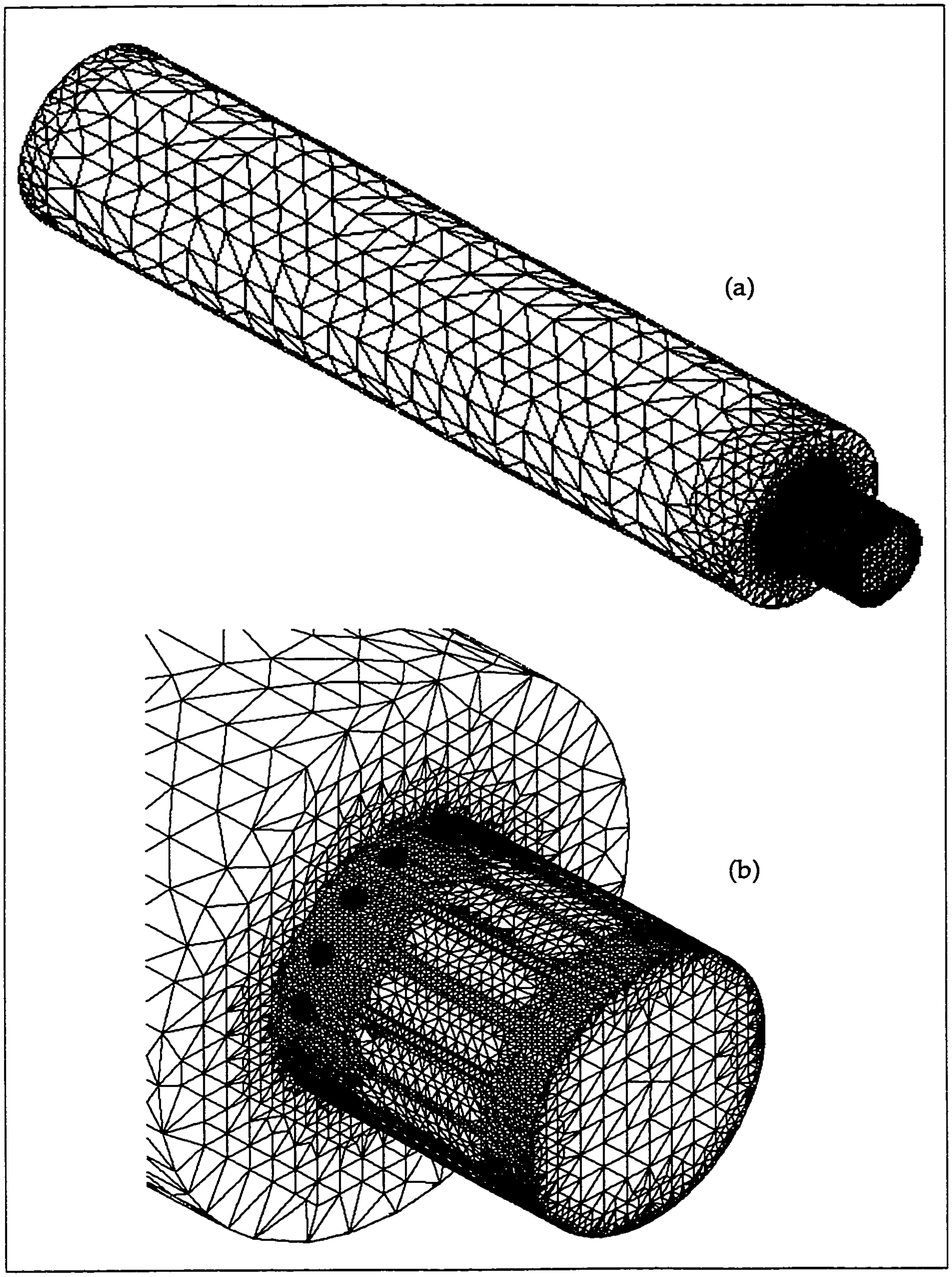


Figure 6.47 (a) Original surface mesh of hydrogen can combustor
(b) Magnified view of original surface mesh of swirler

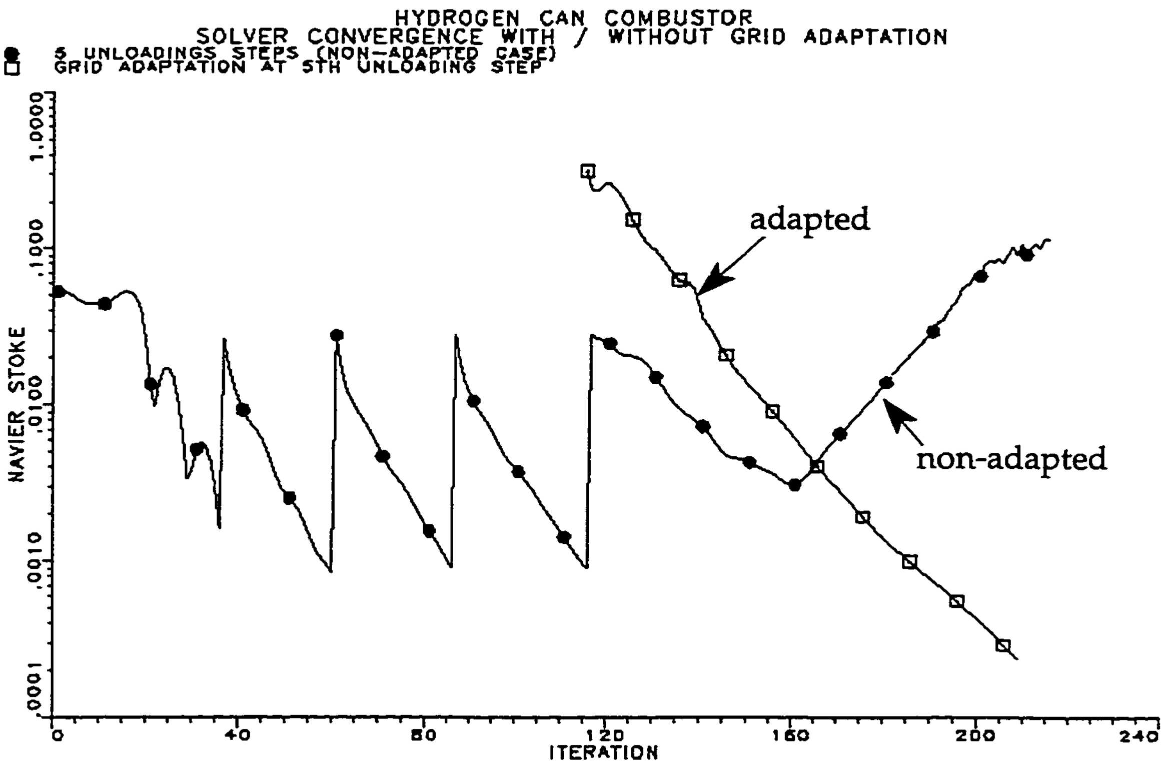
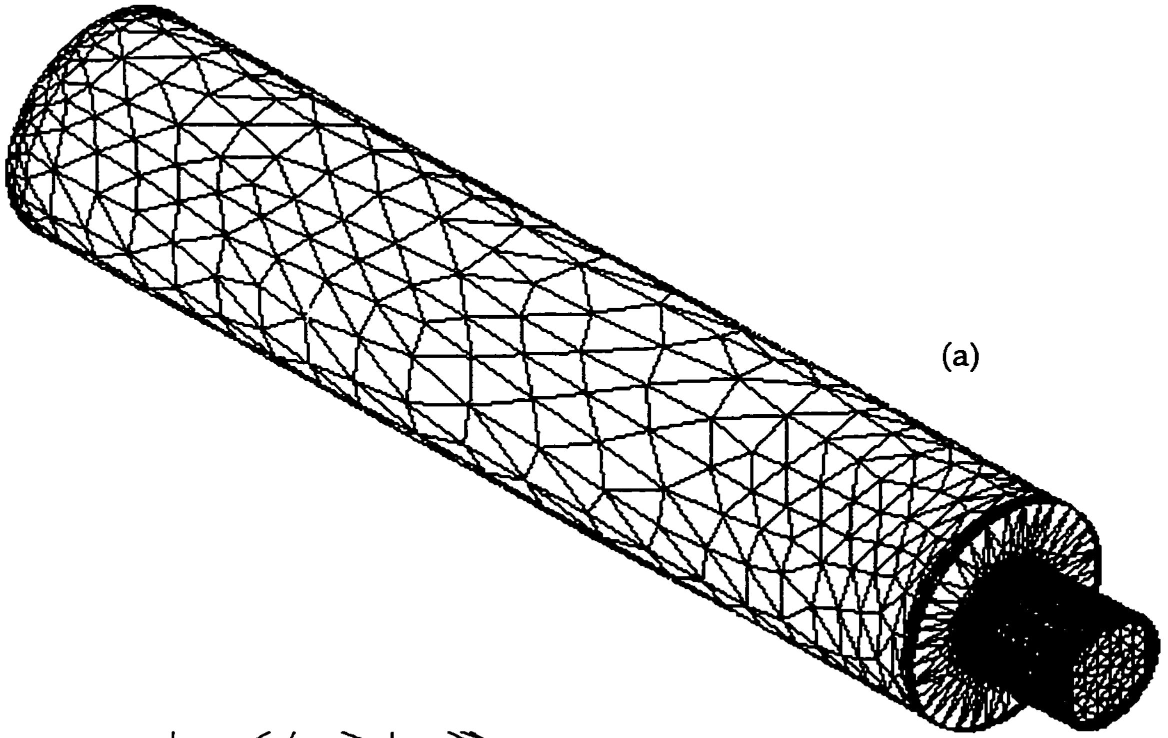


Figure 6.48 Flow solver convergence history with and without adaptive mesh movement for hydrogen can combustor test case



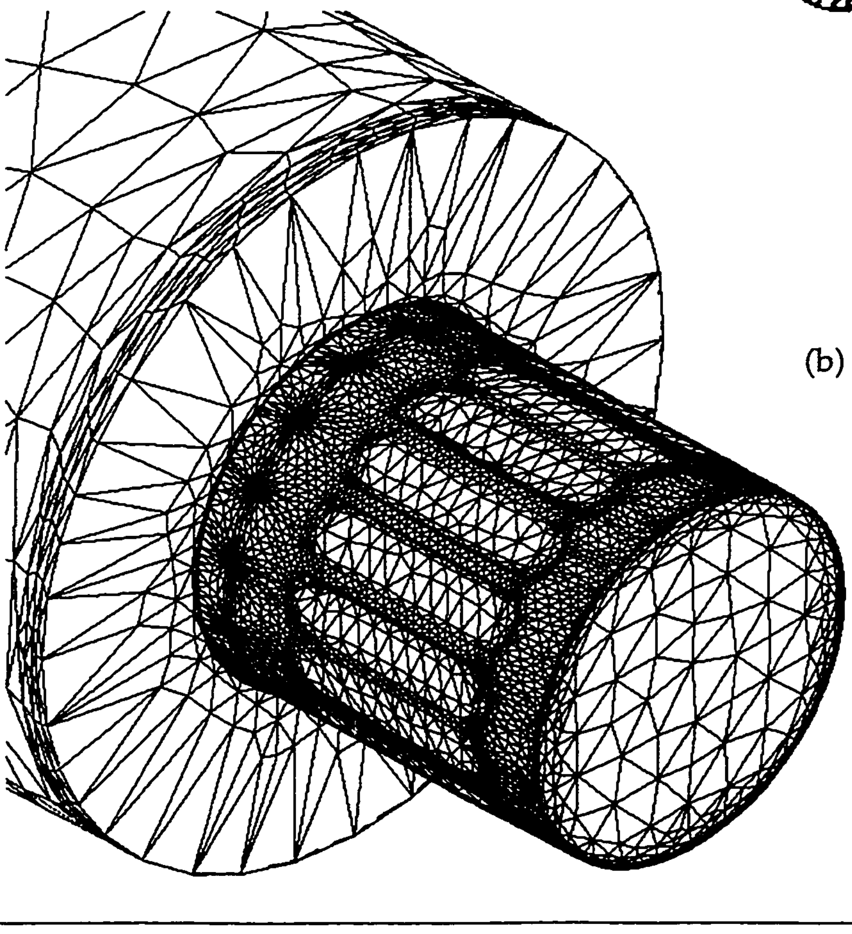


Figure 6.49 (a) Adapted surface mesh of hydrogen can combustor
(b) Magnified view of adapted surface mesh of swirler

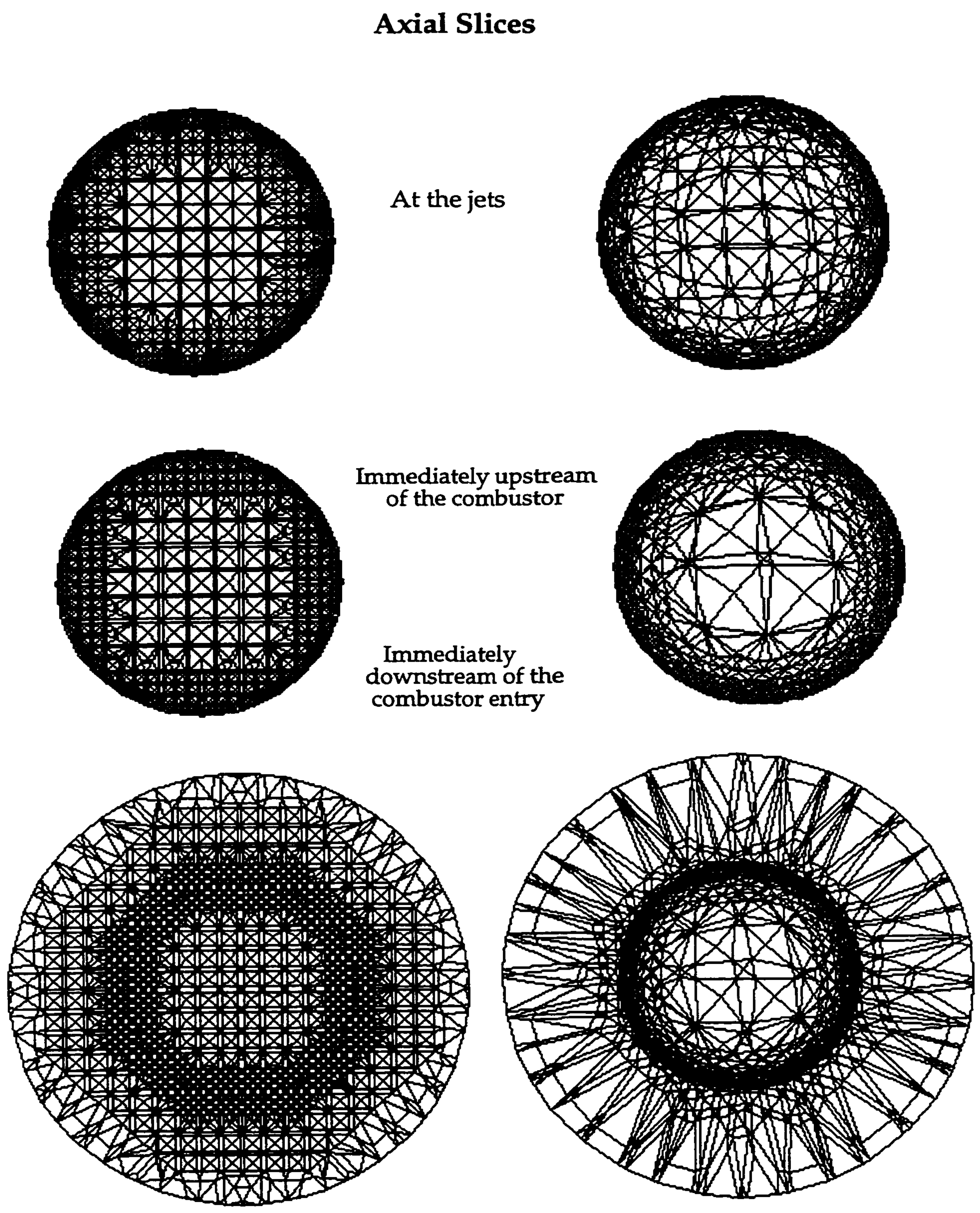

Axial Slices
At the jets
Immediately upstream of the combustor
Immediately downstream of the combustor entry

Figure 6.50 Comparison of original and adapted grids at various axial positions

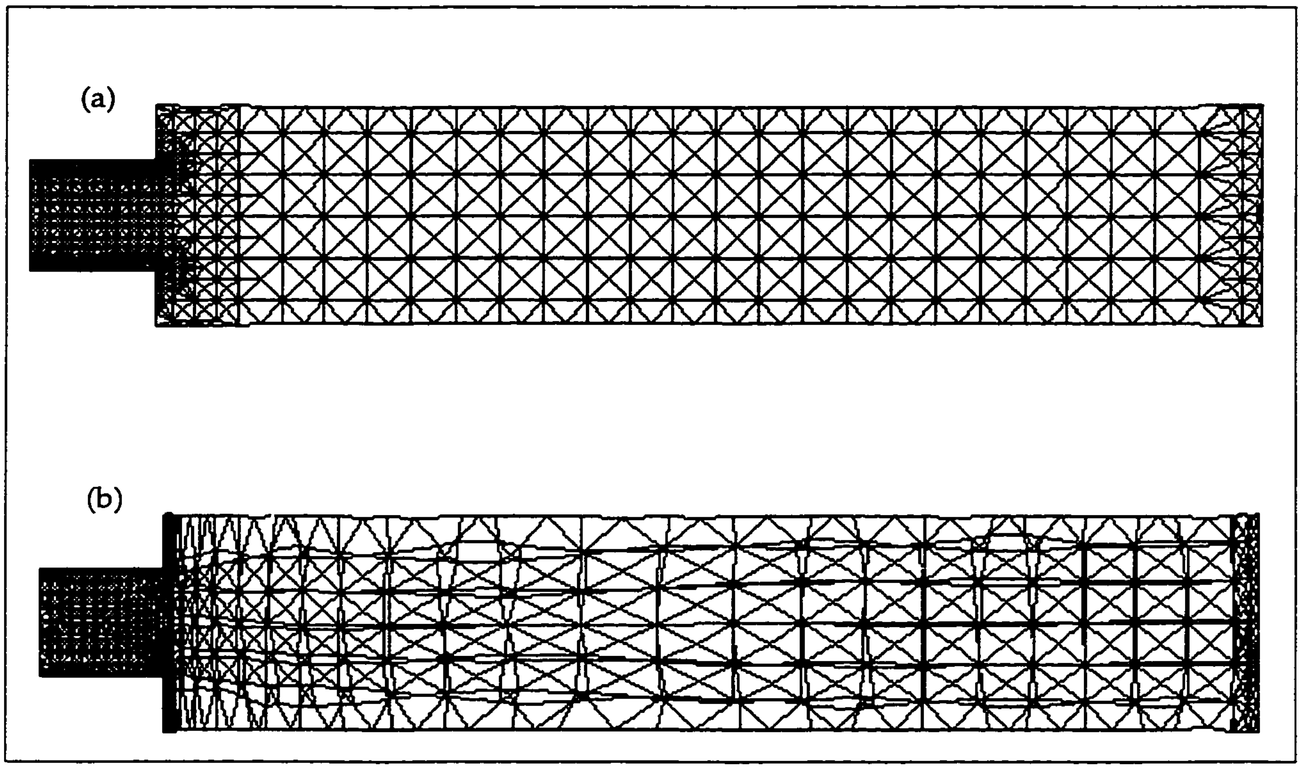


Figure 6.51 Longitudinal view of (a) original and (b) adapted grids of hydrogen combustor

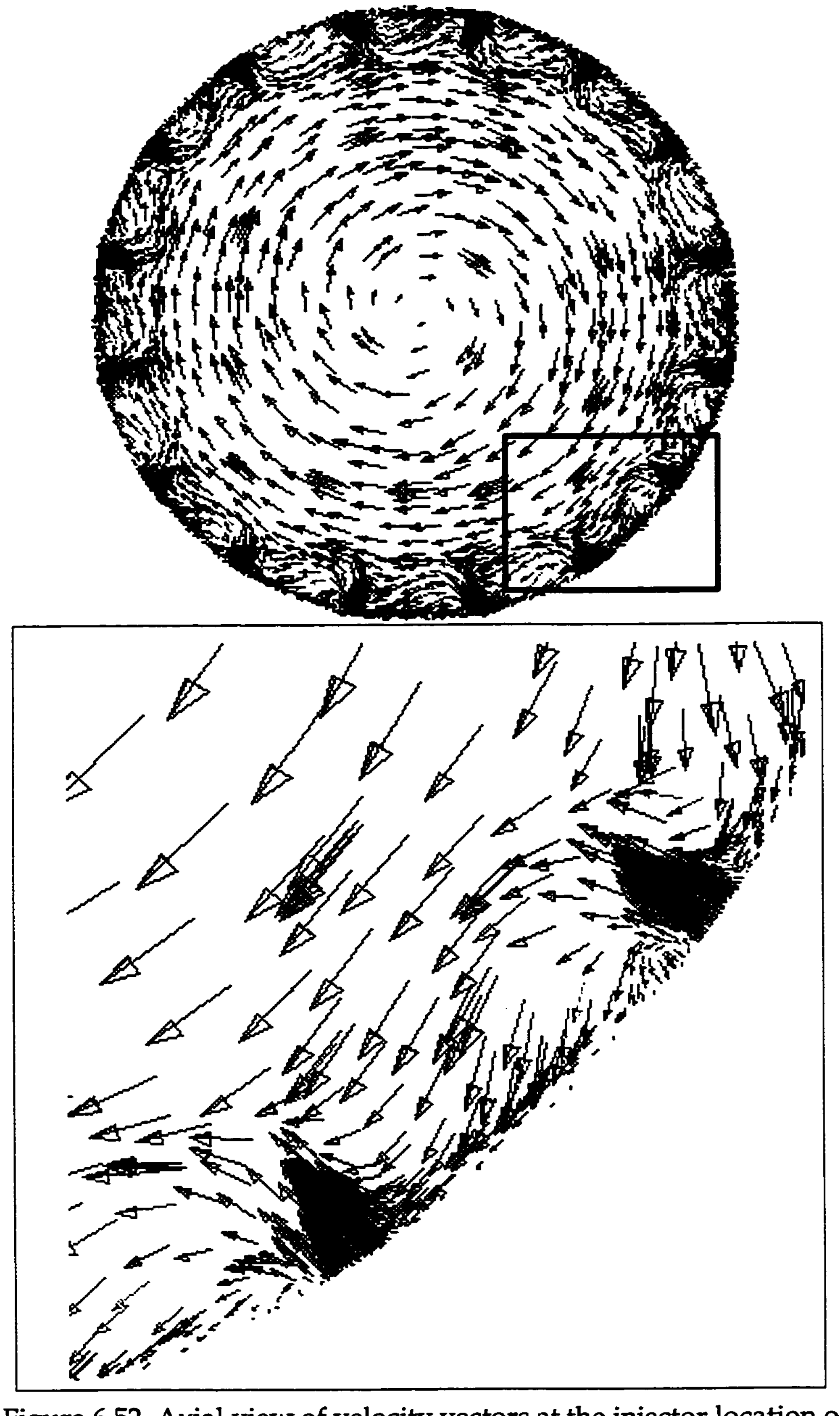


Figure 6.52 Axial view of velocity vectors at the injector location of hydrogen combustor

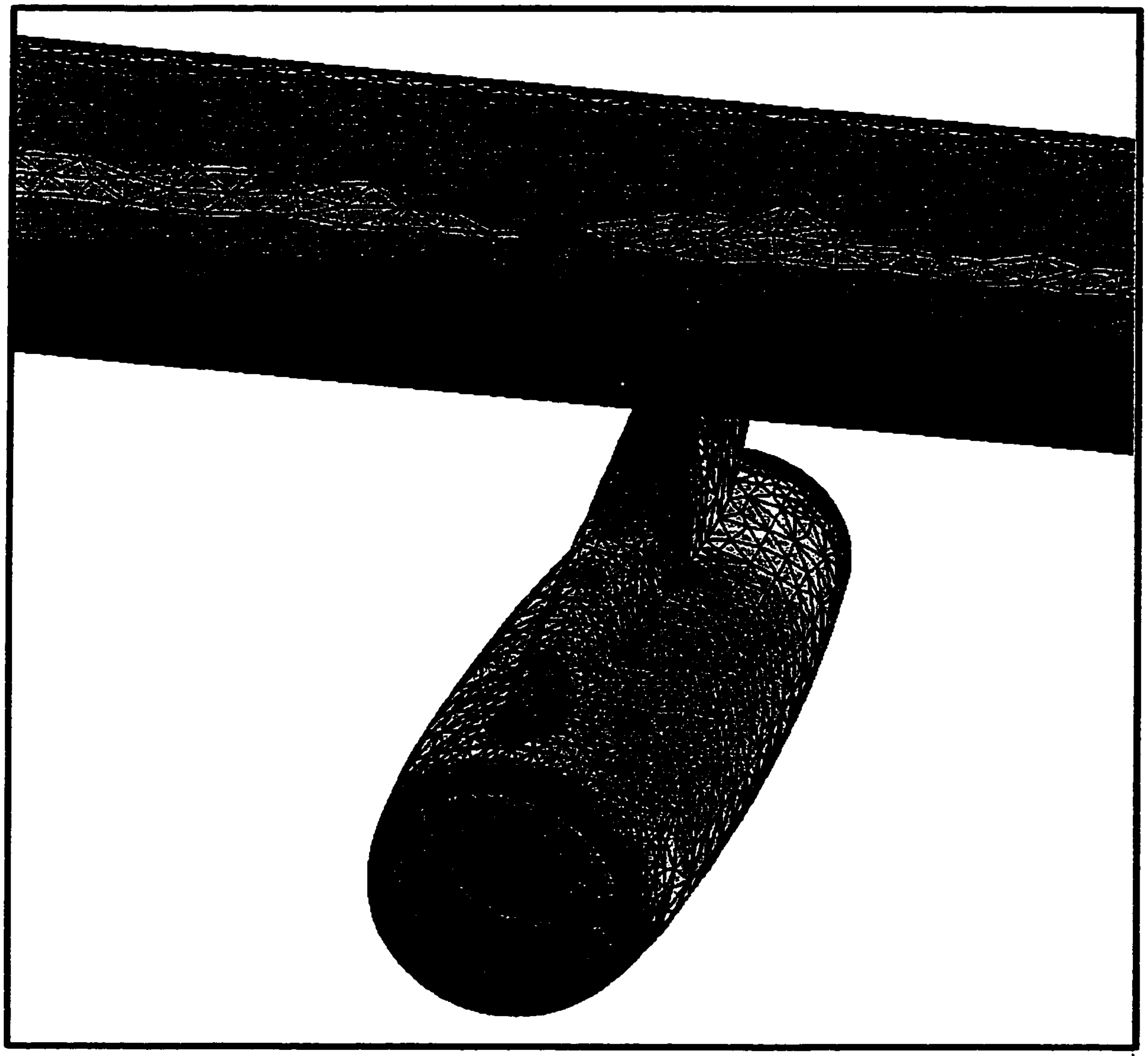


Figure 6.53 Wing-Pylon-Nacelle original grid

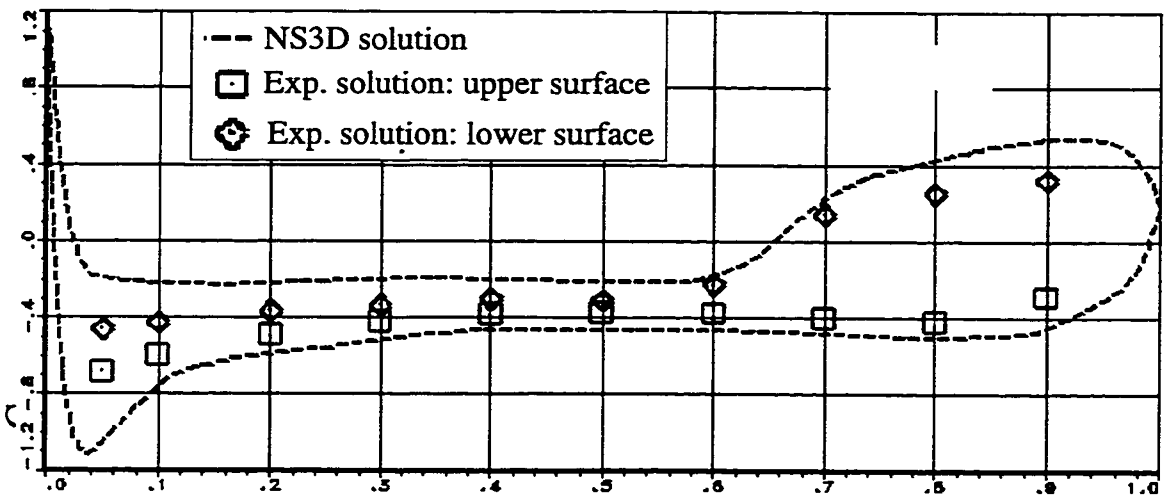


Figure 6.54 Cp distribution on wing far from nacelle using original grid

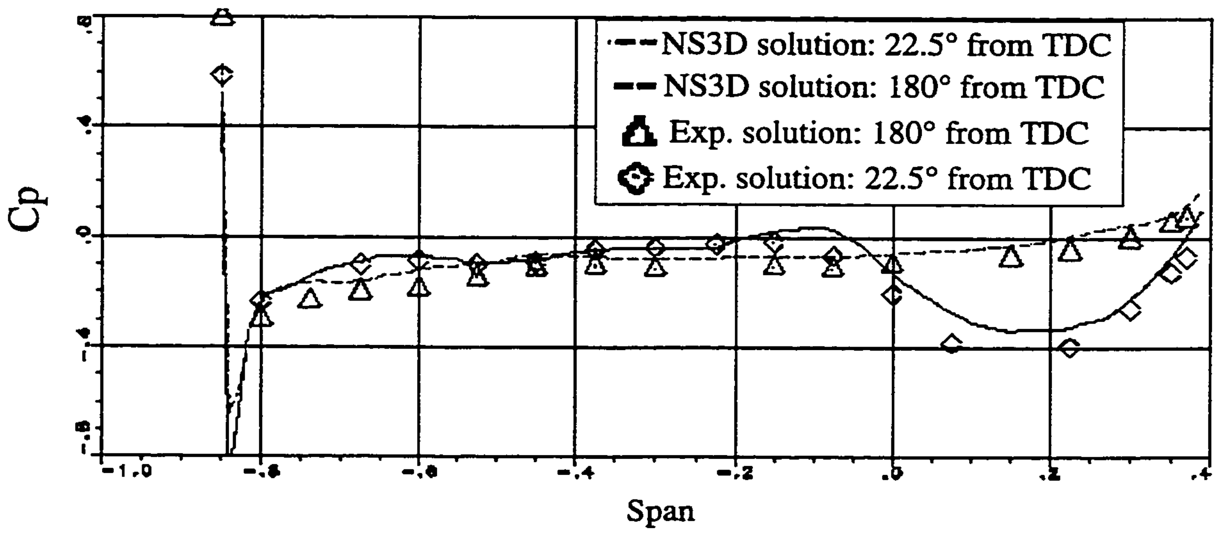


Figure 6.55 Cp distribution on nacelle using original grid

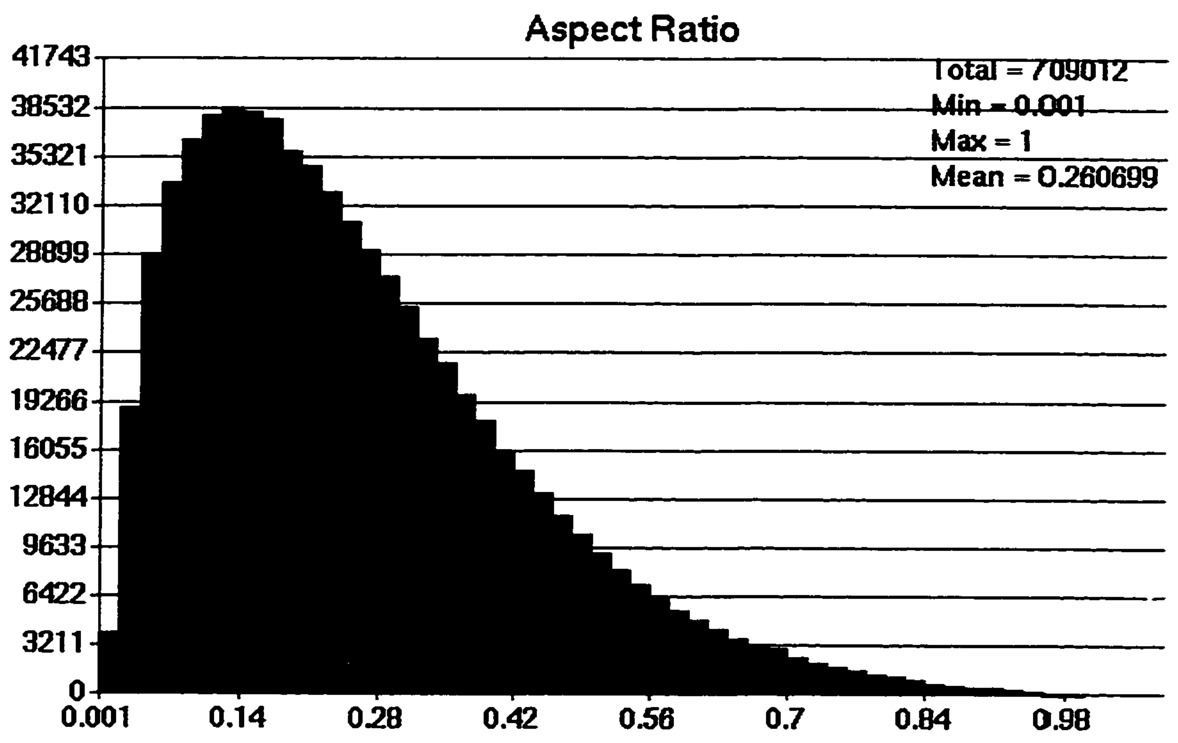


Figure 6.56 Aspect ratio histogram of adapted grid for wing-pylon-nacelle (specified minimum aspect ratio = 0.001)

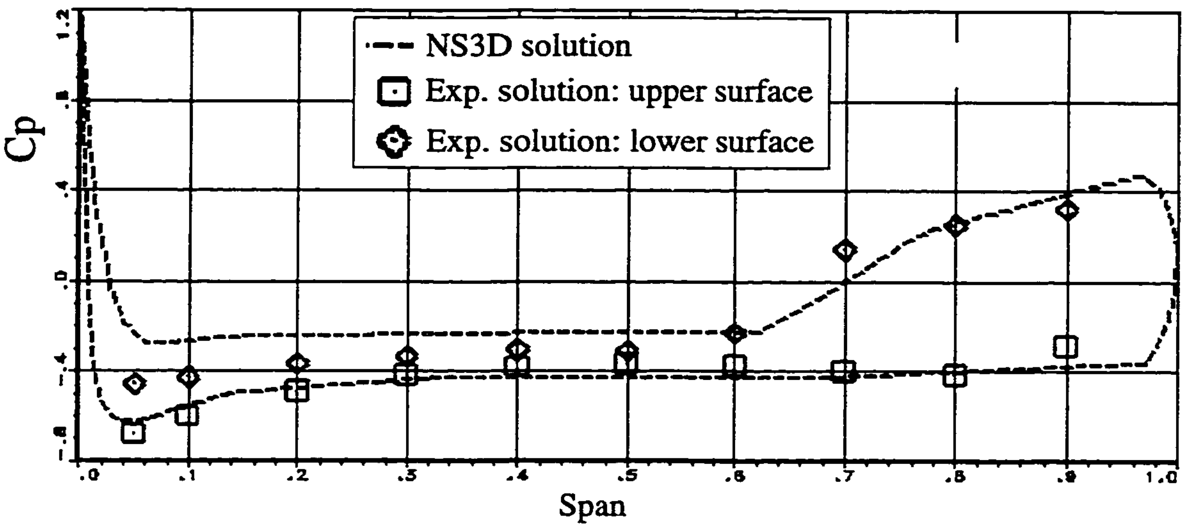


Figure 6.57 Cp distribution on wing far from nacelle using adapted grid

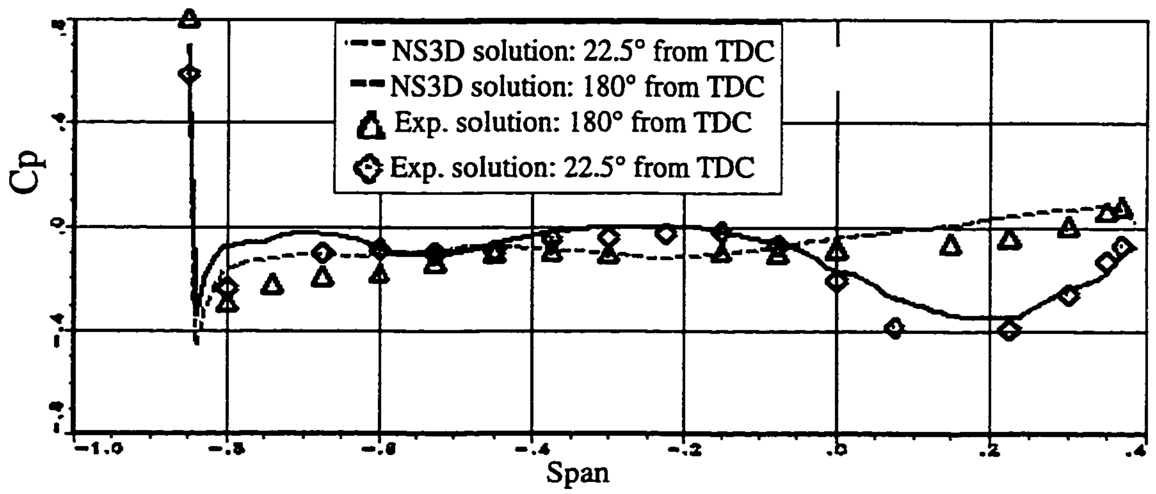


Figure 6.58 Cp distribution on nacelle using adapted grid

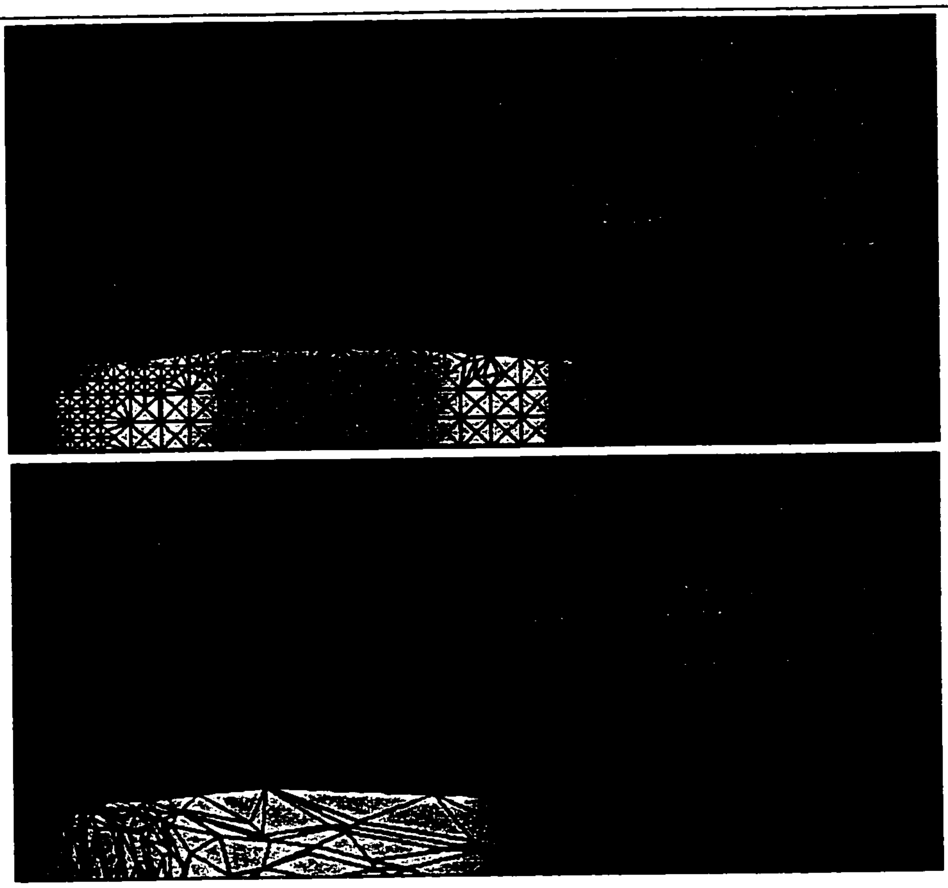


Figure 6.59 Views of (a) original and (b) adapted grids with corresponding Mach number distributions at pylon cutting plane

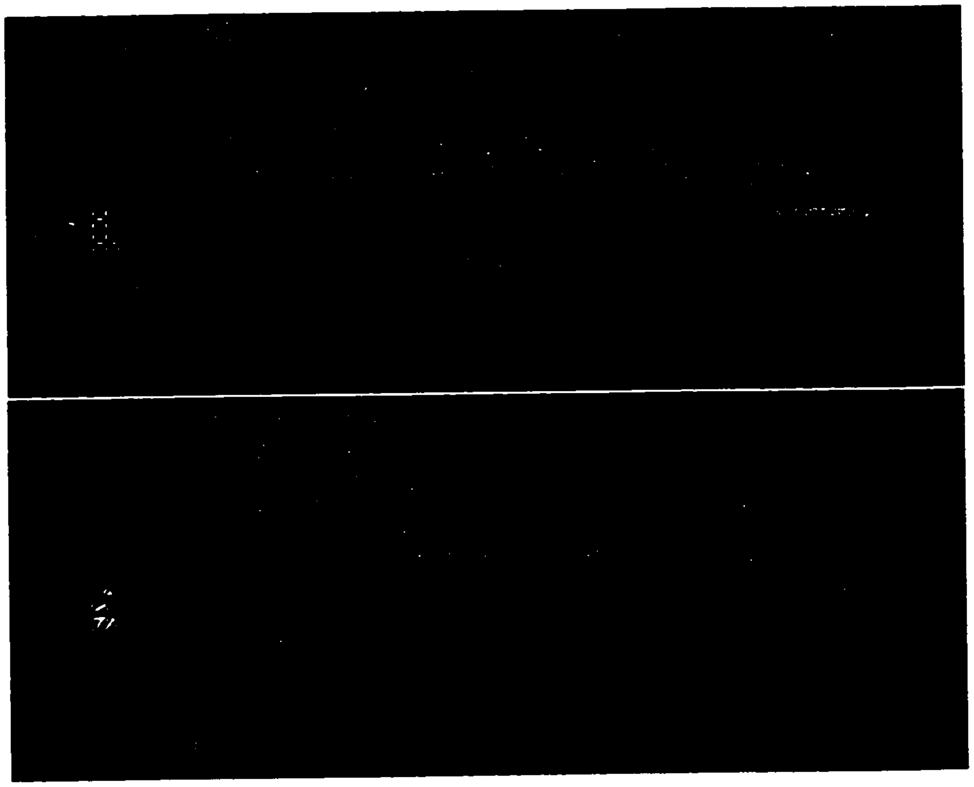
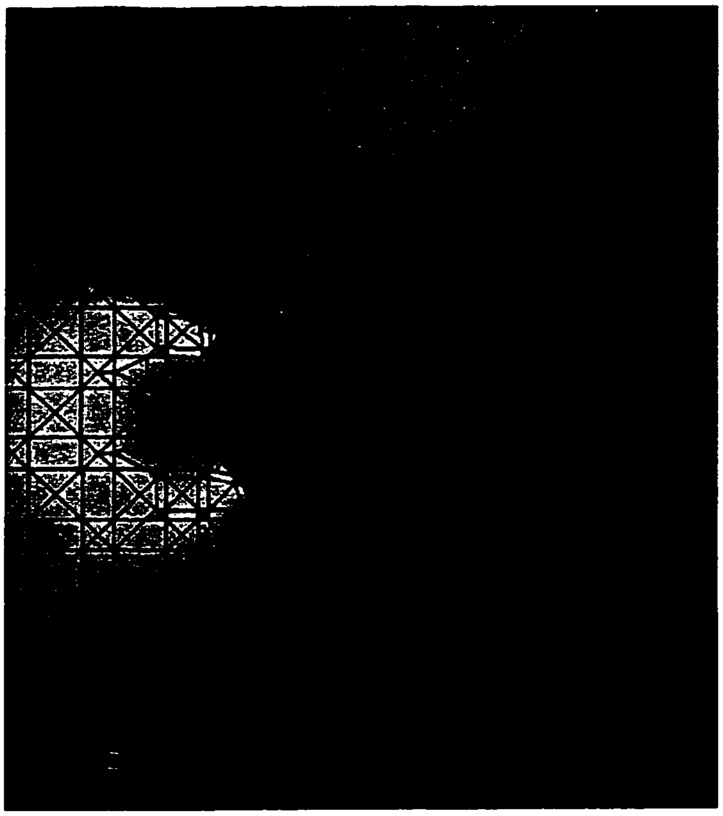


Figure 6.60 Views of (a) original and (b) adapted grids with corresponding Mach number distributions at wing cutting plane


(a)

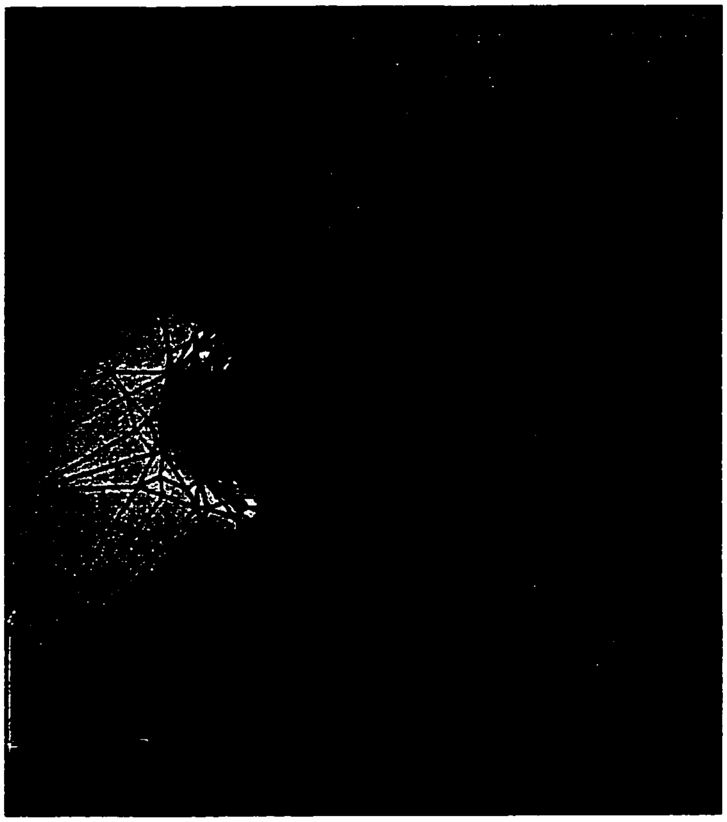

(b)

Figure 6.61 Views of (a) original and (b) adapted grids and corresponding Mach number distributions at wing leading edge

7. Discussion

7.1 Conclusions

An anisotropic adaptive grid method has been described for the solution of three-dimensional inviscid and viscous flows by the finite element method. The adaptive procedure has been validated on both tetrahedral and hexahedral grids. The method consists of an edge-based error estimate and an improved mesh movement strategy. The interpolation error of the numerical solution at each point in the domain is a function of its Hessian matrix. The error estimate is built from a modified positive-definite form of the Hessian tensor. The resulting error (Riemannian) metric tensor controls the magnitude as well as the direction of the grid stretching. The desired anisotropic mesh is constructed as the transform of an isotropic mesh by a coordinate transformation based on this tensor. In fact, the edge-based error estimate is the precisely the length of the edge in the Riemannian metric and may be interpreted as a 1-D interpolation error estimate projected on the edge. The approach taken in this research seeks a near isotropic mesh in the transformed metric space and an equidistribution of the error over the elemental edges. The adaptive strategy is believed to be the first 3-D implementation of an improved spring analogy-based algorithm originally applied on 2-D quadrilateral meshes.

This effort represents the CFD Laboratory's/P&WC's first foray into 3-D grid adaptation and the reasons for choosing a mesh movement scheme over mesh enrichment methods was its portability into an existing code (*NS3D*) with relatively little change in data structure. However, some difficulties were encountered in making the extension to three-dimensions. One was the need for a robust search algorithm for interpolation of the metric tensor from the background to current meshes as well as for interpolation of the old solution on the newly adapted grid. Another challenge was presented in making the mesh movement applicable to both hexahedral and tetrahedral elements. Unlike quadrilateral grids in two-dimensions, it was observed that control of the mesh quality had to be much tighter. A narrower range of adaptive parameters had to be applied, thus limiting the range of nodal displacement permitted.

The effectiveness of the adaptive scheme to equidistribute the interpolation error over the edges of tetrahedral and hexahedral meshes has been shown on an analytical test case where near Gaussian distributions of the error were obtained. It was further demonstrated on non-adapted and adapted hexahedral grids that the estimated error closely follows the exact error.

The UTRC second stator validation case in which only volume nodes were moved showed that with very limited mesh adaptation by three-dimensional standards (3 adaptive cycles, each consisting of 200 iterations of the movement scheme), one can still obtain significantly improved results in comparison to experimental and non-adapted results.

The adapted tetrahedral and hexahedral grids for the transonic 10% circular arc case clearly contained elements which were highly stretched and aligned along the shock. Sharper shock structures were also obtained compared to other numerical results using other schemes.

The aim of the test case involving laminar viscous flow through a square duct with a 90° bend was to determine the size of a non-adapted grid that would be equivalent to that of a given adapted grid for the same level of solution error. Based on the assumption that the finest adapted grid studied would provide the best solution, it was determined that for the same level of error, that adapted 21x21x61 (26,901 nodes) hexahedral grid was equivalent to a non-adapted grid of approximately 270,000 nodes. One achieved not only the same level of solution error by solving and adapting on a much coarser grid, but also a four to five-fold reduction in solution time.

The can combustor test case exemplifies the necessity of coupling an adaptive mesh movement scheme to automatic tetrahedral mesh generator. The initial non-adapted grid, generated with isotropic elements, was not suitable for resolving the flow features. By applying only one mesh adaptation cycle, a converged solution was reached using the same amount of artificial dissipation as applied on the original grid, but for which no converged solution could be obtained.

It is worth remarking on a few observations common to the results of most of the test cases. All test cases except the final one revealed that the flow solver required lower amounts of artificial dissipation for solution on the final adapted grids. The convergence histories for the nodal displacement algorithm for all cases showed that an error threshold is reached below which the error can no longer be decreased. This is to be expected of minimum error mesh problems. However, with full mesh adaptation, which includes all strategies, one has the capability of possibly lowering this error threshold.

7.2 Future Work

The error estimate in this work is based only on one selected solution variable. Yet, a true characterization of the error would necessarily require information from all implicated flow variables. Future work on extending the error estimate to encompass several flow variables would have to account for the varying orders of magnitude and dimensions.

The present adaptive method can improve its applicability by considering turbulent flows. In such a scenario, the mesh movement strategy could be modified to move nodes normal to the wall to maintain a user-specified y+ value. Along the same lines of thought, the adaptive procedure should be extended to operate on prismatic elements which would be beneficial for turbulence modeling. The node redistribution scheme for prismatic layers would permit the clustering of nodes in the normal direction to better resolve the viscous stresses and a certain y+ value specified by the user could be maintained.

As demonstrated by the final test case involving the wing-pylon-nacelle configuration, there are limitations to applying this scheme on complex 3-D flow problems and geometries. Using isotropic tetrahedral grid generation to properly capture the complexity of the geometry and flow phenomena often associated with these cases would lead to an unrealistically large grid size. The anisotropic nodal displacement method, as presented in this work, is dependent on the size of the initial non-adapted mesh. If the initial mesh is too coarse, nodes are moved away from one flow region to another resulting in poor resolution of certain flow features. Furthermore, as the complexity of the geometry increases, the mesh quality constraints imposed on the node movement strategy limit the possible range of grid point displacement. It is clear that in the solution of complex 3-D geometries involving sharp directional flow features an anisotropic mesh adaptation procedure which incorporates *all* strategies such as refinement/coarsening, nodal displacement and edge swapping, would be a more practical approach and one which would lead to a reasonable grid size. This enhancement of strategies would eliminate the dependence of the final adapted solution on the choice of an initial mesh and allow the possibility of lowering the error threshold.

References

1. J. T. Oden and A. K. Noor, "Advances in Adaptive Improvements: A Survey of Adaptive Finite Element Methods in Computational Mechanics", *State-of-the-Art-Survey on Computational Mechanics*, ASME, N. Y. 1989.

2. Oden, J. T. and Brauchli, H. J., "On the Calculation of Consistent Stress Distributions in Finite Element Approximations", *Intl. J. Num. Meth. Engrg.*, Vol. 3, pp. 317-325, 1971.

3. Hinton, E. and Campbell, J. S., "Local and Global Smoothing of Discontinuous Finite Element Functions Using a Least Squares Method", *Intl. J. Num. Meth. Engrg.*, Vol. 8, pp. 461-480, 1974.

4. O. C. Zienkiewicz and J. Z. Zhu, "A Simple Error Estimator and Adaptive Procedure for Practical Engineering Analysis", *Intl. J. Num. Engrg.*, Vol. 24, pp. 337-357, 1987

5. O. C. Zienkiewicz and J. Z. Zhu, "The Superconvergent Patch Recovery and A Posterioiri Error Estimates. Part I: The Recovery Technique", *Intl. J. Num. Engrg.*, Vol. 33, pp. 1331-1364, 1992.

6. Ainsworth, M., Zhu, J. Z., Craig, A. W. and Zienkiewicz, O. C. "Analysis of the Zienkiewicz-Zhu A-Posteriori Error Estimator in the Finite Element Method", *Intl. J. Num. Engrg*, Vol. 28 , pp. 2161-2171, 1989.

7. Zienkiewicz, O. C. and Zhu, J. Z., "A Posteriori Error Estimation and Three-Dimensional Automatic Mesh Generation", *Finite Elements in Analysis and Design*, Vol. 25, pp. 167-184, 1997.

8. Bank, R. E. and Weiser, A., "Some A Posteriori Error Estimators for Elliptic Partial Differential Equations", *Math. Comp.*, Vol. 44, pp. 283-301, 1985.

9. Bank, R. E. and Welfert, B. D., 'A Posteriori Error Estimates for the Stokes Equations: A Comparison", *Comp. Meth. Appl. Mech. Engrg.*, Vol. 82, pp. 323-340, 1990.

10. Hawken, D. F., Gottlieb, J. J. and Hansen, J. S., "Review of Some Adaptive Node-Movement Techniques in Finite-Element and Finite-Difference Solutions of Partial Differentail Equations", *J. Comp. Physics*, Vol. 95, pp. 254-302, 1991.

11. Kallinderis, Y. and Baron, J. R., "A New Adaptive Algortihm for Turbulent Flows", *Comp. Fluids*, Vol. 21, No. 1, ppp. 77-96, 1992.

12. Palmerio, B., "A Two-Dimensional FEM Adaptive Moving-Node Method for Steady Euler Flow Simulations", *Comp. Meth. Appl. Mech. and Engrg* , Vol. 71, pp. 315-340 , 1988.

13. Fortin, M., Habashi, W. G., Dompierre, J., Vallet, M.-G., Bourgault, Y., and Ait Ali Yahia, D., "Anisotropic Mesh adaptation: Towards a User-

Independent, Mesh-Independent and Solver-Independent CFD, Part I. General Principles", *Intl. J. Num. Meth. Fluids,* (submitted)

14. Babuska, I., "The *p* and *h-p* Versions of the Finite Element Method, Basic Principles and Properties", *SIAM Review,* Vol. 36, No. 4, pp. 578-632, 1994.

15. Oden, J. T. and Demkowicz, L., "*h-p* Adaptive Finite Element Methods in Computational Fluid Dynamics", *Comp. Meth. Appl. Mech. and Engrg.,* Vol. 89, pp. 1-40, 1991.

16. Oden, J. T. and Wu, W., "An *h-p* Adaptive Strategy for Finite Elemetn Approximations of the Navier-Stokes Equations", *Intl. J. Num. Meth. Fluids,* Vol. 20, pp. 831-851, 1995.

17. Löhner, R., "Automatic Unstructured Grid Generators", *Finite Elements in Analysis and Design,* Vol. 25, pp. 111-134, 1997.

18. Borouchaki, H., George, P. L., Hecht, F., Laug, P. and Saltel, E., "Delaunay Generation Governed by Metric Specifications. Part I. Algorithms", *Finite Elements in Analysis and Design,* Vol. 25, pp. 61-83, 1997.

19. Castro-Díaz, M. J., Hecht, F., Mohammadi, B., and Pironneau, O., "Anisotropic Unstructured Mesh Adaptation for Flow Simulations", *Intl. J. Num. Meth. Fluids,* Vol. 25, pp. 475-491, 1997.

20. Mavriplis, D. J., "Adaptive Mesh Generation for Viscous Flows Using Delaunay Triangulation", *J. Comp. Physics,* Vol. 90 , pp. 271-291, 1990.

21. Vallet, M.-G., "Génération de Maillages Éléments Finis Anisotropes et Adaptatifs", Doctoral Thesis, Paris VI, 1992.

22. Mavriplis, D., "Unstructured Mesh Generation and Adaptivity", Computational Fluid Dynamics Lectures Series 1995-03, von Karman Institute for Fluid Dynamics, Rhode-Saint-Genèse, Belgium, March 13-17, 1995.

23. Nakahashi, K., "Semi-Unstructured, Adaptive Grid Approach for External Viscous Flow Computations", *Proc. of the 5th Intl, Symp. on Computational Fluid Dynamics,* Vol. II, JSCFD, 1993.

24. Khawaja, A., Kallinderis, Y., and Parthasarathy, "Implementation of Adaptive Hybrid Grids for 3-D Turbulent Flows", *AIAA* 96-0026, 34th Aerospace Sciences Meeting, Reno. NV., Jan. 1996.

25. Kallinderis, Y., Khawaja, A. and McMorris, H., "Hybrid Prismatic-Tetrahderal Grid Generation For Complex Geometries, *AIAA* 95-0211, 33rd Aerospace Sciences Meeting, Reno, NV., Jan. 1995.

26. Tchon, K.-F., Hirsch, C. and Schneiders, R., "Octree-Based Hexahedral Mesh Generation for Viscous Flow Simulations", *AIAA* 97-1980, 13th AIAA Computation Fluid Dynamics Conference, June 29-July 2, 1997, Snowmass Village, CO.

27. Demkowicz, L., Oden, J. T., Rachowicz, W., and Hardy, O. "Toward a Universlal *h-p* AdaptiveFinite Element Strategy, Part 1. Constrained Approximation and Data Structure", *Comp. Meth. Appl. Mech. and Engrg*, Vol. 77, pp. 79-112, 1989.

28. Tan, Z. and Varghese, P., "Directionally Adaptive Finite Element Method for Multidimensional Euler and Navier-Stokes Equations", *AIAA* 93-3320, 1993.

29. Davis, R. L. and Dannenhoffer, J. F., "3-D Adaptive Grid-Embedding Euler Technique", *AIAA* 93-0330, 31st Aerospace Sciences Meeting, Reno, NV., Jan. 1993.

30. Quirk, J. J.,"A Cartesian Grid Approach with Hierachicial Refinement for Compressible Flows", *Computational Fluid Dynamics '94*, pp. 200-209, 1994.

31. Aftosmis, M. J., "Viscous Flow Simulation Using an Upwind Method for Hexahedral Based Adaptive Meshes", *AIAA* 93-0772, 31st Aerospace Sciences Meeting, Reno, NV., Jan. 1993.

32. Nakahashi, K. and Deiwert, G. S., "Self-Adaptive Grid Method with Application to Airfoil Flow", *AIAA J.*, Vol. 25, No. 4, pp. 513-520, 1987.

33. Davies, C. B. and Venkatapathy, E., "Application of a Solution Adaptive Grid Scheme to Complex Three-Dimensional Flows", *AIAA J.*, Vol. 30, No. 9, 1992.

34. Löhner, R., Morgan, K. and Zienkiewicz, O. C., "An Adaptive Finite Element Procedure for Compressible High Speed Flows", *Comp. Meth. Appl. Mech. Engrg.*, Vol. 51, pp. 441-465, 1985.

35. Löhner, R., "An Adaptive Finite Element Scheme for Transient Problems in CFD" , *Comp. Meth. Appl. Mech. Engrg.*, Vol. 61, pp. 323-338, 1987.

36. Löhner, R. and Baum, J., "Adaptive *h*-Refinement on 3D Unstructured Grids for Transient Problems, *Intl. J. Num. Fluids*, Vol. 14, pp. 1407-1419, 1992.

37. Simpson, B., "Anisotropic Mesh Transformations and Optimal Error Control", August 1992.

38. D'Azevedo, E. F. and Simpson, R. B. "On Optimal Triangular Meshes for Minimizing the Gradient Error", *Nümerische Mathematik*, Vol. 59, pp. 321-348, 1991.

39. D'Azevedo, E. F. and Simpson, R. B., "Triangular Coordinate Transformations", *SIAM J. Sci. and Stat. Comp.*, Vol. 10, pp. 1063-1075, 1989.

40. Peraire, J., Vahdati, M., Morgan, K. and Zienkiewicz, O. C., "Adaptive Remeshing for Compressible Flow Computations", *J. Comp. Physics,* Vol. 72, pp. 449-466, 1987.

41. Morgan, K., Peraire, J., Peiro, J. and O. Hassan, "The Computation of Three-Dimensional Flows Using Unstructured Grids", *Comp. Meth. Appl. Mech. Engrg.,* Vol. 87, pp. 335-352, 1991.

42. Fortin, M., Vallet, M. G., Poirier, D. and Habashi, W. G., "Error Estimation and Directionally-Adaptive Meshing", *AIAA* 94-2211, 25th AIAA Fluid Dynamics Conference, Colorado Springs, June 1994.

43. Vilsmeier, R. and Hanel, D., "Adaptive Solutions for Compressible Flows on Unstructured, Strongly Anisotropic Grids", *Computational Fluid Dynamics '92,* Volume 2, Ch. Hirsch et al. (Eds), Elsevier Science Publishers B.V. , 1992.

44. Tilch, R., "Unstructured Grids, Adaptive Remeshing and Mesh Generation for Navier-Stokes", *Twelfth International Conference on Numerical Methods in Fluid Dynamics,* Springer-Verlag, 1991.

45. Kornhuber, R. and Roitzsch, R., "On Adaptive Grid Refinement in the Presence of Internal or Boundary Layers", *Impact of Computing in Science and Engineering,* Vol. 2, pp. 40-72, 1990.

46. Hassan, O., Probert, E. J., Weatherill, N. P., Marchant, M. J., Morgan, K., and Marcum, D., "The Numerical Simulation of Viscous Transonic flow

using Unstructured Grids", *AIAA* 94-2346, 25th AIAA Fluid Dynamics Conference, Colorado Springs, June 1994.

47. Pirzadeh, S., "Unstructured Viscous Grid Generation by Advancing-Layers Method", *AIAA* 93-3453, AIAA 11th Applied Aerodynamics Conference, Monterey, CA, 1993.

48. Löhner, R., "Matching Semi-Structured and Unstructured Grids for Navier-Stokes Calculations", *AIAA* 93-3348, 1993.

49. Holmes, D. G. and Connell, S. D. "Solution of the 2D Navier-Stokes Equations on Unstructured Adaptive Grids", *AIAA* 89-1932, 1989.

50. Ramakrishnan, R. , Bey, K. S., and Thorton, E. A., "Adaptive Quadrilateral and Triangular Finite Element Scheme for Compressible Flows", *AIAA J.*, Vol. 28, pp. 51-59, 1990.

51. Gnoffo, P. A., "A Vectorized, Finite-Volume, Adaptive-Grid Algorithm for Navier-Stokes Calculations", *Numerical Grid Generation* (Ed. Joe F. Thompson), Elsevier Science Publishing, pp. 819-835, 1982.

52. Gnoffo, P. A., "A Finite-Volume, Adaptive Grid Algorithm Applied to Planetary Entry Flowfields", *AIAA J.*, Vol. 21, No. 9, pp. 1249-1254, 1983.

53. Löhner, R., Morgan, K., and Zienkiewicz, O. C., "Adaptive Grid Refinement for the Compressible Euler Equations", *Accuracy Estimates and Adaptive Refinement in Finite Element Computations*, (Eds.

Babuska, I., Zienkiewicz, O. C., Gago, J., and Oliveira, A.), John Wiley & Sons, 1986.

54. Catherall, D. "The Adaptation of Structured Grid to Numerical Solutions for Transonic Flow", *Intl J. Num. Meth. Engrg.*, Vol. 32, pp. 921-937, 1991.

55. Diaz, A. R., Kikuchi, N. and Taylor, J. E., "A Method of Grid Optimization for Finite Element Methods, *Comp. Meth. Appl. Mech. Engrg.*, Vol. 41, pp. 29-45, 1983.

56. Oden, J. T., Strouboulis, T. and Devloo, P., "Adaptive Finite Element Methods for the Analysis of Inviscid Compressible Flow: Part I. Fast Refinement/Unrefinement and Moving Mesh Methods for Unstructured Meshes", *Comp. Meth. Appl. Mech. Engrg.*, Vol. 59, pp. 327-362, 1986.

57. Ciarlet, P. G., *Numerical Analysis of the Finite Element Method*, Les Presses de L'Université de Montréal, 1976.

58. *NS3D*: Navier-Stokes in 3D Users' Guide, Version 4.2, Pratt and Whitney Canada, Sept. 1994.

59. McBride, B. J., Gordon, S., and Reno, M. A.,"Coefficients for Calculating Thermodynamic and Transport Properties of Individual Species", *NASA TM 4513*, Oct. 1993.

60. Baker, A. J., *Finite Element Computational Fluid Mechanics*, John Wiley and Sons, 1983.

61. Dhatt, G. and Touzot, G., *The Finite Element Method Displayed*, John Wiley and Sons, 1984.

62. Hughes, T. J. R., *The Finite Element Method: Linear Static and Dynamic Finite Elements*, Prentice-Hall, 1987.

63. Baruzzi, G. S., *A Second Order Finite Element Method for the Soluton of the Transonic Euler and Navier-Stokes Equations*, Ph.D. thesis, 1995.

64. Sleiman, M., *A Time-Accurate Finite Element Solution of the Navier-Stokes Equations*, M.A.Sc. Thesis, 1995.

65. Reddy, J. N. *Applied Functional Analysis and Variational Methods in Engineering*, Krieger Publishing Co., 1991.

66. Tam, A., Habashi, W. G., Ait Ali Yahia, D., Robichaud, M. P., Sleiman, M., and Vallet, M.-G., "A 3-D Adaptive Finite Element Method for Turbomachinery", *AIAA* 96-2659, 32nd Joint Propulsion Conference, Orlando, FL., July 1996.

67. Sleiman, M., Tam, A., Robichaud, M. P., Peeters, M. F., Habashi, W. G., and Fortin, M., "Turbomachinery Multistage Simulation by a Finite Element Adaptive Approach", Paper No. 96-GT-418, 41st ASME Gas Turbine and Aeroengine Congress, Birmingham, U.K., June 1996.

68. Tam, A., Robichaud, M. P., Tremblay, P., Habashi, W. G., Hohmeyer, M., Guèvremont, G., Peeters, M. F., Germain, P., "A 3-D Adaptive Anisotropic Method for External and Internal Flows", *AIAA* 98-0771, 36th Aerospace Sciences Meeting, Reno, NV., Jan. 1998.

69. Ait Ali Yahia, D., Habashi, W. G., Tam, A., Vallet, M.-G., and Fortin, M., "A Directionally Adaptive Methodology Using an Edge-Based Error Estimate on Quadrilateral Grids", *Intl. J. Num. Meth. Fluids*, Vol. 23, pp. 673-690, 1996.

70. *ICEM-CFD HEXA Training Manual*, Version 2.4, Sept. 1996.

71. *ICEM-CFD / CAE Unstructured Grid Generation*, Vol. 2., Nov. 1994.

72. Löhner, R. , "Robust, Vectorized Search Algorithms for Interpolation on Unstructured Grids", *J. Comp. Physics*, Vol. 118, pp. 380-387, 1995.

73. Dring, R. P., VanSeters, R. J. and Zacaharias, R. M., "A Three-Dimensional Navier-Stokes Calculation Applied to an Axial Compressor for Rotor and Stator", 93-GT-113, ASME International Gas Turbine and Aeroengine Congress, May 1993.

74. Ni, R., "A Multiple Grid Scheme for Solving the Euler Equations", *AIAA* 81-1025, 1981.

75. Humphrey, J. A. C., Taylor, A. M. K. and Whitelaw, J. H., "Laminar Flow in a Square Duct of Strong Curvature", *J. Fluid Mech.*, Vol. 83, pp. 509-527, 1974.

76. Lin, F. B. and Sotiropoulos, F., "Assessment of Artificial Models for Three-Dimensional Incompressible Flow Solutions", *J. Fluids Engrg.*, Vol. 119, pp. 331-340, 1997.

77. Yeo, R. W., and Wood, P. E., and Hrymak, A. N., "A Numerical Study of Laminar 90-Degree Bend Duct Flow With Different Discretization Schemes", *J. Fluids Engrg.*, Vol. 113, pp. 563-568, 1991.

78. Moore, M. , "Hydrogen Combustor Analysis Using Unstructured Adapted Grids", Internal Memo (Pratt & Whitney Canada), March 20, 1997.

79 Carlson, J. R. and Compton, W. B., "An Experimental Investigation of Nacelle-Pylon Installation on an Unswept Wing at Subsonic and Transonic Speeds", *NASA TP-2246*, 1984.

80. Lord, W. K. and Zysman, S. H., "VSAERO Analysis of a Wing/Pylon/Nacelle Configuration", *AIAA* 86-1523, AIAA 22nd Joint Propulsion Conference, June 16-18, 1986.

81. Dring , R. P. and Joslyn, H. D., "Through-Flow Analysis of a Multistage Compressor: Part I - Aerodynamic Input", *Journal of Turbomachinery*, Vol. 68, pp. 17-2, 1986.

82. White, F. M.,*Viscous Fluid Flow*, Second Edition, McGraw-Hill, 1991.

Appendix A

Weak Galerkin Formulation of the Navier-Stokes Equations

The weak Galerkin formulation of the system of equations consisting of the continuity (2.2) and the Navier-Stokes momentum equations (2.3) in Cartesian coordinates will be demonstrated. Each equation is multiplied by a weight function, W, which is identical to the shape function, and integrated over the volume:

$$\int_V W\left[C_t + D_x + E_y + F_z\right] dV = 0 \tag{A.1}$$

where

$$C = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \end{bmatrix} \tag{A.2}$$

$$D = \begin{bmatrix} \rho u - \lambda\left(p_x - \bar{p}_x\right) \\ \rho u^2 + p - \frac{2}{3}\frac{\mu}{Re}\left(2u_x - v_y - w_z\right) \\ \rho uv - \frac{\mu}{Re}\left(u_y - v_x\right) \\ \rho uw - \frac{\mu}{Re}\left(u_z - w_x\right) \end{bmatrix} \tag{A.3}$$

$$E = \begin{bmatrix} \rho v - \lambda\left(p_y - \bar{p}_y\right) \\ \rho uv - \frac{\mu}{Re}\left(v_x + u_y\right) \\ \rho v^2 + p - \frac{2}{3}\frac{\mu}{Re}\left(2v_y - u_x - w_z\right) \\ \rho vw - \frac{\mu}{Re}\left(v_z + w_y\right) \end{bmatrix} \tag{A.4}$$

$$F=\begin{bmatrix} \rho w-\lambda\left(p_z-\bar{p}_z\right) \\ \rho uw-\frac{\mu}{Re}\left(w_x+u_z\right) \\ \rho vw-\frac{\mu}{Re}\left(w_y+v_z\right) \\ \rho w^2+p-\frac{2}{3}\frac{\mu}{Re}\left(2w_z-u_x-v_y\right) \end{bmatrix} \tag{A.5}$$

Subsequent integration by parts yields the weak Galerkin form of the system of equations

$$\int_V \left[WC_t+W_xD+W_yE+W_zF\right]dV=\int_S WGdS \tag{A.6}$$

where the surface integral term, G, is of the form:

$$G=\begin{bmatrix} \left[\rho\bar{V}\bullet\lambda\left(\nabla p-\nabla\bar{p}\right)\right]\bullet\bar{n} \\ \rho u\left(\bar{V}\bullet\bar{n}\right)+\left[p-\frac{2}{3}\frac{\mu}{Re}\left(2u_x-v_y-w_z\right)\right]\frac{dydz}{dS}-\frac{\mu}{Re}\left[\left(v_x+u_y\right)\frac{dxdz}{dS}+\left(w_x+u_z\right)\frac{dxdy}{dS}\right] \\ \rho v\left(\bar{V}\bullet\bar{n}\right)+\left[p-\frac{2}{3}\frac{\mu}{Re}\left(2v_y-u_x-w_z\right)\right]\frac{dxdz}{dS}-\frac{\mu}{Re}\left[\left(v_x+u_y\right)\frac{dydz}{dS}+\left(w_y+v_z\right)\frac{dxdy}{dS}\right] \\ \rho w\left(\bar{V}\bullet\bar{n}\right)+\left[p-\frac{2}{3}\frac{\mu}{Re}\left(2w_z-u_x-v_y\right)\right]\frac{dxdy}{dS}-\frac{\mu}{Re}\left[\left(w_x+u_z\right)\frac{dxdz}{dS}+\left(w_y+v_z\right)\frac{dxdz}{dS}\right] \end{bmatrix} \tag{A.7}$$

Appendix B
Time Discretization and Newton Linearization of the Navier-Stokes Equations

B.1 Time Discretization

The weak Galerkin form of the Navier-Stokes system of equations (A.6-A.7) is discretized in time using a first order Euler backward scheme as follows:

$$\int_V \left[W C_t' + W_x D' + W_y E' + W_z F' \right] dV - \int_S W G' dS = 0 \tag{B1.1}$$

where

$$C_t' = \begin{bmatrix} \frac{1}{\Delta t}\left(\rho^t - \rho^{t-1}\right) \\ \frac{1}{\Delta t}\left((\rho u)^t - (\rho u)^{t-1}\right) \\ \frac{1}{\Delta t}\left((\rho v)^t - (\rho v)^{t-1}\right) \\ \frac{1}{\Delta t}\left((\rho w)^t - (\rho w)^{t-1}\right) \end{bmatrix} \tag{B1.2}$$

$$D' = \begin{bmatrix} \left[\rho u - \lambda(p_x - \overline{p}_x)\right]^t \\ \left[\rho u^2 + p - \frac{2}{3}\frac{\mu}{Re}\left(2u_x - v_y - w_z\right)\right]^t \\ \left[\rho u v - \frac{\mu}{Re}\left(u_y - v_x\right)\right]^t \\ \left[\rho u w - \frac{\mu}{Re}\left(u_z - w_x\right)\right]^t \end{bmatrix} \tag{B1.3}$$

$$E' = \begin{bmatrix} \left[\rho v - \lambda\left(p_y - \bar{p}_y\right)\right]^t \\ \left[\rho u v - \frac{\mu}{Re}\left(v_x + u_y\right)\right]^t \\ \left[\rho v^2 + p - \frac{2}{3}\frac{\mu}{Re}\left(2v_y - u_x - w_z\right)\right]^t \\ \left[\rho v w - \frac{\mu}{Re}\left(v_z + w_y\right)\right]^t \end{bmatrix} \tag{B1.4}$$

$$F' = \begin{bmatrix} \left[\rho w - \lambda\left(p_z - \bar{p}_z\right)\right]^t \\ \left[\rho u w - \frac{\mu}{Re}\left(w_x + u_z\right)\right]^t \\ \left[\rho v w - \frac{\mu}{Re}\left(w_y + v_z\right)\right]^t \\ \left[\rho w^2 + p - \frac{2}{3}\frac{\mu}{Re}\left(2w_z - u_x - v_y\right)\right]^t \end{bmatrix} \tag{B1.5}$$

$$G' = \begin{bmatrix} \left\{\left[\rho\vec{V} - \lambda\left(\nabla p - \overline{\nabla p}\right)\right]\bullet\vec{n}\right\}^t \\ \left\{\rho u\left(\vec{V}\bullet\vec{n}\right) + \left[p - \frac{2}{3}\frac{\mu}{Re}\left(2u_x - v_y - w_z\right)\right]\frac{dydz}{dS} - \frac{\mu}{Re}\left[\left(v_x + u_y\right)\frac{dxdz}{dS} + \left(w_x + u_z\right)\frac{dxdy}{dS}\right]\right\}^t \\ \left\{\rho v\left(\vec{V}\bullet\vec{n}\right) + \left[p - \frac{2}{3}\frac{\mu}{Re}\left(2v_y - u_x - w_z\right)\right]\frac{dxdz}{dS} - \frac{\mu}{Re}\left[\left(v_x + u_y\right)\frac{dydz}{dS} + \left(w_y + v_z\right)\frac{dxdy}{dS}\right]\right\}^t \\ \left\{\rho w\left(\vec{V}\bullet\vec{n}\right) + \left[p - \frac{2}{3}\frac{\mu}{Re}\left(2w_z - u_x - v_y\right)\right]\frac{dxdy}{dS} - \frac{\mu}{Re}\left[\left(w_x + u_z\right)\frac{dxdz}{dS} + \left(w_y + v_z\right)\frac{dxdz}{dS}\right]\right\}^t \end{bmatrix} \tag{B1.6}$$

B.2 Newton Linearization

The terms of the discretized matrix and residual equations (2.24-2.27) will be written in detail below:

element influence matrices and residual from x-momentum equation:

$$\left[k^{\rho u}{}_{ij}\right]_{\rho u}=\int_V\left[\frac{1}{\Delta t}\left(W_iN_j\right)-N_j\left(2u^n\frac{\partial W_i}{\partial x}+v^n\frac{\partial W_i}{\partial y}+w^n\frac{\partial W_i}{\partial x}\right)\right.$$
$$\left.+\frac{\mu}{Re\rho_j}\left(\frac{4}{3}\frac{\partial W_i}{\partial x}\frac{\partial N_j}{\partial x}+\frac{\partial W_i}{\partial y}\frac{\partial N_j}{\partial y}+\frac{\partial W_i}{\partial z}\frac{\partial N_j}{\partial z}\right)\right]dV \tag{B2.1}$$

$$\left[k^{\rho v}{}_{ij}\right]_{\rho u}=-\int_V\left[N_ju^n\frac{\partial W_i}{\partial y}+\frac{\mu}{Re\rho_j}\left(\frac{2}{3}\frac{\partial W_i}{\partial x}\frac{\partial N_j}{\partial y}-\frac{\partial W_i}{\partial y}\frac{\partial N_j}{\partial x}\right)\right]dV \tag{B2.2}$$

$$\left[k^{\rho w}{}_{ij}\right]_{\rho u}=-\int_V\left[N_ju^n\frac{\partial W_i}{\partial z}+\frac{\mu}{Re\rho_j}\left(\frac{2}{3}\frac{\partial W_i}{\partial x}\frac{\partial N_j}{\partial z}-\frac{\partial W_i}{\partial z}\frac{\partial N_j}{\partial x}\right)\right]dV \tag{B2.3}$$

$$\left[k^{p}{}_{ij}\right]_{\rho u}=-\int_V\left[N_j\frac{\partial W_i}{\partial x}\right]dV \tag{B2.4}$$

$$\mathrm{Re}\,s^n_{\rho u}=\sum_{e=1}^{nelem}\int_V\left\{p^n\frac{\partial W_i}{\partial x}+\left[\rho u^2-\frac{2\mu}{3Re}\left(2\frac{\partial u}{\partial x}-\frac{\partial v}{\partial y}-\frac{\partial w}{\partial z}\right)-\mu^{artR}\frac{\partial u}{\partial x}\right]^n\frac{\partial W_i}{\partial x}\right.$$
$$\left.+\left[\rho uv-\frac{\mu}{Re}\left(\frac{\partial v}{\partial x}+\frac{\partial u}{\partial y}\right)-\mu^{artR}\frac{\partial u}{\partial y}\right]^n\frac{\partial W_i}{\partial y}+\left[\rho uw-\frac{\mu}{Re}\left(\frac{\partial w}{\partial x}+\frac{\partial u}{\partial z}\right)-\mu^{artR}\frac{\partial u}{\partial z}\right]^n\frac{\partial W_i}{\partial z}\right\}dV$$
$$+\int_S W_i\left\{\left[\rho u^2+p+\frac{2\mu}{3Re}\left(\frac{\partial v}{\partial y}+\frac{\partial w}{\partial z}\right)-\frac{\mu}{3Re}\frac{\partial u}{\partial x}\right]^n\right\}n_xdS$$
$$+\int_S W_i\left\{\left[\rho uv-\frac{\mu}{Re}\frac{\partial v}{\partial x}\right]^n\right\}n_ydS+\int_S W_i\left\{\left[\rho uw-\frac{\mu}{Re}\frac{\partial w}{\partial x}\right]^n\right\}n_zdS \tag{B2.5}$$

element influence matrices and residual from y-momentum equation:

$$\left[k^{\rho u}{}_{ij}\right]_{\rho v}=-\int_V\left[N_jv^n\frac{\partial W_i}{\partial x}+\frac{\mu}{Re\rho_j}\left(\frac{2}{3}\frac{\partial W_i}{\partial y}\frac{\partial N_j}{\partial x}-\frac{\partial W_i}{\partial x}\frac{\partial N_j}{\partial y}\right)\right]dV \tag{B2.6}$$

$$\left[k^{\rho v}{}_{ij}\right]_{\rho v}=\int_V\left[\frac{1}{\Delta t}\left(W_iN_j\right)-N_j\left(2v^n\frac{\partial W_i}{\partial y}+u^n\frac{\partial W_i}{\partial x}+w^n\frac{\partial W_i}{\partial z}\right)\right.$$
$$\left.+\frac{\mu}{Re\rho_j}\left(\frac{4}{3}\frac{\partial W_i}{\partial y}\frac{\partial N_j}{\partial x}+\frac{\partial W_i}{\partial x}\frac{\partial N_j}{\partial x}+\frac{\partial W_i}{\partial z}\frac{\partial N_j}{\partial z}\right)\right]dV \tag{B2.7}$$

$$\left[k^{\rho w}{}_{ij}\right]_{\rho v}=-\int_V\left[N_jv^n\frac{\partial W_i}{\partial z}+\frac{\mu}{Re\rho_j}\left(\frac{2}{3}\frac{\partial W_i}{\partial y}\frac{\partial N_j}{\partial z}-\frac{\partial W_i}{\partial z}\frac{\partial N_j}{\partial y}\right)\right]dV \tag{B2.8}$$

$$\left[k^{p}{}_{ij}\right]_{\rho v}=-\int_V\left[N_j\frac{\partial W_i}{\partial y}\right]dV \tag{B2.9}$$

$$\mathrm{Re}s^n_{\rho v} = \sum_{e=1}^{nelem} \int_V \left\{ p^n \frac{\partial W_i}{\partial y} + \left[\rho v^2 - \frac{2\mu}{3Re}\left(2\frac{\partial v}{\partial y} - \frac{\partial u}{\partial x} - \frac{\partial w}{\partial z} \right) - \mu^{artR} \frac{\partial v}{\partial y} \right]^n \frac{\partial W_i}{\partial y} \right.$$
$$\left. + \left[\rho uv - \frac{\mu}{Re}\left(\frac{\partial v}{\partial x} + \frac{\partial u}{\partial y} \right) - \mu^{artR} \frac{\partial v}{\partial x} \right]^n \frac{\partial W_i}{\partial x} + \left[\rho vw - \frac{\mu}{Re}\left(\frac{\partial w}{\partial y} + \frac{\partial v}{\partial z} \right) - \mu^{artR} \frac{\partial v}{\partial z} \right]^n \frac{\partial W_i}{\partial z} \right\} dV$$
$$+ \int_S W_i \left\{ \left[\rho v^2 + p + \frac{2\mu}{3Re}\left(\frac{\partial u}{\partial x} + \frac{\partial w}{\partial z} \right) - \frac{\mu}{3Re}\frac{\partial v}{\partial y} \right]^n \right\} n_y dS$$
$$+ \int_S W_i \left\{ \left[\rho uv - \frac{\mu}{Re}\frac{\partial u}{\partial y} \right]^n \right\} n_x dS + \int_S W_i \left\{ \left[\rho vw - \frac{\mu}{Re}\frac{\partial w}{\partial y} \right]^n \right\} n_z dS$$

(B2.10)

element influence matrices and residual from z-momentum equation:

$$\left[k^{\rho u}{}_{ij}\right]_{\rho w} = -\int_V \left[N_j w^n \frac{\partial W_i}{\partial x} + \frac{\mu}{Re\rho_j}\left(\frac{2}{3}\frac{\partial W_i}{\partial z}\frac{\partial N_j}{\partial x} - \frac{\partial W_i}{\partial x}\frac{\partial N_j}{\partial z} \right) \right] dV \qquad \text{(B2.11)}$$

$$\left[k^{\rho v}{}_{ij}\right]_{\rho w} = -\int_V \left[N_j w^n \frac{\partial W_i}{\partial y} + \frac{\mu}{Re\rho_j}\left(\frac{2}{3}\frac{\partial W_i}{\partial z}\frac{\partial N_j}{\partial y} - \frac{\partial W_i}{\partial y}\frac{\partial N_j}{\partial z} \right) \right] dV \qquad \text{(B2.12)}$$

$$\left[k^{\rho w}{}_{ij}\right]_{\rho w} = \int_V \left[\frac{1}{\Delta t}\left(W_i N_j\right) - N_j\left(2w^n \frac{\partial W_i}{\partial z} + u^n \frac{\partial W_i}{\partial x} + v^n \frac{\partial W_i}{\partial z} \right) \right.$$
$$\left. + \frac{\mu}{Re\rho_j}\left(\frac{4}{3}\frac{\partial W_i}{\partial z}\frac{\partial N_j}{\partial z} + \frac{\partial W_i}{\partial x}\frac{\partial N_j}{\partial x} + \frac{\partial W_i}{\partial y}\frac{\partial N_j}{\partial y} \right) \right] dV \qquad \text{(B2.13)}$$

$$\left[k^{p}{}_{ij}\right]_{\rho w} = -\int_V \left[N_j \frac{\partial W_i}{\partial z} \right] dV \qquad \text{(B2.14)}$$

$$\mathrm{Re}s^n_{\rho w} = \sum_{e=1}^{nelem} \int_V \left\{ p^n \frac{\partial W_i}{\partial z} + \left[\rho w^2 - \frac{2\mu}{3Re}\left(2\frac{\partial w}{\partial z} - \frac{\partial u}{\partial x} - \frac{\partial v}{\partial y} \right) - \mu^{artR} \frac{\partial w}{\partial z} \right]^n \frac{\partial W_i}{\partial z} \right.$$
$$\left. \left[\rho uw - \frac{\mu}{Re}\left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x} \right) - \mu^{artR} \frac{\partial w}{\partial x} \right]^n \frac{\partial W_i}{\partial x} + \left[\rho vw - \frac{\mu}{Re}\left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y} \right) - \mu^{artR} \frac{\partial w}{\partial y} \right]^n \frac{\partial W_i}{\partial y} \right\} dV$$
$$+ \int_S W_i \left\{ \left[\rho w^2 + p + \frac{2\mu}{3Re}\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \right) - \frac{\mu}{3Re}\frac{\partial w}{\partial z} \right]^n \right\} n_z dS$$
$$+ \int_S W_i \left\{ \left[\rho uw - \frac{\mu}{Re}\frac{\partial u}{\partial z} \right]^n \right\} n_x dS + \int_S W_i \left\{ \left[\rho vw - \frac{\mu}{Re}\frac{\partial v}{\partial z} \right]^n \right\} n_y dS$$

(B2.15)

Appendix C
Numerical Discretization of the Energy Equation

The weak Galerkin formulation, time discretization, and Newton linearization of the energy equation (2.4) will be demonstrated. Equation (2.4) is first multiplied by a weight function, W, which is identical to the interpolation function, and then integrated over the domain:

$$\int_V \left\{ \rho C_p \frac{DT_0}{Dt} - \frac{1}{PrRe} \nabla \bullet \left[\kappa \nabla \left(T_0 - \frac{\vec{V}^2}{2C_p} \right) \right] = Ec \frac{\partial p}{\partial t} + \frac{Ec}{Re} \nabla \bullet \left(\vec{V} \tau_{ij} \right) \right\} W dV = 0 \quad \text{(C.1)}$$

After integration by parts, the weak Galerkin form of eq. (C.1) can be written as:

$$\int_V \left\{ \left[\rho C_p \left(\frac{\partial T_0}{\partial t} + \vec{V} \bullet \nabla T_0 \right) - Ec \frac{\partial p}{\partial t} \right] W + \left[\frac{\kappa}{PrRe} \nabla \left(T_0 - \frac{\vec{V}^2}{2C_p} \right) + \frac{Ec}{Re} \vec{V} \tau_{ij} \right] \bullet \nabla W \right\} dV$$
$$- \int_S \left\{ \left[\frac{\kappa}{PrRe} \nabla T_0 + \frac{Ec}{Re} \vec{V} \tau_{ij} \right] \bullet \vec{n} \right\} W dS = 0 \quad \text{(C.2)}$$

The viscous dissipation term may be explicitly expressed as:

$$\nabla \bullet \vec{V} \tau_{ij} = \frac{\partial}{\partial x} \theta_1 + \frac{\partial}{\partial y} \theta_2 + \frac{\partial}{\partial z} \theta_3 \quad \text{(C.3)}$$

where

$$\begin{aligned} \theta_1 &= u\tau_{xx} + v\tau_{xy} + w\tau_{xz} \\ \theta_2 &= u\tau_{xy} + v\tau_{zy} + w\tau_{zz} \\ \theta_3 &= u\tau_{zx} + v\tau_{zy} + w\tau_{zz} \end{aligned} \quad \text{(C.4)}$$

The weak Galerkin form of the energy equation can be expanded as:

$$\int_V \left\{ \rho C_p \frac{\partial T_0}{\partial t} W + C_p \left[\rho u T_{0x} + \rho v T_{0y} + \rho w T_{0z} \right] W \right.$$
$$\left. + \frac{\kappa}{PrRe} \left[T_{0x} W_x + + T_{0y} W_y + T_{0z} W_z \right] \right\} dV + \int_S \left[\frac{\kappa}{PrRe} T_n \right] W dS$$
$$= \int_V \left\{ Ec \frac{\partial p}{\partial t} W + \frac{\kappa}{PrRe} \left[\left(\frac{\vec{V}^2}{2C_p} \right)_x W_x + \left(\frac{\vec{V}^2}{2C_p} \right)_y W_y + \left(\frac{\vec{V}^2}{2C_p} \right)_z W_z \right] \right\} dV \tag{C.5}$$
$$+ \frac{Ec}{Re} \int_S \left[\theta_1 n_x + \theta_2 n_y + \theta_3 n_z \right] W dS$$

The time-dependent term of equation (C.5) is then discretized using a first order Gear scheme (Euler backward) as follows:

$$\int_V \left\{ \left(\rho C_p \right)^t \frac{1}{\Delta t} \left[T_0^{\,t} - T_0^{\,t-1} \right] W + C_p \left[\rho u T_{0x} + \rho v T_{0y} + \rho w T_{0z} \right]^t W \right.$$
$$\left. + \left[\frac{\kappa}{PrRe} \left(T_{0x} W_x + + T_{0y} W_y + T_{0z} W_z \right) \right]^t \right\} dV + \int_S \left[\frac{\kappa}{PrRe} T_n \right]^t W dS$$
$$= \int_V \left\{ \frac{Ec}{\Delta t} \left[p^t - p^{t-1} \right] W - \frac{Ec}{Re} \left[\theta_1 W_x + \theta_2 W_y + \theta_3 W_z \right]^t \right.$$
$$\left. + \frac{\kappa}{PrRe} \left[\left(\frac{\vec{V}^2}{2C_p} \right)_x W_x + \left(\frac{\vec{V}^2}{2C_p} \right)_y W_y + \left(\frac{\vec{V}^2}{2C_p} \right)_z W_z \right]^t \right\} dV \tag{C.6}$$
$$+ \frac{Ec}{Re} \int_S \left[\theta_1 n_x + \theta_2 n_y + \theta_3 n_z \right]^t W dS$$

The Newton method is applied whereby the total temperature is expressed in delta form, $\Delta T_0 = T_0^{\,n+1} - T_0^{\,n}$, and the second order terms are neglected. Upon substitution of equation (2.16) into the Newton linearized equation, the delta form of the energy equation is assembled, over the elements of the domain, in terms of the nodal unknown, ΔT_0:

$$\sum_{e=1}^{nelem}\left[\sum_{j=1}^{ndperl}\left[k^{T_0}{}_{ij}\right]_{T_0}\Delta(T_0)_j\right]=-Res_{T_0} \tag{C.7}$$

where the element influence matrices and residual vectors from the energy equation take the form:

$$\begin{aligned}\left[k^{T_0}{}_{ij}\right]_{T_0}=\int_V\Bigg\{&\frac{1}{\Delta t}\rho C_p N_j W_i + C_p\left(\rho u\frac{\partial N_j}{\partial x}+\rho v\frac{\partial N_j}{\partial y}+\rho w\frac{\partial N_j}{\partial z}\right)W_i\\&+\frac{\kappa}{PrRe}\left(\frac{\partial N_j}{\partial x}\frac{\partial W_i}{\partial x}+\frac{\partial N_j}{\partial y}\frac{\partial W_i}{\partial y}+\frac{\partial N_j}{\partial z}\frac{\partial W_i}{\partial z}\right)\Bigg\}dV\end{aligned} \tag{C.8}$$

$$\begin{aligned}Res_{T_0}=\int_V\Bigg\{&C_p\left[\rho u T_{0x}+\rho v T_{0y}+\rho w T_{0z}\right]^n W\\&+\left[\frac{\kappa}{PrRe}\left(T_{0x}W_x++T_{0y}W_y+T_{0z}W_z\right)\right]^n\\&+\frac{Ec}{Re}\left[\theta_1 W_x+\theta_2 W_y+\theta_3 W_z\right]^n\\&+\frac{\kappa}{PrRe}\left[\left(\frac{\vec{V}^2}{2C_p}\right)_x W_x+\left(\frac{\vec{V}^2}{2C_p}\right)_y W_y+\left(\frac{\vec{V}^2}{2C_p}\right)_z W_z\right]^n\Bigg\}dV\\&+\int_S\left[\frac{\kappa}{PrRe}T_n\right]^n W dS+\frac{Ec}{Re}\int_S\left[\theta_1 n_x+\theta_2 n_y+\theta_3 n_z\right]^n W dS\end{aligned} \tag{C.9}$$